**Department of Computer Science**
**Course Structure (w.e.f. 2021)**
**Semester – I**

| Part | Component | Course Code | Course Title | Contact Hours/ Week | Credits | Max. Marks | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CIA | ESE | Total |
| I | Tamil /<br>French | 21ULTA11/<br>21ULFB11 | பொதுத்தமிழ் தாள் - 1 இக்கால இலக்கியம் (செய்யுள், இலக்கணம், இலக்கிய வரலாறு, உரைநடை, சிறுகதை)<br>Preliminary French and Commercial terms | 6 | 3 | 40 | 60 | 100 |
| II | General English | 21UGEN11 | Poetry, Prose, Extensive Reading and Communicative English – I | 6 | 3 | 40 | 60 | 100 |
| III | Core I | 21UCSC11 | C Programming | 4 | 4 | 40 | 60 | 100 |
| | Core Practical I | 21UCSCR1 | C Programming Lab | 4 | 2 | 40 | 60 | 100 |
| | Allied I | 21UCSA11 | Mathematics for Computer Science | 3 | 3 | 40 | 60 | 100 |
| | Allied Practical I | 21UCSAR1 | Office Automation Lab | 3 | 2 | 40 | 60 | 100 |
| IV | Skill Enhancement Course - I | 21UCSPE1 | Professional English for Computer Science - I | 2 | 2 | 20 | 30 | 50 |
| | Ability Enhancement Course - I | 21UAVE11 | Value Education | 2 | 2 | 20 | 30 | 50 |
| | | | **Total** | **30** | **21** | | | |

## Semester II

| Part | Component | Course Code | Course Title | Contact Hours/ Week | Credits | Max.Marks | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CIA | ESE | Total |
| I | Tamil /<br>French | 21ULTA21 /<br>21ULFB21 | பொதுத்தமிழ் தாள் 2 சமய இலக்கியங்களும் நீதி இலக்கியங்களும் (செய்யுள், இலக்கணம், இலக்கிய வரலாறு, உரைநடை, வாழ்க்கை வரலாறு)<br>Progressive French and Commercial Correspondence | 6 | 3 | 40 | 60 | 100 |
| II | General English | 21UGEN21 | Poetry, Prose, Extensive Reading and Communicative English –II | 6 | 3 | 40 | 60 | 100 |
| III | Core II | 21UCSC21 | C ++ Programming | 4 | 4 | 40 | 60 | 100 |
| | Core Practical II | 21UCSCR2 | C ++ Programming Lab | 4 | 2 | 40 | 60 | 100 |
| | Allied II | 21UCSA21 | Digital Electronics | 3 | 3 | 40 | 60 | 100 |
| | Allied Practical II | 21UCSAR2 | Open Source Multimedia Lab | 3 | 2 | 40 | 60 | 100 |
| | Skill Enhancement Course - II | 21UCSPE2 | Professional English for Computer Science – II | 2 | 2 | 20 | 30 | 50 |
| | Ability Enhancement Course - II | 21UAEV21 | Environmental Studies | 2 | 2 | 20 | 30 | 50 |
| | | | **Total** | **30** | **21** | | | |

## Semester III

| Part | Component | Course Code | Course Title | Contact Hours/ Week | Credits | Max. Marks | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CIA | ESE | Total |
| I | Tamil /<br>French | 21ULTA31<br>21ULFA31 | பொதுத்தமிழ் தாள் 3 : காப்பியங்களும் சிற்றிலக்கியங்களும் (செய்யுள், இலக்கணம், இலக்கிய வரலாறு, உரைநடை, புதினம்)<br>Advanced French Language | 6 | 4 | 40 | 60 | 100 |
| II | General English | 21UGEN31 | Poetry, Prose, Extensive Reading and Communicative English-III | 6 | 4 | 40 | 60 | 100 |
| III | Core III | 21UCSC31 | JAVA Programming | 4 | 4 | 40 | 60 | 100 |
| | Core Practical III | 21UCSCR3 | JAVA Programming Lab | 3 | 2 | 40 | 60 | 100 |
| | Allied III | 21UCSA31 | Data Structures | 3 | 3 | 40 | 60 | 100 |
| | Allied Practical III | 21UCSAR3 | Data Structures Lab | 2 | 1 | 40 | 60 | 100 |
| | Skill Based Elective | 21UCSS31/<br>21UCSS32 | Microprocessors/<br>E-Commerce | 2 | 2 | 20 | 30 | 50 |
| IV | NME I | 21UCSN31 | Introduction to Computers | 2 | 2 | 20 | 30 | 50 |
| | Ability Enhancement Course - III | 21UAWS31 | Women's Synergy | 2 | 2 | 20 | 30 | 50 |
| | Self Study/ On-line Course/ Internship (Compulsory) | 21UCSSS1 | Computer Architecture | | 2 | | 50 | 50 |
| Total | | | | 30 | 26 | | | |

**Semester IV**

| Part | Component | Course Code | Course Title | Hrs/Week | Credits | Max. Marks | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CIA | ESE | Total |
| I | Tamil /<br>French | 21ULTA41<br>21ULFA41 | பொதுத்தமிழ் தாள் 4: சங்க இலக்கியம்: (செய்யுள், இலக்கணம், இலக்கிய வரலாறு, உரைநடை, நாடகம்)<br>French Course and Literature | 6 | 4 | 40 | 60 | 100 |
| II | General English | 21UGEN41 | Poetry, Prose, Extensive Reading and Communicative English- IV | 6 | 4 | 40 | 60 | 100 |
| III | Core IV | 21UCSC41 | RDBMS with PHP and MySQL | 4 | 4 | 40 | 60 | 100 |
| | Core Practical IV | 21UCSCR4 | PHP & MySQL Lab | 3 | 2 | 40 | 60 | 100 |
| | Allied IV | 21UCSA41 | Big data Analytics | 3 | 3 | 40 | 60 | 100 |
| | Allied Practical IV | 21UCSAR4 | Web designing Lab | 2 | 1 | 40 | 60 | 100 |
| | Skill Based Elective | 21UCSS41/<br>21UCSS42 | DTP Lab/<br>Cyber Security | 2 | 2 | 20 | 30 | 50 |
| IV | NME II | 21UCSN41 | Introduction to Internet | 2 | 2 | 20 | 30 | 50 |
| | Ability Enhancement Course - IV | 21UAYM41 | Yoga & Meditation | 2 | 2 | 20 | 30 | 50 |
| | Self- Study Course / On-line Course / Internship (Optional) | 21UCSSS2 | Web Technology MOOC | | +2 | | 50 | 50 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| V | NCC, NSS & Sports | | | | 1 | | | |
| | Extension Activity | | | | +1 | | | |
| **Total** | | | | **30** | **25+3** | | | |

**Semester V**

| Part | Component | Course Code | Course Title | Contact Hours/ Week | Credits | Max. Marks | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CIA | ESE | Total |
| | Core V (Common Core) | 21UCMC51 | Computer Oriented Numerical Methods | 6 | 5 | 40 | 60 | 100 |
| III | Core VI | 21UCSC51 | Operating Systems | 4 | 4 | 40 | 60 | 100 |
| | Core VII | 21UCSC52 | Python Programming | 4 | 4 | 40 | 60 | 100 |
| | Core Elective- 1 | 21UCSE51/<br>21UCSE52 | Data Mining/Introduction to IoT | 4 | 4 | 40 | 60 | 100 |
| | Mini Project | 21UCSP51 | Mini Project | 5 | 2 | 40 | 60 | 100 |
| | Core Practical V | 21UCSCR5 | Python Programming Lab | 5 | 3 | 40 | 60 | 100 |
| IV | Common Skill Based | 21UCSB51 | Computer for Digital Era and Soft Skills | 2 | 2 | 20 | 30 | 50 |
| | Self-Study / Online course (optional) | 21UCSSS3 | Mathematical Reasoning | | +2 | | 50 | 50 |
| **Total** | | | | **30** | **24+2** | | | |

## Semester VI

| Part | Component | Course Code | Course Title | Contact Hours/ Week | Credits | Max. Marks | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | CIA | ESE | Total |
| III | Core VIII | 21UCSC61 | .NET Programming | 5 | 4 | 40 | 60 | 100 |
| | Core IX | 21UCSC62 | Software Engineering | 5 | 4 | 40 | 60 | 100 |
| | Core X | 21UCSC63 | Computer Networks | 5 | 4 | 40 | 60 | 100 |
| | Core Elective-2 | 21UCSE61/ 21UCSE62 | Cloud Computing / Mobile Computing | 4 | 4 | 40 | 60 | 100 |
| | Core practical VI | 21UCSCR6 | .NET Programming Lab | 5 | 3 | 40 | 60 | 100 |
| IV | Project | 21UCSP61 | Project | 6 | 4 | 40 | 60 | 100 |
| | | | **Total** | **30** | **23** | | | |
| **Grand Total** | | | | **180** | **140+5** | | | |

| Semester | Hours | Credits | Extra Credits |
|---|---|---|---|
| I | 30 | 21 | --- |
| II | 30 | 21 | --- |
| III | 30 | 26 | -- |
| IV | 30 | 25 | 3 |
| V | 30 | 24 | 2 |
| VI | 30 | 23 | -- |
| Total | **180** | **140** | **5** |

LESSON PLAN

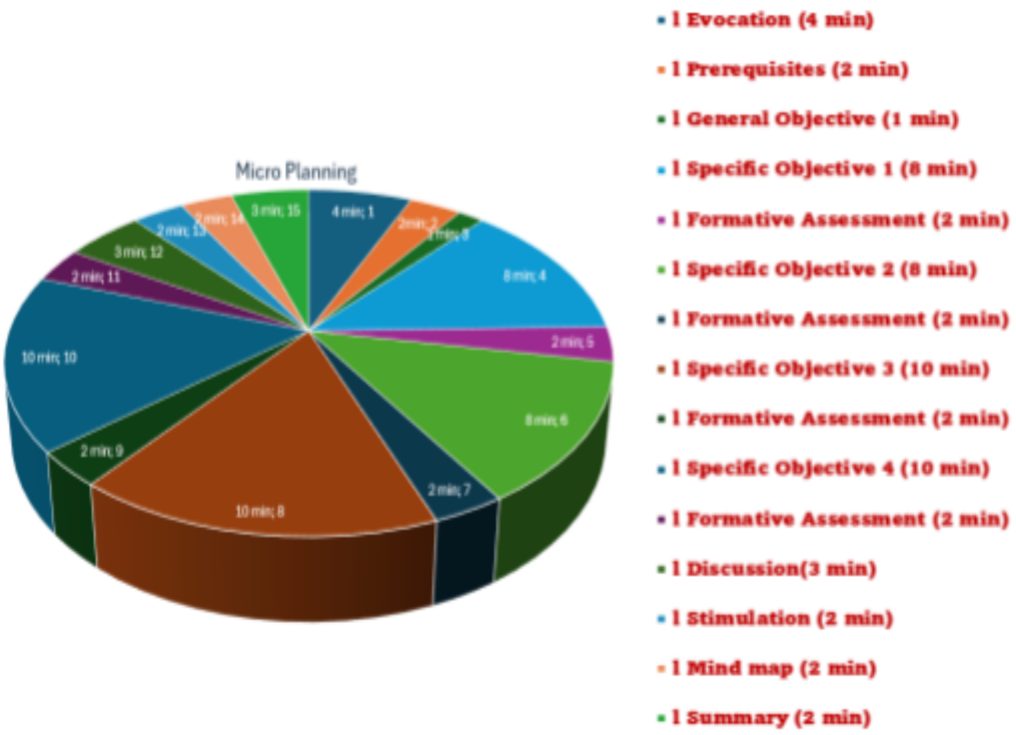
Objective Oriented Learning Process RBT

| | |
|---|---|
| Programme | B.Sc. Computer Science |
| Semester | I |
| Subject title | Major Core: C Programming |
| Code | 21UCSC11 |
| Hours | 4 |
| Total Hours | 60 |
| Credits | 5 |
| Max Marks | 100 |
| Unit & Title | Unit: III – Passing arrays to function |
| Name of the Faculty | M.Jothimani |
| T-L tools | Lecture method, Visual aid: PPT, online references, and real-time execution |

**Prerequisite Knowledge:**

Basic understanding of functions arrays in C programming.
Understanding of functions in C programming (declaration, definition, and calling).

**Micro -planning**



**1. Topic for Learning through Evocation:**

Imagine you are managing the semester marks obtained in the subject Mathematics, including the register number and the name of the student. You want to create a program that can store all the marks in an array and calculate statistics like the total, and average marks obtained in the class, the first three best marks in the subject Mathematics, or the mark list's standard deviation.

However, instead of doing this within the main program, you create a function that takes in the array of marks and performs the calculation. This scenario allows us to think about how arrays, which are collections of related data, can be passed to functions for efficient processing.

Relating the  concept to C programming:

- Arrays in C are passed by reference when used as function arguments, meaning the function works directly with the original array in memory.
- By passing an array to a function, we avoid duplicating data, which saves memory and time.
- It might need to pass the whole mark details to an expert for analysis, in C programming, arrays are passed to functions for manipulation or analysis.

Students should participate in the discussion of operations on a list of data in manual operation and will be asked to think about how they could structure a program to pass an array of integers (representing, for example, marks) to a function and perform various operations, like calculating the sum or average of the values in the array.

---

**2. Topic Introduction:**

1. In C programming, an array is a data structure that stores a fixed-size sequence of related elements of the same data type and the term 'array' is derived from the concept of a collection of elements stored in a sequential order in memory.
   . Arrays allow efficient manipulation of a large number of related values under a single identifier.
2. Function Syntax for Passing Arrays:

   Declaration: void functionName(int arr[], int size)

3. Passing Arrays to Functions:

   Unlike variables that are passed by value, arrays in C are passed by reference when used as function arguments. This eliminates the need to copy the entire array each time it is used, improving both memory efficiency and performance. When an array is passed to a function, the function operates on the original array rather than a copy.

4. According to the C programming language's specifications, arrays decay into pointers when passed to functions.
   This means that, within the function, the array is treated as a pointer to its first element. Thus, any modification made to the array in the function reflects back in the original array from which it was passed.

**2.1. General Objective:**

Enables the students to:

- Understand the concept of passing arrays to functions.

**2.2. Specific Objective:**

- Describe how arrays are passed to functions in C programming.
- Indicate to declare and define functions that accept arrays
- Show to access and modify array values within functions.
- Apply this knowledge to solve problems involving arrays in functions. and \
  troubleshoot pointer-related issues when passing arrays to functions.

**First Phase :**

SO1 - Explain formal and actual parameters

SO2 - Outline the format to declare and define functions that accept arrays

SO3 - Discuss how array elements are referenced inside the function from the main function

SO4 - Write a program to pass an integer array to a function to find the sum of the array Elements.

**Second Phase:**

SO1 - Show how the arrays are passed as parameters to a function as formal and actual parameters

SO2 - Demonstrate the format to declare and define functions that accept arrays

SO3 - Experiment with how array elements are referenced inside the function from the main function

SO4 - Solve a problem to find the sum of an integer array of elements using a function.

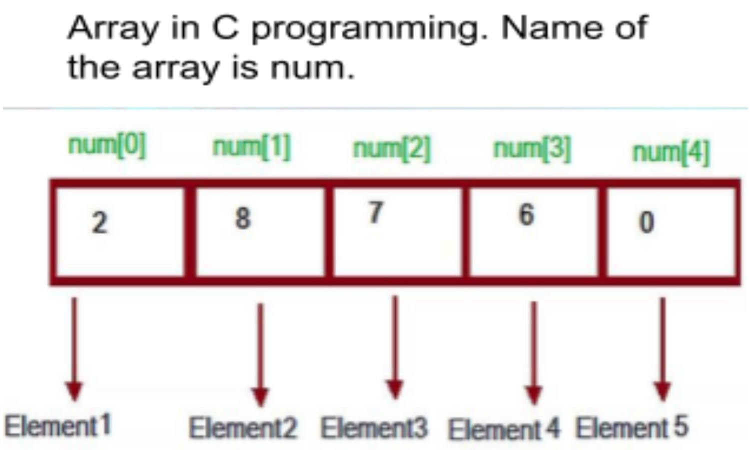
## 2.3:Taxonomy of objectives:

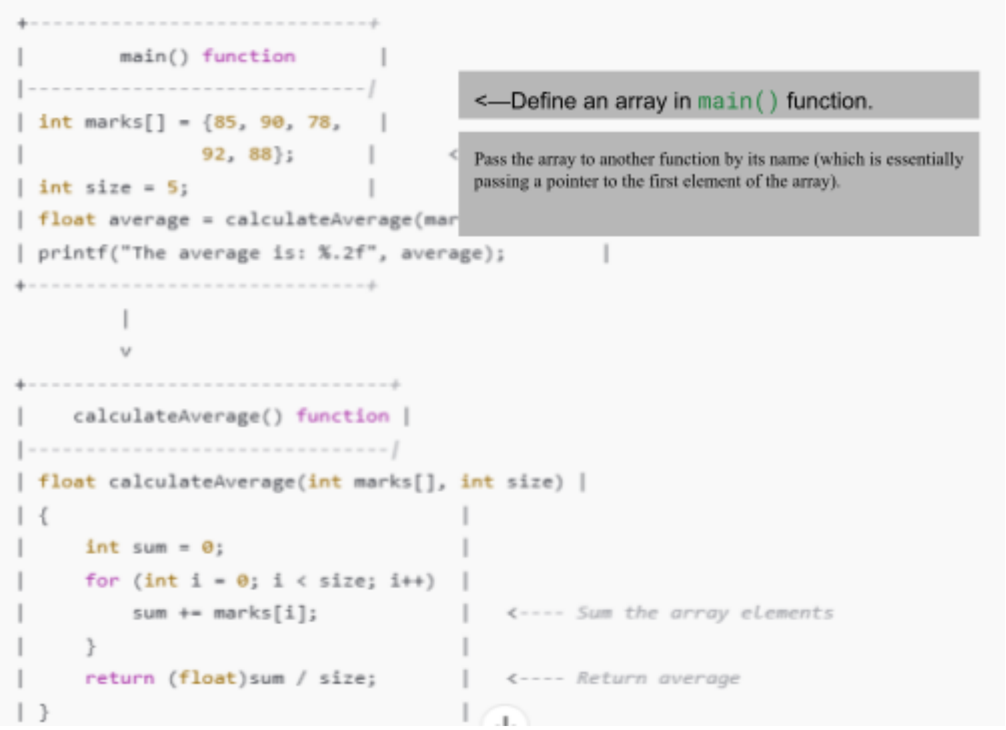
| Taxonomy of Objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | 1 | | | | | |
| B.Conceptual Knowledge | | 1,2 | | | | |
| C. Procedural Knowledge | | | 3,4 | 4 | | |
| D. Meta Cognitive Knowledge | | | 2,3 | 4 | | |

## 2.4. Key Words:

- Arrays, Function, Pass by Reference, Pointer, Size of Array

## 2.5. Key Diagrams (if any)





```
+------------------------------+
|        main() function       |
|------------------------------/
| int marks[] = {85, 90, 78,   |
|               92, 88};       |        <
| int size = 5;                |
| float average = calculateAverage(mar
| printf("The average is: %.2f", average);         |
+------------------------------+
        |
        v
+--------------------------------+
|   calculateAverage() function |
|--------------------------------/
| float calculateAverage(int marks[], int size) |
| {                                    |
|     int sum = 0;                     |
|     for (int i = 0; i < size; i++)   |
|         sum += marks[i];             |   <---- Sum the array elements
|     }                                |
|     return (float)sum / size;        |   <---- Return average
| }                                    |
```

<—Define an array in main() function.

Pass the array to another function by its name (which is essentially passing a pointer to the first element of the array).

**3. Discussion:**

In C, **arrays are always passed to function as pointers.** They cannot be passed by value because of the array decay due to which, whenever array is passed to a function, it decays into a pointer to its first element. However, there can be different syntax for passing the arrays as pointers.

The easiest way to pass an array to a function is by defining the parameter as the **undefined-sized array**. Let's take a look at an example:

```
include <stdio.h>
// Array passed as an array without the size of its
// dimension
void printArr(int arr[], int n) {
   for (int i = 0; i < n; i++)
      printf("%d ", arr[i]);
}
int main() {
   int arr[] = {1, 2, 3, 4, 5};
   // Pass array to function
   printArr(arr, 5);
```

```
    return 0;

}
```

Output

```
2 4 6 8 10
```

In this example, we pass an integer array **arr** to the function **printArr()**. In the function prototype, we define the array as an array of integers with no size information.

We may also notice that we have passed the array size as an argument as a second parameter to the function. It is recommended to pass the size of the array to the function as an additional parameter.

The other ways of passing arrays as pointers include:

### Passing as Sized Array

Similar to the method above, we can also pass the array size as an argument when passing the array to the function. However, it will still be treated as a pointer within the function.

```
#include <stdio.h>
// Array passed as an array with the size of its dimension
void printArr(int arr[5], int n) {
for (int i = 0; i < n; i++)
  printf("%d ", arr[i]);
}
int main() {
    int arr[] = {1, 2, 3, 4, 5};
    // Pass array to function
    printArr(arr, 5);
    return 0;
}
```

Output

```
10 20 30 40 50
```

Pass the array arr and its size size to the function printArray. The function then prints all elements of the array based on the given size.

Example:

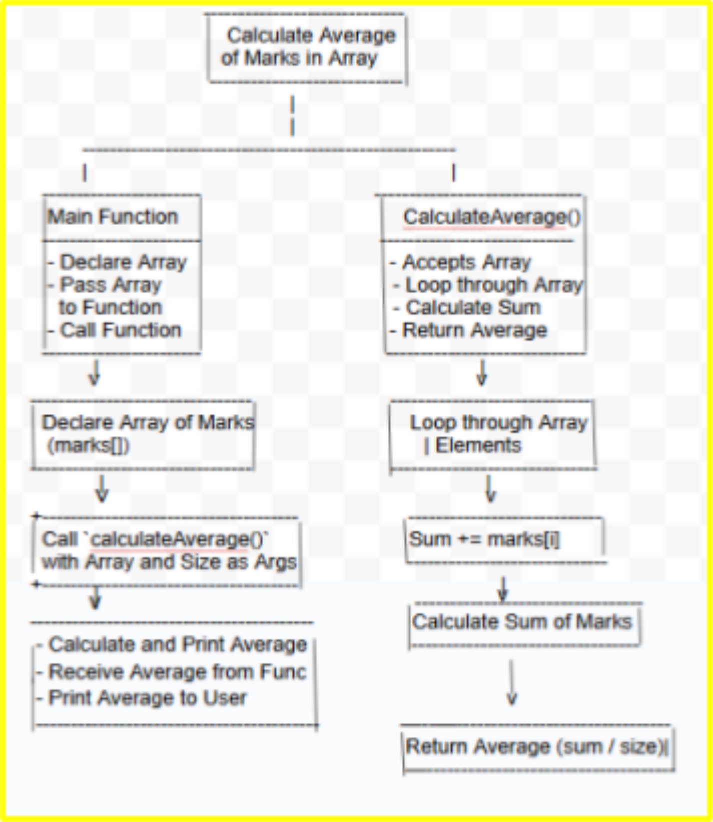
```
#include <stdio.h>
// Function to calculate sum of array elements
void sumArray(int arr[], int size) {
    int sum = 0;
    for(int i = 0; i < size; i++) {
        sum += arr[i];
    }
    printf("Sum of array elements: %d\n", sum);
}
int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int size = sizeof(arr) / sizeof(arr[0]);

    // Passing array to function
    sumArray(arr, size);
    return 0;
}
```

- How does C handle arrays when passed to a function?
- What happens if we try to modify an array inside a function?
- How can we avoid common mistakes when passing arrays?

One student will be asked to demonstrate a basic code example of passing an array to a function.

**PPT Link:**
https://docs.google.com/presentation/d/1iSYNynOYHP3yCTnGE6cGdd82Nn--yyActl5IEufqFvo/edit?usp=sharing

**4. Mind Map:**


Calculate Average of Marks in Array
Main Function
- Declare Array
- Pass Array to Function
- Call Function
CalculateAverage()
- Accepts Array
- Loop through Array
- Calculate Sum
- Return Average
Declare Array of Marks (marks[])
Loop through Array | Elements
Call `calculateAverage()` with Array and Size as Args
Sum += marks[i]
Calculate Sum of Marks
- Calculate and Print Average
- Receive Average from Func
- Print Average to User
Return Average (sum / size)

A mind map showing the relationship between functions, arrays, and pointers. The map will include:

- Arrays are passed by reference.
- Functions that modify arrays.
- The syntax for array passing.

**5. Summary:**

The students will be asked to summarize how arrays are passed to functions and the effects of passing by reference in C programming. This will be followed by solving a short coding problem on the board to reinforce the concept.

**6. Assessment through Stimulating Questions/Analogy/New Ideas and Concepts:**

Formative Assessment 1 (FA1) (2 minutes)

- Ask the students to write definitions for actual and formal parameters.

Formative Assessment 2 (FA2) (2 minutes)

- A short quiz on a function declaration and definition inside the function

Formative Assessment 3 (FA2) (2 minutes)

- Students execute the given code to change the array elements.

Formative Assessment 4 (FA4) (2 minutes)

- Ask students to discuss in groups how to access a book from a array books in a library array and write the code.
- Describe how arrays are passed to functions in C programming.
- Indicate to declare and define functions that accept arrays
- Show to access and modify array values within functions.
- Apply this knowledge to solve problems involving arrays in functions and \
  troubleshoot pointer-related issues when passing arrays to functions.

**7.FAQ's:**

| |
|---|
| 1. What happens if an array is passed to a function without specifying its size?<br>The size of the array will not be known within the function unless you explicitly pass the size as a separate argument. |
| 2. Can we modify an array in the function?<br>Yes, since arrays are passed by reference, changes made to the array within the function will reflect outside the function. |
| 3. What is the difference between passing an array and passing a pointer to a function? |

Both are essentially the same because arrays decay into pointers when passed to a function, but a pointer allows more flexibility in terms of memory manipulation.

4. Can we pass multidimensional arrays to functions?
   Yes, but you need to specify the sizes of all but the first dimension in the function declaration.

## 8. References

1. Kernighan, B. W., & Ritchie, D. M. (1988). *The C Programming Language (2nd Edition)*. Englewood Cliffs, NJ: Prentice Hall.
2. Gottfried, B. S.  *Programming with C (4th  Edition)*. New York: McGraw-Hill.
3. Deitel, P., & Deitel, H.*C How to Program (9th Edition)*. (2020). Hoboken, NJ: Pearson Education.

M.Jothimani
Associate Professor in computer Science

S.Gnanathangam
Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

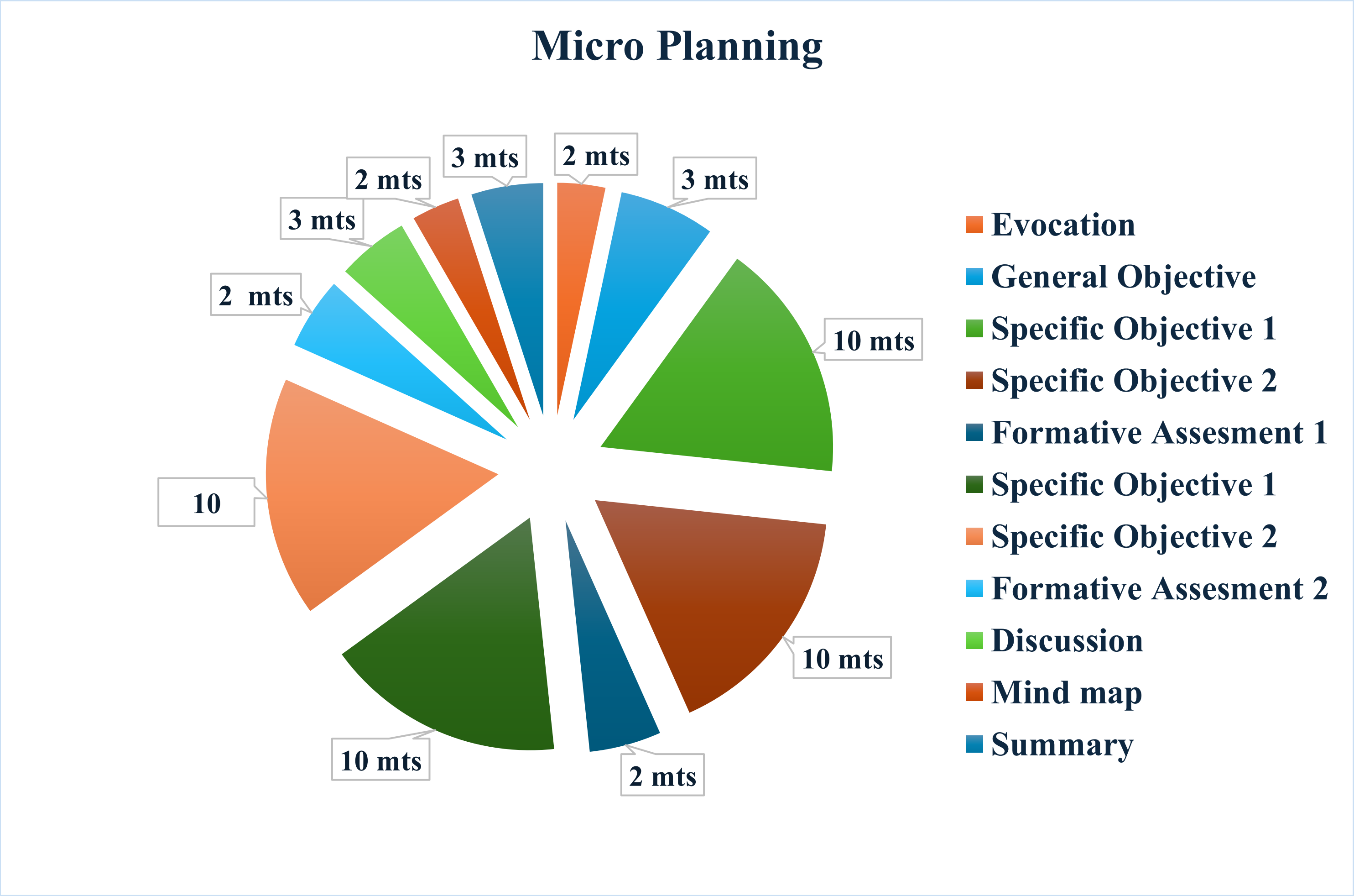
Verified by Subject Expert

Approved by HOD

# LESSON PLAN

## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | I |
| **Course Title** | Allied : Mathematics for Computer Science |
| **Course Code** | 21UCSA11 |
| **Hours** | 3 |
| **Total Hours** | 45 |
| **Credits** | 3 |
| **Max Marks** | 100 |
| **Unit & Title** | Unit: I Prepositional Logic |
| **Name of the Faculty** | Ms. J. Anila Maily |
| **T-L tools** | Lecture method, **Visual aid**: PPT |



1. **Learning through evocation**

Learners are introduced to Logic.

## 2. Topic Introduction:

Students are introduced to propositions and compound propositions.

### 2.1 General Objective:

Enables the students to understand what is a preposition and logical operations.

### 2.2 Specific Objectives:

Enables the students to:

1. To identify whether the given statement is a proposition or not?
2. To describe compound preposition.

First Phase:

SO1 : Understand what is preposition, types of preposition

SO2 : Represent a given statement as a preposition and determine whether a preposition is true or false.

Second Phase:

SO1 : Describe conjunction, disjunction and negation

SO2 : Generate the truth table using alternative method. Logic equivalence.

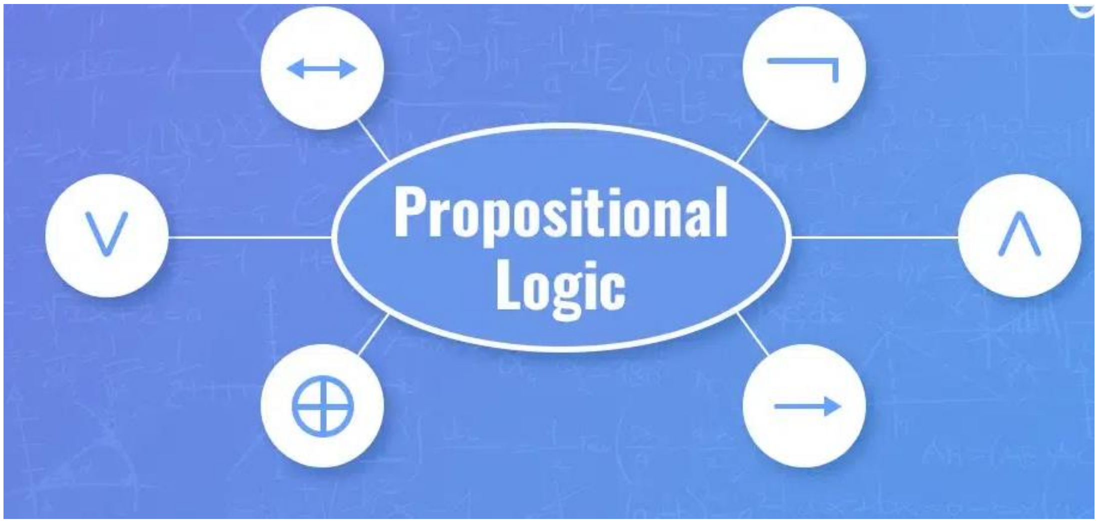
### 2.3: Taxonomy of objectives:

| **Taxonomy of Objectives** | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyse** | **Evaluate** | **Create** |
| A. Factual Knowledge | 1,2 | | | | | |
| B. Conceptual Knowledge | | **1,2** | **1,2** | | | |
| C. Procedural Knowledge | | | | 1,2 | | |
| D. Meta Cognitive Knowledge | | | | | **1,2** | 2 |

### 2.4: Key words:

Preposition, Conjunction, Disjunction, Negation

### 2.5: Key diagrams :

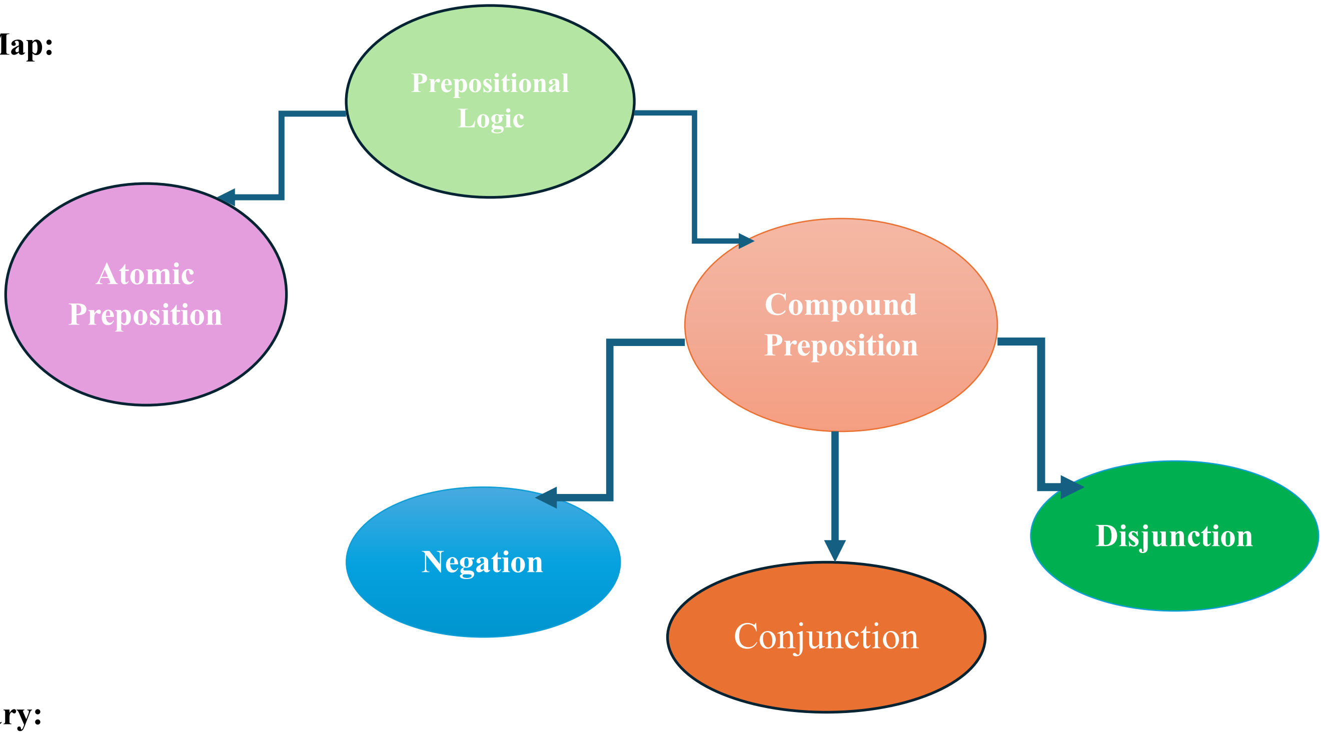


**PowerPoint Presentation:** Propositional Logic.pptx

3. **Discussion:**

   The students will be asked to reflect how prepositional logic can be used in real life.

**4. Mind Map:**



\

**5. Summary:**

Students will be asked to identify combinational circuits, half adder and full adders.

**6. Assessment**

**Formative assessment1 :** Describe what is a preposition and types of prepositions.

**Formative assessment2 :** Explain logic connectives.

Let p, q, and r be the propositions

p: You get an A on the final exam.

q: You do every exercise in this book.

r: You get an A in this class.

**Write these propositions using p, q, and r and logical connectives** (including negations)

a) You get an A in this class, but you do not do every exercise in this book.

b) You get an A on the final, you do every exercise in this book, and you get an A in this class.

c) To get an A in this class, it is necessary for you to get an A on the final.

**7. FAQ's:**

| |
|---|
| 1. Which one of the following is a preposition |
| a) 1+2=2 b) Give me a pen c) I love India d) What is the time now? |
| 2. The declarative sentences to which it is possible to assign one and only one of the two possible truth values are called _________. |
| a) Conjunctions b) statements c) disjunctions d) connectives |
| 3. The atomic statements are those which do not have any |
| a) Conjunctions b) statements c) disjunctions d) connectives |
| 4. If truth value of statement P is , then the truth value of $\neg p$ is _______. |
| a) True b) False c) True or False d) True and False |
| 5. If truth values of statement P is true and Q is false, then the truth value of $P \wedge Q$ is |
| True b) False c) True or False d) True and False |

**8. References**

1 Seymour Lipschutz, Marc Lipson, *Discrete Mathematics,* New Delhi : Tata McGraw Hill, Revised Third Edition, 2017

2. B.S. Vatsa, *Discrete Mathematics*, New Delhi: New Age International (P) Ltd.**,** Fourth Edition, 2009.

3 K.D. Joshi, *Foundation of Discrete Mathematics* ,New Delhi: New Age International (P) Ltd.,2014

4 3. Kenneth H. Rosen *, "Discrete mathematics and its application"*, New Delhi : Tata McGraw Hill,8th Edition, 2021

**Verified by Subject Expert**
J.Anila Maily
Associate Professor of Computer Science

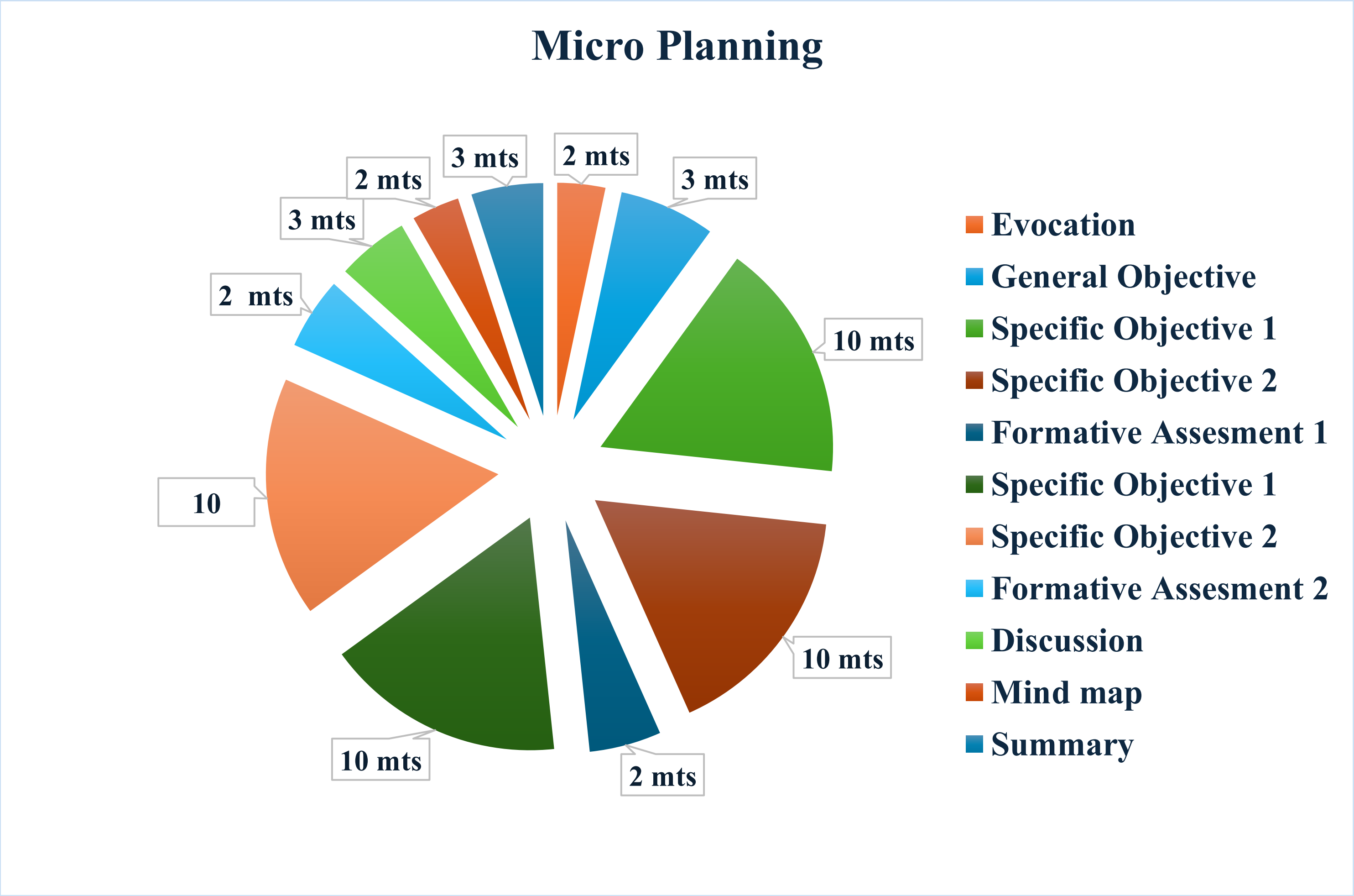
**Approved by HOD**
S.Gnanathangam

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

# LESSON PLAN

## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | I |
| **Course Title** | Skill Enhancement Course-I Professional English for Computer Science –I |
| **Course Code** | 21UCSPE1 |
| **Hours** | 2 |
| **Total Hours** | 30 |
| **Credits** | 2 |
| **Max Marks** | 50 |
| **Unit & Title** | Unit II: Process Description , Flow Chart |
| **Name of the Faculty** | Ms. J. Anila Maily |
| **T-L tools** | Video, **Visual aid**: PPT |



1. **Learning through evocation**

Students are asked to think of process involved in their day to day activities.

## 2. Topic Introduction:

Students are introduced to the concept of process description.

### 2.1 General Objective:

Enables the students to represent a process description as a flow chart.

### 2.2 Specific Objectives:

Enables the students to:

1. To understand flow chart and different types of flow chart.
2. To learn various symbols used in flow charts.

First Phase:

SO1 : Why Process description is essential.

SO2 : To learn the steps involved in creating process flow diagram.

Second Phase:

SO1 : To understand the process description through process flowchart.

SO2 : Convert process flowchart into process description.

### 2.3 : Taxonomy of objectives:

| **Taxonomy of Objectives** | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyse** | **Evaluate** | **Create** |
| A. Factual Knowledge | 1,2 | | | | | |
| B. Conceptual Knowledge | | **1,2** | **1,2** | | | |
| C. Procedural Knowledge | | | 1,2 | | | |
| D. Meta Cognitive Knowledge | | | | 1,2 | | 2 |

### 2.4 : Key words:

**Process, flowchart, Process flowchart**

### 2.5 : Key diagrams :

**PowerPoint Presentation:** https://smctuty-my.sharepoint.com/:p:/g/personal/anilamaily_stmaryscollege_edu_in/EUSbDIwukEtIkiuPrsPKi7sBH5roxpjPELv6oi-JpRRzNw?e=Wf8wfJ

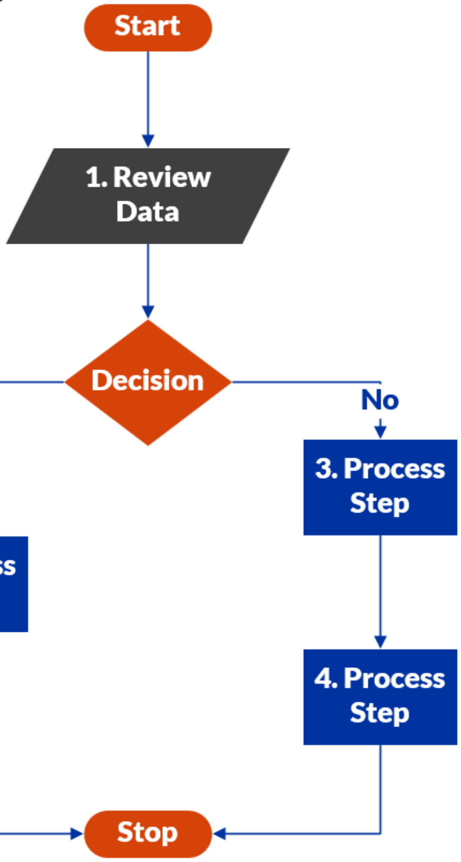
3. **Discussion:**

The students will be asked to create a process flowchart. Problem will be given to think of a process and create process flow diagram.

**4. Mind Map:**

**How to make a flowchart in a few simple steps:**

1. **Determine the purpose or function of the flowchart**
2. **Add steps and connect them with arrows**
3. **Add decisions**
4. **Show any loops back to previous steps**
5. **Share your flowchart and ask for feedback**



**5. Summary:**

Students will be asked to use create a process flow diagram.

## 6. Assessment

**Formative assessment1 :** Discuss about the various steps involved in process flow chart.

**Formative assessment2 :** Solve given problem using process flow chart.

## 7. References

1. https://www.smartdraw.com/flowchart/flowchart-maker.htm?srsltid=AfmBOor7LrqRZ9bbPd-HOgcWyC5_sam4eTF2pKmI_mh86zhxOz-zOlon
2. https://asana.com/resources/what-is-a-flowchart
3. https://miro.com/flowchart/
4. https://www.canva.com/graphs/process-flow/

**Verified by Subject Expert**
J. Anila Maily
Associate Professor of Computer Science

**Approved by HOD**
S. Gnanathangam
Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

## LESSON PLAN

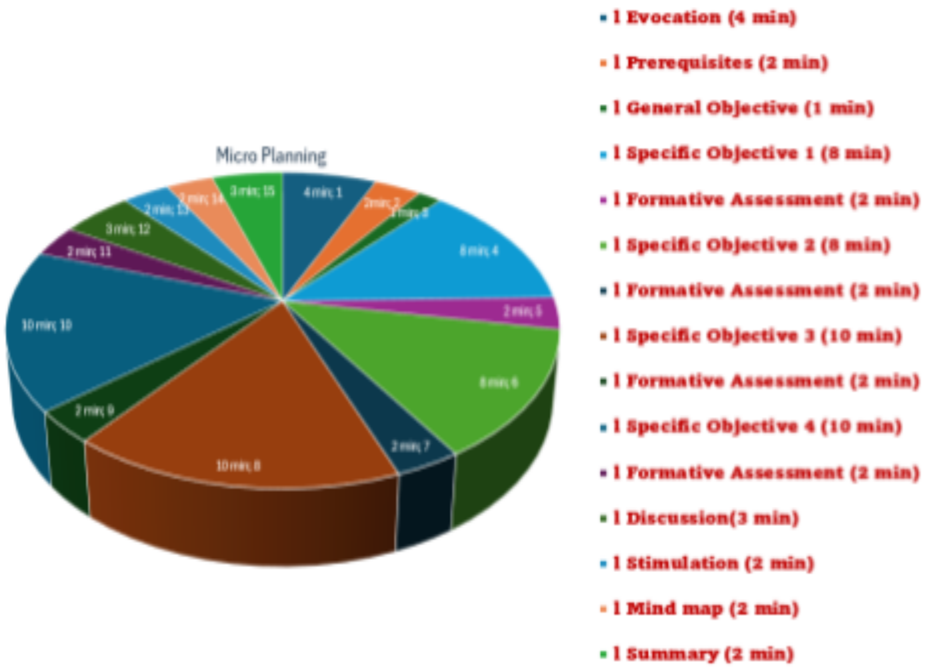
## Objective Oriented Learning Process RBT

| | |
|---|---|
| Programme | B.Sc. Computer Science |
| Semester | II |
| Subject title | Major Core:  C++ Programming |
| Code | 21UCSC21 |
| Hours | 4 |
| Total Hours | 60 |
| Credits | 5 |
| Max Marks | 100 |
| Unit & Title | Unit: II – Constructors in C++ |
| Name of the Faculty | M.Jothimani |
| T-L tools | Lecture method, Visual aid: PPT, online references, and real-time execution |

Prerequisite Knowledge

- Basic understanding of functions in C++ (declaration, definition, and calling).
- Knowledge of classes and objects in C++.
- Understanding arrays in C programming.

### Micro -planning



## 1. Topic for Learning through Evocation:

To design a program for a library system. You need to store details about books (like the title, author, and ISBN number). Instead of manually assigning these values for each book, you want the program to initialize them automatically when creating a book object. This introduces the concept of constructors in C++.

Relating the concept to programming:

In C++, a constructor is a special member function that is automatically invoked when an object is created. It is used to initialize the object's data members with default or specific values.

**Discussion :**

How can we avoid repetitive initialization of object attributes in a program?

How does automation in object creation relate to constructors?

### 2. Topic Introduction

Definition:

A constructor is a special function in a class that has the same name as the class. It is used to initialize objects when they are created.

Characteristics of Constructors in C++:

- Same name as the class.
- No return type, not even void.
- Automatically called when an object is created.
- Can be overloaded (multiple constructors with different parameter lists).

### 3.1. General Objective

- Enable students to understand the concept of constructors and their importance in automating object initialization in C++.

### 2.2. Specific Objectives

- Define constructors and their purpose in C++.
- Differentiate between default, parameterized, and copy constructors.
- Write code to demonstrate constructors in  C++.
- Apply constructors to solve real-world problems, such as initializing a list of objects

**First Phase :**
SO1  -Define constructors
SO2  -Explain the types of constructors
SO3  - Develop coding to construct constructors
SO4  - Analse the coding for different types of constructors

**Second Phase:**

SO1 - Show how the member functions and constructors are differing
SO2 - Demonstrate operations of different types of Constructors
SO3 - Develop coding with varying types of constructors
SO4 - Analyse the programs with constructors

## 2.3. Taxonomy of Objectives

| Taxonomy of Objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | 1 | | | | | |
| B.Conceptual Knowledge | | 1,2 | | | | |
| C. Procedural Knowledge | | | 3,4 | | | |
| D. Meta Cognitive Knowledge | | | | 2,3, 4 | | |

## 2.4. Keywords

Constructor

Default Constructor

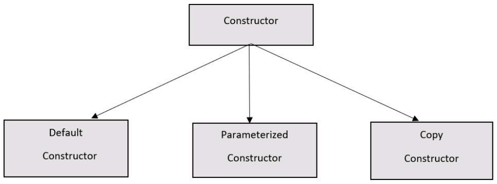
Parameterized Constructor

Copy Constructor

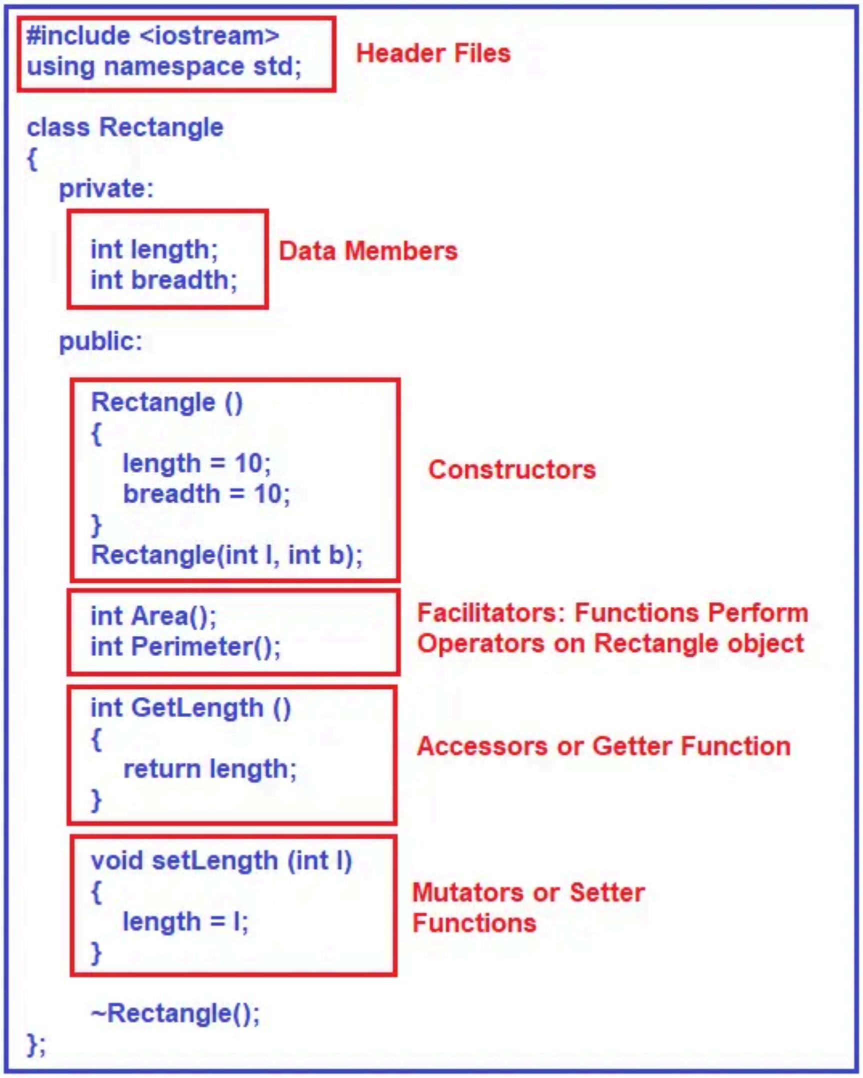
Object Initialization

## 2.5. Key Diagrams

Include diagrams to show:



The process flow of object creation and constructor invocation.



```
#include <iostream>
using namespace std;

class Rectangle
{
    private:

        int length;
        int breadth;

    public:

        Rectangle ()
        {
            length = 10;
            breadth = 10;
        }
        Rectangle(int l, int b);

        int Area();
        int Perimeter();

        int GetLength ()
        {
            return length;
        }

        void setLength (int l)
        {
            length = l;
        }

        ~Rectangle();
};
```

Header Files
Data Members
Constructors
Facilitators: Functions Perform Operators on Rectangle object
Accessors or Getter Function
Mutators or Setter Functions

**3. Discussion:**

A constructor is a special type of member function in a class that is automatically invoked when an object of the class is created. The primary role of a constructor is to initialize the object's data members to appropriate values, ensuring that the object starts its life in a valid and well-defined state.

In simpler terms, constructors are used to set up an object when it is instantiated, allowing the programmer to define the initial state of the object.

Characteristics of Constructors:

- Same Name as the Class:
  The constructor has the same name as the class it belongs to.
  For example, if you have a class named `Car`, the constructor will also be named `Car`.
- No Return Type:
  Constructors do not have any return type, not even `void`. They simply initialize the object.

- Automatically Called:
  Constructors are automatically called when an object is created. You do not need to call them explicitly.
- Overloading:
  C++ allows constructor overloading, which means a class can have multiple constructors with different parameter lists. This helps create objects in different ways.

Types of Constructors in C++

Default Constructor:
Parameterized Constructor:
Copy Constructor:

Explicit Constructor:

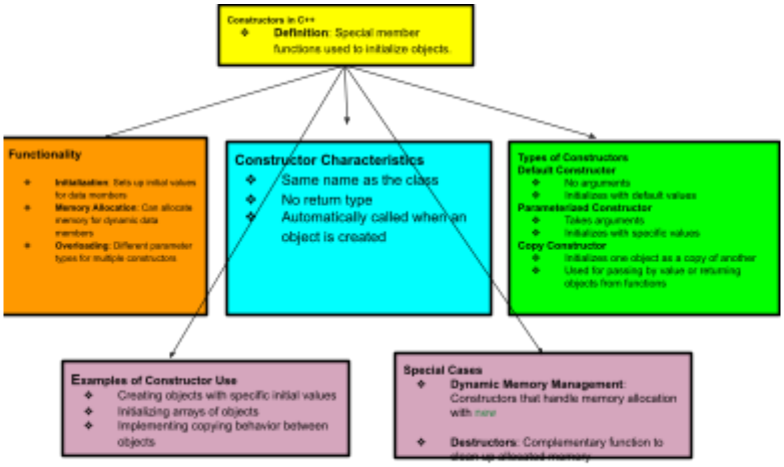
- The `explicit` keyword is used to prevent the compiler from automatically converting types during implicit type conversions.
- This is useful when you want to prevent a constructor from being used for implicit conversions or copying in certain contexts.

Role of Constructors in Object Initialization:

Real-world Examples:

**PPT LINK:**
https://docs.google.com/presentation/d/1a1r_x3mMFNUiqKQbytj0WYJK98JN_S1wSOLq4VMGWso/edit?usp=sharing

**4. Mind Map:**



Constructors in C++
- **Definition**: Special member functions used to initialize objects.

Functionality
- **Initialization**: Sets up initial values for data members
- **Memory Allocation**: Can allocate memory for dynamic data members
- **Overloading**: Different parameter types for multiple constructors

Constructor Characteristics
- Same name as the class
- No return type
- Automatically called when an object is created

Types of Constructors
Default Constructor
- No arguments
- Initializes with default values
Parameterized Constructor
- Takes arguments
- Initializes with specific values
Copy Constructor
- Initializes one object as a copy of another
- Used for passing by value or returning objects from functions

Examples of Constructor Use
- Creating objects with specific initial values
- Initializing arrays of objects
- Implementing copying behavior between objects

Special Cases
- **Dynamic Memory Management**: Constructors that handle memory allocation with new
- **Destructors**: Complementary function to clean up allocated memory

**5. Summary**

- A constructor is a special member function automatically called when an object is created.
- Constructors can be the default, parameterized, or copy constructors.
- They automate object initialization and enhance code efficiency.
- Explicit Constructors:

## 6. Assessment through Stimulating Questions/Analogy/New Ideas and Concepts:

Formative Assessment 1 (FA1) (2 minutes)

- With quiz questions to review the basic concepts of constructors in C++.

Formative Assessment 2 (FA2) (2 minutes)

- Ask to discuss on types of constructors

Formative Assessment 3 (FA3) (2 minutes)

- Oral questions to explain the operation of the code step by step.

Formative Assessment 4 (FA4) (2 minutes)

- Implement and execute a C++ program using constructors.

## 7.FAQ's:

How is a constructor different from a normal function?

1. A constructor has the same name as the class.
2. It does not return any value, not even void.
3. It is automatically invoked when an object is instantiated.

2. Can a constructor be private

Yes, constructors can be private, which is useful in singleton design patterns to restrict object creation from outside the class.

3. What happens if a constructor is not defined in a class?

If no constructor is explicitly defined, C++ provides a default constructor that does nothing (except for built-in types, which remain uninitialized).

4.Why use an initializer list in a constructor?

An initializer list is a more efficient way to initialize class members, especially for const or reference members.

```
class Example {
          const int x;
      public:
          Example(int a) : x(a) {}  // Initializer list
      };
```

## 8. References

- Kanetkar, Y. (2017). *Let Us C++* (2nd Edition). New Delhi, India: BPB Publications.
- Dey, D. (2018). *Programming in C++* (2nd Edition). New Delhi, India: Oxford University Press.
- Kamthane, A. N. (2013). *Object-Oriented Programming with ANSI & Turbo C++*. New Delhi, India: Pearson Education India.
- Stroustrup, B. (2013). *The C++ Programming Language* (4th Edition). Boston, MA: Addison-Wesl

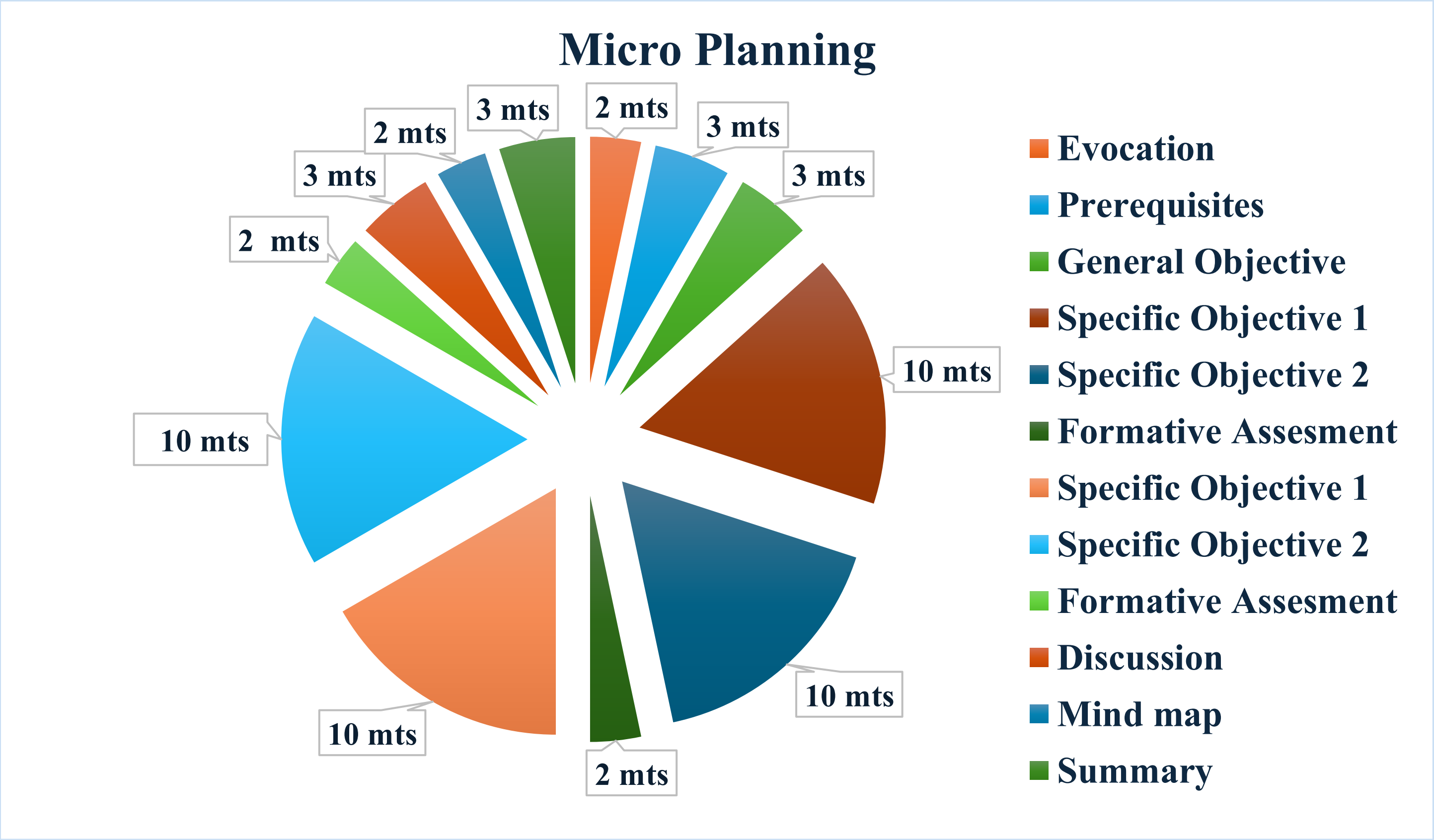
M.Jothimani
Associate Professor in computer Science

S. Gnanathangam
S.Gnanathangam
Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

# LESSON PLAN

## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | II |
| **Course Title** | Allied : Digital Electronics |
| **Course Code** | 21UCSA21 |
| **Hours** | 3 |
| **Total Hours** | 45 |
| **Credits** | 3 |
| **Max Marks** | 100 |
| **Unit & Title** | Unit: III Combinational Logic |
| **Name of the Faculty** | Ms. J. Anila Maily |
| **T-L tools** | Lecture method, **Visual aid**: PPT |



1. **Learning through evocation**

Various number systems, binary, octal, decimal and hexadecimal. Students are asked to do the conversion.

**Prerequisite:**

Binary operations, Binary addition

## 2. Topic Introduction:

Students are introduced to the different digital circuits, sequential circuits and combinational circuits

### 2.1 General Objective:

Enables the students to understand what is combinational circuit and its types.

### 2.2 Specific Objectives:

Enables the students to:

1. To design half adder
2. describe and design a full adder

First Phase:

SO1 : Understand what is a half adder, Perform the addition operation, Draw the truth table

SO2 : Understand what is a full adder and draw the truth table

Second Phase:

SO1 : Simplify the half adder equation with K-map and design the circuit

SO2 : Use K-map to simplify the full adder equation and design the circuit. Design a full adder using two half adders

### 2.3: Taxonomy of objectives:

| Taxonomy of Objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyse** | **Evaluate** | **Create** |
| A. Factual Knowledge | 1,2 | | | | | |
| B. Conceptual Knowledge | | **1,2** | **1,2** | | | |
| C. Procedural Knowledge | | | | 1,2 | | |
| D. Meta Cognitive Knowledge | | | | | **1,2** | 2 |

### 2.4: Key words:

Combinational Circuit, Adder, Half adder, Full adder

### 2.5: Key diagrams :

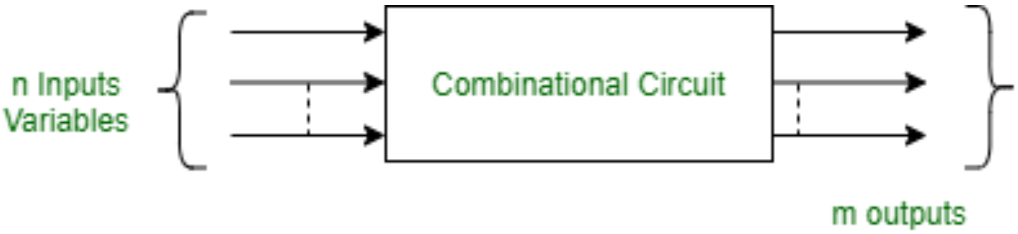


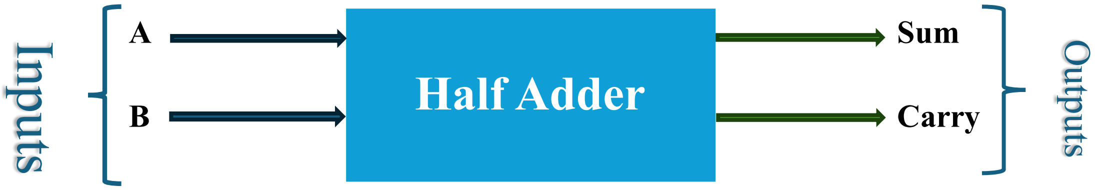
Figure - Block diagram of Combinational circuit

**Points to Remember on Combinational Logic Circuit:**

- Output depends upon the combination of inputs.
- Output is a pure function of present inputs only i.e., Previous State inputs won't have any effect on the output. Also, It doesn't use memory.

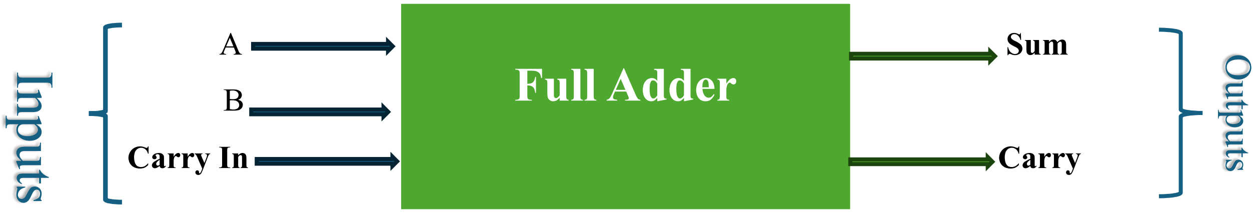
**Design of Half Adder:**

A combinational logic circuit that performs the addition of two single bits is called Half Adder.



**Design of Full Adder:**

A combinational logic circuit that performs the addition of three single bits is called Full Adder.

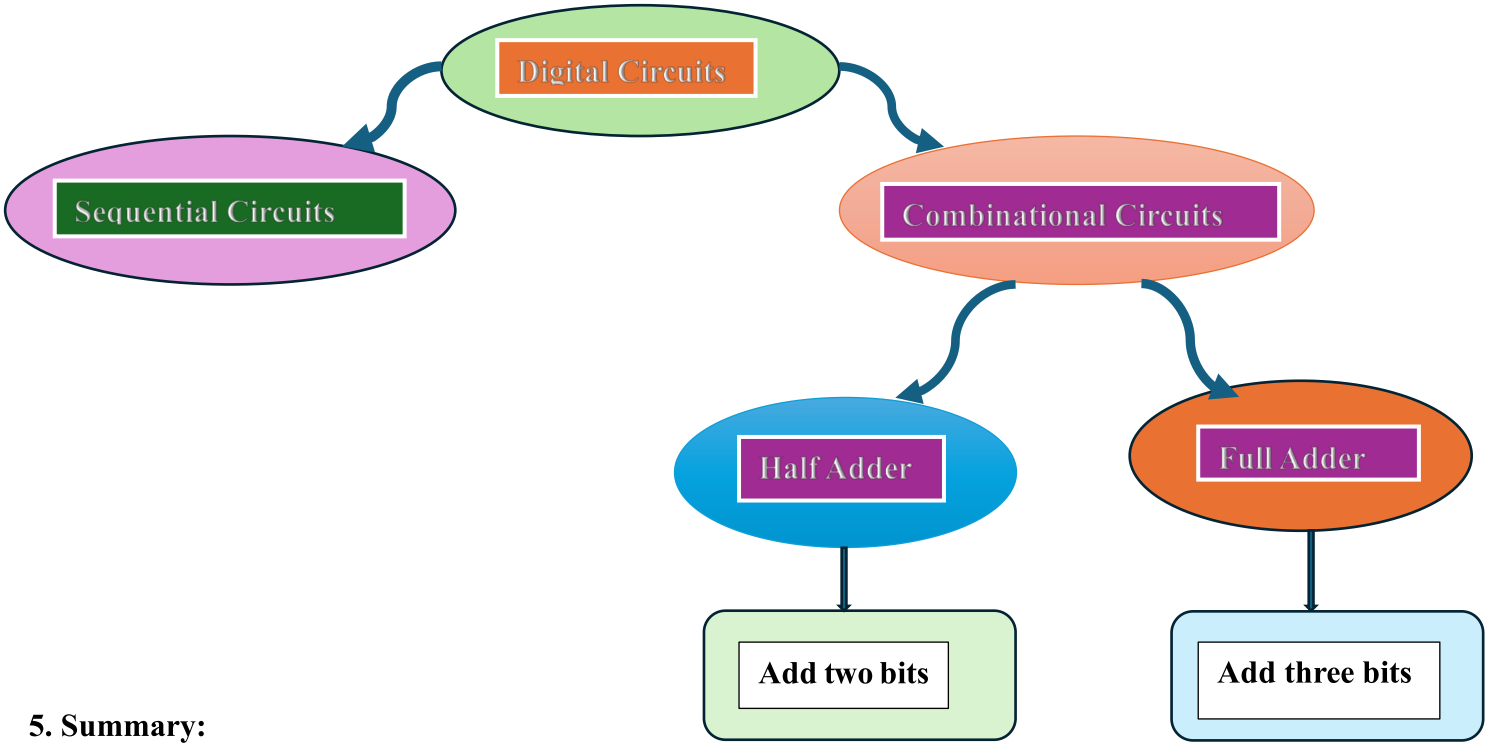


**PowerPoint Presentation:** Adders.pptx

3. **Discussion:**

The students will be asked questions how combinational circuits can be used for other binary operations.

**4. Mind Map:**



**5. Summary:**

Students will be asked about characteristics of combinational circuits and how they are used to perform the binary addition operations.

**6. Assessment**

**Formative assessment1 :** Get the simplified equation using k-map for half adder.

**Formative assessment2 :** Get the simplified equation using k-map for full adder.

**7. FAQ's:**

1. Total number of inputs in a half adder is ___________

   a) 1 b) 2 c) 3 d) Any integer number

2. Total number of outputs in a half adder is __________

   a) 1 b) 2 c) 3 d) Any integer number

3. Total number of inputs in a full adder is ___________

   a) 1 b) 2 c) 3 d) Any integer number

4. Total number of outputs in a full adder is __________

   a) 1 b) 2 c) 3 d) Any integer number

5. If A and B are the inputs of a half adder, the sum is given by ___________

   a) A AND B b) A OR B c) A XOR B d) A EX-NOR B

**8. References**

1. M. Morris Mano, *Digital Logic and Computer Design*, Noida: Pearson education India, First Edition,2016
2. Charles H.Roth, Jr. *Fundamentals of Logic Design*, New Delhi: Cengage Learning India Private Limited, 7th Edition, 2015
3. DonaldD.Givone, *Digital Principles and Design*, New Delhi: Tata McGraw-Hill, First Edition,2012.
4. Donald P.Leach and Albert Paul Malvino, *Digital Principles and Applications*, New Delhi: Tata McGraw Hill, 8th Edition, 2014.

**Verified by Subject Expert**
J. Anila Maily
Associate Professor of Computer Science

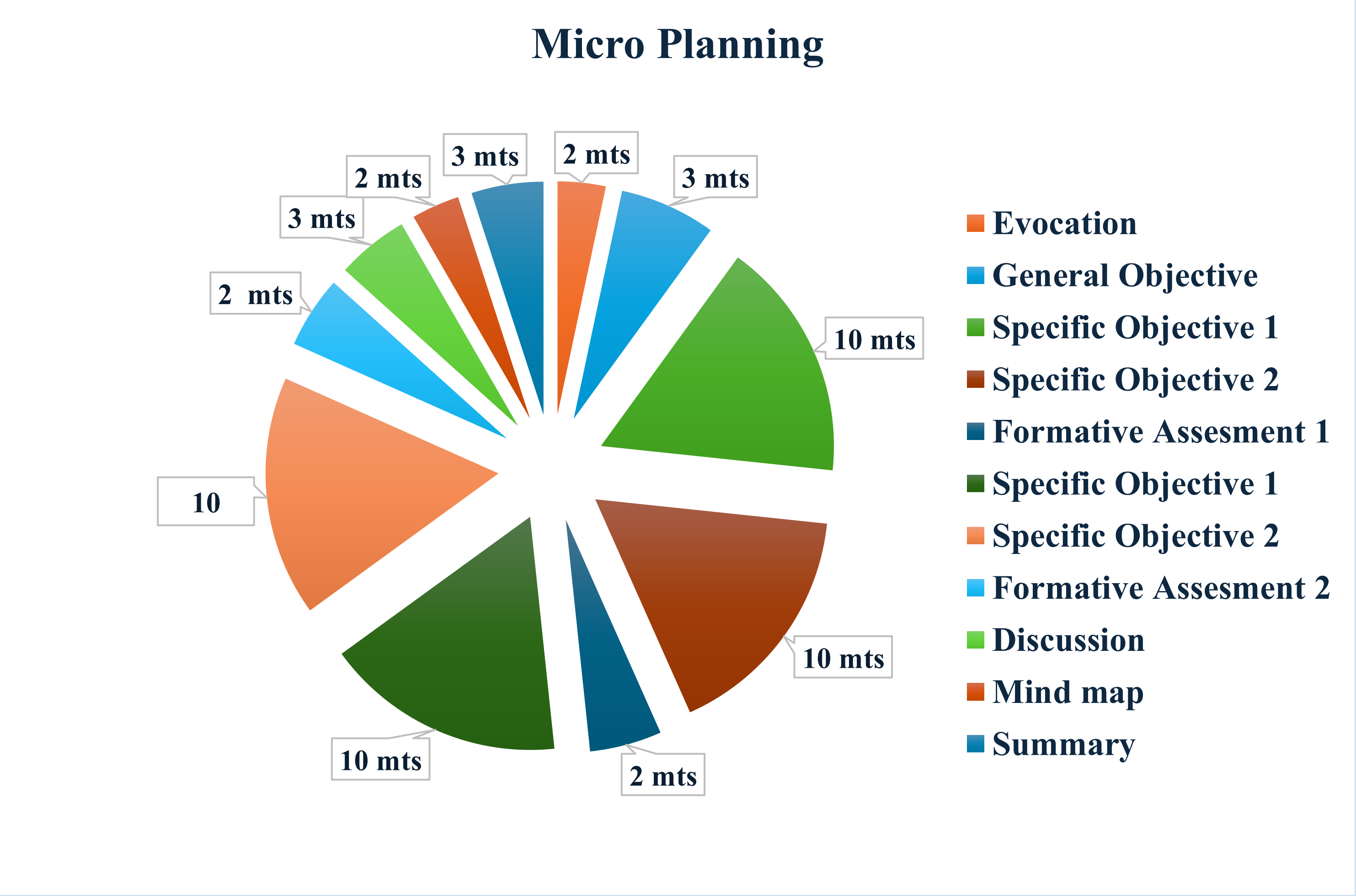
**Approved by HOD**
S. Gnanathangam

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

# LESSON PLAN

## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | II |
| **Course Title** | Skill Enhancement Course-I Professional English for Computer Science –II |
| **Course Code** | 21UCSPE2 |
| **Hours** | 2 |
| **Total Hours** | 30 |
| **Credits** | 2 |
| **Max Marks** | 100 |
| **Unit & Title** | Unit II : Creating a Flyer |
| **Name of the Faculty** | Ms. J. Anila Maily |
| **T-L tools** | **Visual aid**: Videos,PPT |



1. **Learning through evocation**

Students are asked to think of various advertising modes.

2. **Topic Introduction:**

Students are introduced to brochures and flyers.

**2.1 General Objective:**

Enables the students to create flyers

**2.2 Specific Objectives:**

Enables the students to:

1. To create content for the flyers
2. To know about various software used to create flyers

First Phase:

SO1 : How to use google docs to create a flyer

. SO2 : To create flyers using MS-word and PowerPoint

Second Phase:

SO1 : Create a flyer using canva or visme.

SO2 : Create a attractive flyer for the imaginary brand owned by them.

**2.3: Taxonomy of objectives:**

| **Taxonomy of Objectives** | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyse** | **Evaluate** | **Create** |
| A. Factual Knowledge | 1,2 | | | | | |
| B. Conceptual Knowledge | | **1,2** | **1,2** | | | |
| C. Procedural Knowledge | | | | 1,2 | | |
| D. Meta Cognitive Knowledge | | | | | 1,2 | 1,2 |

**2.4: Key words:**

Flyer, Brochure, Canva, Visme

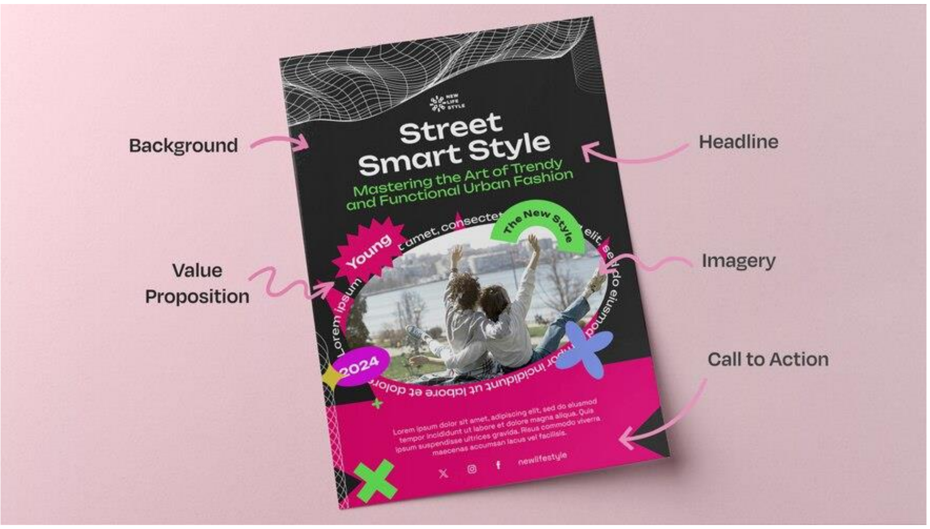
**2.5 : Key diagrams :**

**PowerPoint Presentation:** PE2.pptx

3. **Discussion:**

The students will be asked to discuss which software enable us to create attractive flyers.

**4. Mind Map:**


Background
Value Proposition
Headline
Imagery
Call to Action
NEW LIFE STYLE
Street Smart Style
Mastering the Art of Trendy and Functional Urban Fashion
The New Style
Young
2024
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore
newlifestyle

**5. Summary:**

Students will be asked to use create attractive flyers for their imaginary brands.

**6. Assessment**

**Formative assessment1 :** Discuss about the various steps involved in creating a flyer.

**Formative assessment2 :** Create a flyer.

**7. References**

Link1 :**https://slidemodel.com/how-to-create-a-flyer-in-powerpoint/**
**Link 3: https://slidesgo.com/slidesgo-school/powerpoint-tutorials/how-to-create-a-flyer-in-powerpoint**
**Link 4: https://www.canva.com/learn/the-ultimate-guide-to-flyer-design/**
**Link 5 : https://www.visme.co/flyer-maker/**

S. Gnanathangam

**Verified by Subject Expert**
J. Anila Maily
Associate Professor of Computer Science

**Approved by HOD**
S. Gnanathangam

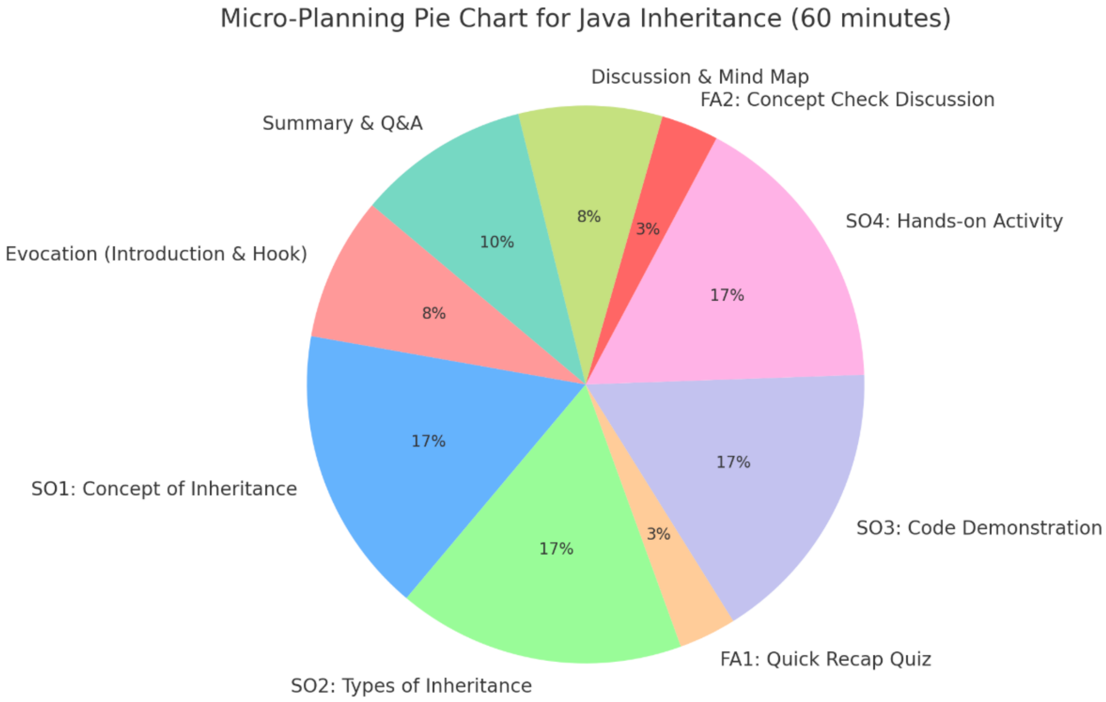
Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

| | |
|---|---|
| Programme | B.Sc |
| Semester | III |
| Course Title | Java Programming |
| Code | 21UCSC31 |
| Hours | 4 |
| Total Hours | 60 |
| Credits | 4 |
| Max Marks | 100 |
| Unit & Title | I / The History and Evolution of Java |
| Name of the Faculty | Sr. K. Reeta |
| T-L tools | PowerPoint slides, Code Examples, Cheat Sheet |

**Pre-requisite Knowledge:**

Basic understanding of Java syntax, classes, and objects.

**Micro planning: 60 minutes**



| Activity | Time (min) |
|---|---|
| Evocation (Introduction & Hook) | 5 min |
| SO1: Concept of Inheritance | 10 min |
| SO2: Types of Inheritance | 10 min |
| FA1: Quick Recap Quiz | 2 min |
| SO3: Code Demonstration | 10 min |
| SO4: Hands-on Activity | 10 min |
| FA2: Concept Check Discussion | 2 min |
| Discussion & Mind Map5 | 5 min |
| Summary & Q&A | 6 min |

**1. Evocation (5 minutes)**

- Ask students:
  - Have you noticed that different car models from the same brand share similar features?
  - How do you think we can achieve this in programming?
- Relate this to **Inheritance** in Java, where a class acquires properties and behaviors of another class.

**2. Introduction to Inheritance (10 minutes)**

General Objective:

- To understand the concept of **inheritance** and its advantages in object-oriented programming.

Specific Outcomes (SO1):

- Define **inheritance** in Java.
- Explain the role of **superclass (parent class)** and **subclass (child class)**.
- Discuss real-world analogies of inheritance.

*Example:*

```
class Animal {
   void eat() { System.out.println("This animal eats food."); }
}

class Dog extends Animal {
   void bark() { System.out.println("The dog barks."); }
}

public class TestInheritance {
   public static void main(String[] args) {
      Dog myDog = new Dog();
      myDog.eat();  // Inherited method
      myDog.bark(); // Own method
```

```
  }
}
```

**3. Types of Inheritance in Java (10 minutes)**

Specific Outcomes (SO2):

- Explain **single, multilevel, and hierarchical inheritance** in Java.
- Use **diagrams** and examples to demonstrate the different types.

| Type of Inheritance | Explanation |
|---|---|
| **Single Inheritance** | A subclass inherits from one superclass. |
| **Multilevel Inheritance** | A subclass inherits from another subclass. |
| **Hierarchical Inheritance** | Multiple subclasses inherit from a single superclass. |

**4. Formative Assessment 1 (FA1) – Quick Recap Quiz (2 minutes)**

- **Question:** What is the main purpose of inheritance?
- **Options:**
  A) Reduce code redundancy
  B) Increase complexity
  C) Improve memory usage
  D) None of the above
- **Answer: A) Reduce code redundancy**

**5. Implementing Inheritance in Java (10 minutes)**

Specific Outcomes (SO3):

- Demonstrate **method overriding** in Java.
- Explain **super() keyword** and its use.
- **Example:**

```
class Parent {
  void show() { System.out.println("Parent class method"); }
}
class Child extends Parent {
  void show() { System.out.println("Overridden method in Child class"); }
}
```

**6. Hands-on Activity – Pair Programming (10 minutes)**

Specific Outcomes (SO4):

- **Task:** Implement a Java program using single inheritance.
- **Instructions:**
  - Create a superclass **Vehicle** with a method **drive()**.
  - Create a subclass **Car** that extends **Vehicle** and adds a method **honk()**.
  - Write a main method to test both methods.

**7. Formative Assessment 2 (FA2) – Concept Check (2 minutes)**

- **Ask students:**
  - What is the difference between **method overriding** and **method overloading**?
  - When should we use **super()** in Java?

**8. Discussion & Mind Map (5 minutes)**

- **Activity:** Create a **mind map** on the board.
- **Key Points to Cover:**
  - **Concept of Inheritance**
  - **Types of Inheritance**
  - **Advantages & Use Cases**

**9. Summary (6 minutes)**

Summary:

- **Inheritance** allows code reuse and reduces redundancy.
- Java supports **single, multilevel, and hierarchical inheritance**.
- **Method overriding** helps modify inherited methods.
- **Hands-on practice is key** to mastering inheritance concepts.

**Assessment Questions**

MCQs:

1. What keyword is used to inherit from a class in Java?
   - a) extends
   - b) inherits
   - c) implements
   - d) none of the above
   - **Answer:** a) extends
2. Which inheritance type is not supported in Java?
   - a) Single
   - b) Multiple
   - c) Multilevel
   - d) Hierarchical

- **Answer:** b) Multiple

**Descriptive Questions:**

1. Explain **method overriding** in inheritance with an example.
2. Differentiate between **single and multilevel inheritance**.

**References**

1. **Java: The Complete Reference** by Herbert Schildt
2. **Effective Java** by Joshua Bloch

**Mind Map:**

```
├── Types of Inheritance (Java)
│   ├── Single
│   │   └── One derived class inherits from one base class
│   │       └── Example: class Animal { ... }; class Dog extends Animal
{ ... };
│   ├── Multilevel
│   │   └── A derived class inherits from another derived class
│   │       └── Example: class Animal { ... }; class Mammal extends
Animal { ... }; class Dog extends Mammal { ... };
│   ├── Hierarchical
│   │   └── Multiple derived classes inherit from the same base class
│   │       └── Example: class Animal { ... }; class Dog extends Animal
{ ... }; class Cat extends Animal { ... };
│   └── (Java does NOT support Multiple Inheritance directly - use
interfaces)
├── Keywords
│   ├── extends
│   │   └── Used to inherit: class ChildClass extends ParentClass{ ...}
│   └── super
│       ├── Refers to the immediate parent class
│       ├── Calls parent constructors: super()
│       ├── Calls parent methods: super.methodName()
│       └── Accesses parent variables: super.variableName()
├── Key Aspects
│   ├── Access Modifiers
│   │   ├── public: Accessible to all classes
│   │   ├── protected: Accessible within package and subclasses
```

```
|   |   ├── default: Accessible within the same package
|   |   └── private: Not accessible by subclasses
|   ├── Constructor Inheritance
|   |   └── Constructors are NOT inherited
|   |       └── super() calls parent's constructor
|   └── Method Overriding
|       ├── Subclass provides different implementation for inherited
method
|       └── @Override annotation (recommended)
|   └── Object Class
|       └── All classes implicitly inherit from ObjectProblem

└── Example Scenario
    ├── class Vehicle { int numWheels; void start(); }
    ├── class Car extends Vehicle { boolean hasSunroof; void drive(); }
    └── class Bicycle extends Vehicle { boolean hasBasket; void
pedal(); }
```

**Taxonomy of objectives:**

| Taxonomy of objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | | 1 | | | | |
| B. Conceptual Knowledge | | 2 | | | | |
| C. Procedural Knowledge | | | | | | 2 |

**PPT Link:**
[https://docs.google.com/presentation/d/10npf4UhNF1T8v7uFI6YZSv5reWnzv51_/edit?usp=sharing&ouid=113456501758187936448&rtpof=true&sd=true](https://docs.google.com/presentation/d/10npf4UhNF1T8v7uFI6YZSv5reWnzv51_/edit?usp=sharing&ouid=113456501758187936448&rtpof=true&sd=true)

**Verified by Subject Expert**

Reetha

S. Gnanathiny

**Course In-charge**

**Approved by HoD**

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

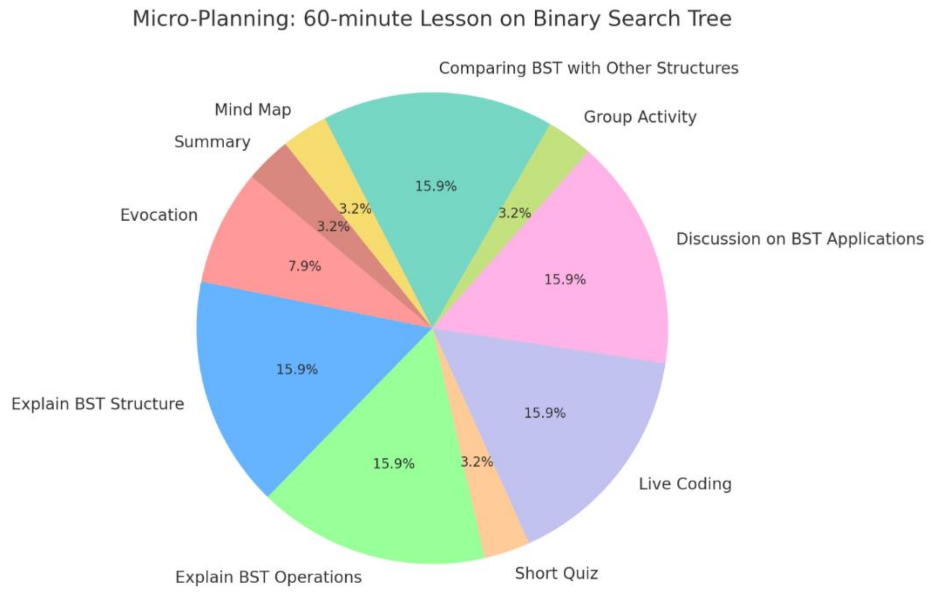
## Lesson Plan

| | |
|---|---|
| Programme | B. Sc. Computer Science |
| Semester | III |
| Course Title | Data Structures |
| Code | 21UCSA31 |
| Hours | 3 |
| Total Hours | 45 |
| Credits | 3 |
| Max Marks | 100 |
| Unit & Title | Unit IV: Binary Search Tree |
| Name of the Faculty | S.Gnanathangam |
| T-L tools | Mind Maps, Video, PowerPoint Presentation |

**Pre-requisite Knowledge:**

- Understanding of basic trees and linked lists
- Concept of recursion and sorting

**Micro-Planning: (60 minutes)**



| Phase | Activity | Duration |
|---|---|---|
| Evocation | Introduction to Trees and Binary Search Tree (BST) | 5 min |
| SO1 | Explain the Structure and Properties of BST | 10 min |
| SO2 | Explain BST Operations: Insertion, Deletion, Search | 10 min |

| | | |
|---|---|---|
| FA1 | Short Quiz on BST Basics | 2 min |
| SO1 | Live Coding of BST Insertion and Traversal | 10 min |
| SO2 | Discussion on BST Applications | 10 min |
| FA2 | Group Activity: Constructing a BST | 2 min |
| Discussion | Comparing BST with Other Data Structures | 10 min |
| Mind Map | Summarizing Key Concepts | 2 min |
| Summary | Recap and Q&A | 2 min |

1**. Evocation (5 min)**

- Ask students how they would efficiently store and search a sorted list of numbers.
- Introduce the concept of trees and explain how BST helps in fast searching and sorting.
- Show a simple tree diagram to illustrate hierarchy.

**2. Topic Introduction:**

**2.1 General Objective:**

- To understand the structure of BST
- To learn the operations of BST.

**2.2** Specific Outcomes (SO1 & SO2):

- Explain the structure, properties, and traversal methods of BST.
- Demonstrate BST operations like insertion, deletion, and searching with examples.

**First Phase**: Explanation (20 min)

**SO1 (10 min):** Explain the data structure, Traversal Methods and properties of BST. A

hierarchical data structure where each node has at most two children.

Left child < Parent < Right child.

**SO2 (10 min)**: Demonstrate BST Operations with diagrams and examples

- Insertion: How to add a node while maintaining BST properties.
- Search: Finding an element using recursive or iterative method.
- Deletion: Removing a node (3 cases: leaf node, one child, two children).

**Second Phase:** Implementations and Application (20 min)

**SO1 (10 min):** Live Coding Demonstration (C/C++/Python, Implement BST insertion and traversal (Inorder, Preorder, Postorder) and show the output of the BST structure for different inputs.

**SO2 (10 min):** Discuss the Applications of BST Used in database indexing, Mathematical expression and file systems.

**Mind Map (2 min)**

Draw a mind map that illustrate BST properties, operations and traversal techniques

**Summary (2 min)**

Summarize the Structure, properties, important operations like insertion, deletion, and search and BST traversal techniques and real-world applications.

**2.3. Taxonomy of Objectives:**

| Taxonomy of objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | 1 | 1 | | | | |
| B. Conceptual Knowledge | | 2 | | | | |
| C. Procedural Knowledge | | | | | 1 | |

**2.4. Key Words:**

Binary Search Tree (BST)

Node, Root, Leaf

Insertion, Deletion, Search

Inorder, Preorder, Postorder Traversal

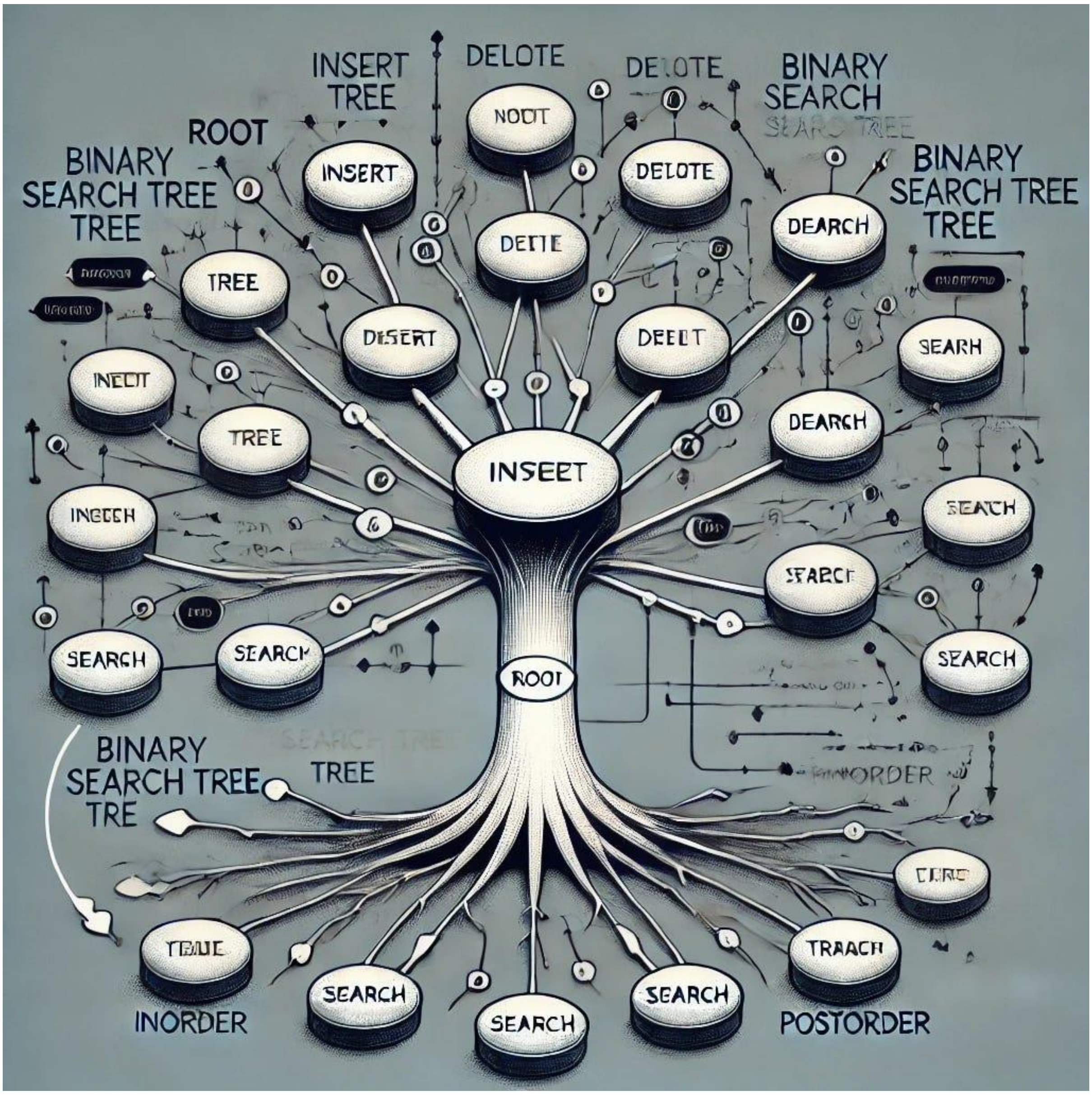
Balanced BST

**2.5 Powerpoint Presentation :** [https://drive.google.com/file/d/1F5F5WnygKDogkfqK7hdBOxCUlSXhQQt8/view?usp=drive_link](https://drive.google.com/file/d/1F5F5WnygKDogkfqK7hdBOxCUlSXhQQt8/view?usp=drive_link)

**3. Discussion:**

- Compare BST with other data structures (Arrays, Linked Lists, Hash Tables).
- Discuss the limitations of BST (Unbalanced BST and performance issues).
- Introduction to Balanced BSTs (AVL, Red-Black Trees)

4. **Mind map** for **Binary Search Tree (BST):**



Binary Search Tree (BST)

|

--------------------------------

| | | |

Definition  Operations Applications Pros & Cons

| | | |

Nodes with  Insert    Searching   ✅ Fast search (O log n)

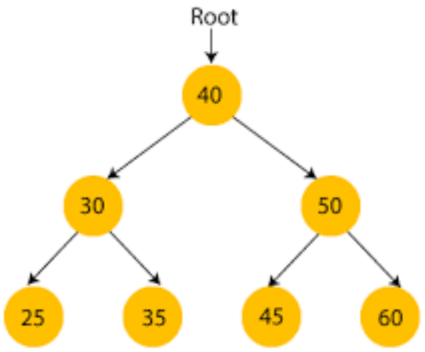
left < root Delete    Sorting    ❌ Can become unbalanced

right > root Search   Indexing    ❌ Worst case O(n)

Traversal

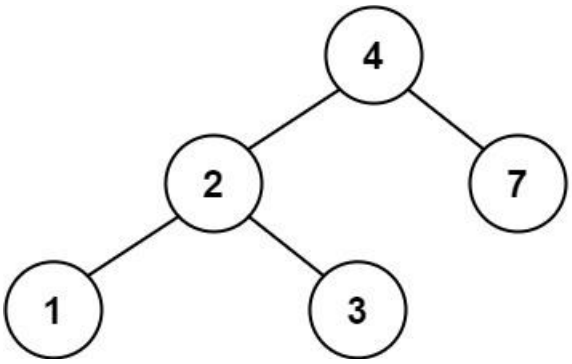
(Inorder, Preorder, Postorder)

**5. Summary:** Walk through the key operations of a Binary Search Tree: Insertion, Search, and Deletion.

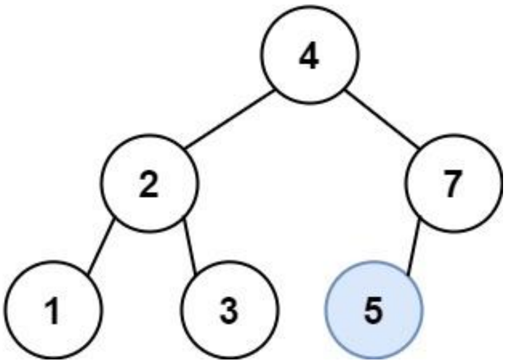


- **Insertion:**
  - Start from the root and compare the new value with the current node.
  - If the value is less than the current node, go left; if greater, go right.
  - When a null pointer is reached, insert the new value there.

**Example**



Before Insertion



After Insertion

- **Search:**
  - Start at the root. If the value is found, return it.
  - If the value is smaller than the current node, search the left subtree; if greater, search the right subtree.

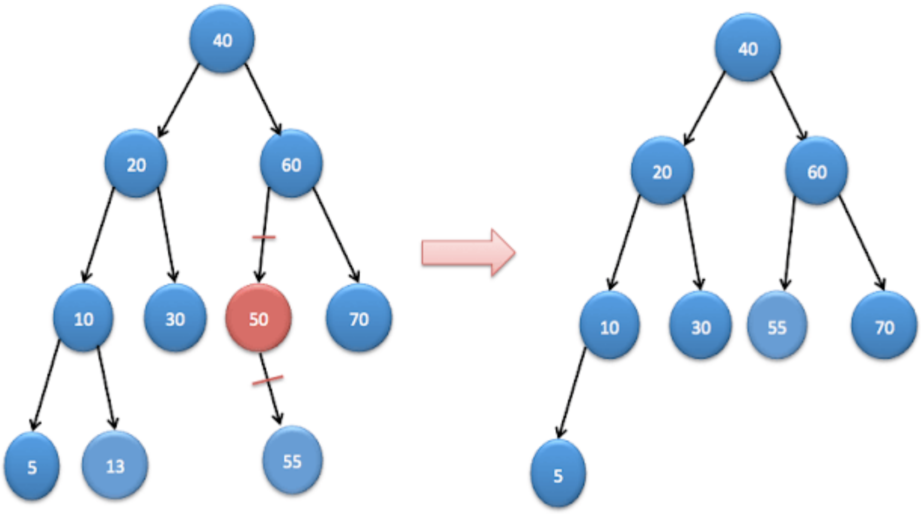
**Example Code:**

```
def search(root, key):
    if root is None or root.value == key:
        return root
    if key < root.value:
        return search(root.left, key)
    return search(root.right, key)
```

- **Deletion (optional, depending on class level):**

**Before** **After**

Deleting node 50 which have one child



1. Node to be deleted has no children (leaf node).
2. Node to be deleted has one child.
3. Node to be deleted has two children.

**Example for Deletion:**

Find the node, then consider the three cases for deletion. If the node has two children, replace it with its inorder successor (smallest node in the right subtree).

**6. Assessment through questions/analogy/new ideas:**

Formative Assessment 1 (FA1) (2 min)

Short quiz with 3-5 multiple-choice questions on BST concepts.

1. What is a Binary Search Tree (BST)?

2.Explain the insertion and deletion process in BST.

3.Write a program to implement BST and perform Inorder traversal.

4.How does BST improve search efficiency compared to arrays?

Formative Assessment 2 (FA2) (2 min)

Group Activity: "Given a set of numbers, construct a BST and perform Inorder traversal."

**7. FAQ’s: MCQ’s/ Descriptive questions:**

1.Examine the relationship between a BST and an AVL Tree.

2.Anayse the efficiency of BST

**8. References:**

1. Richard F.Giolberg&amp;Behrouz,A. forouzan,Data Structures - A Pseudo code Approach with C++, 4 th edition Thomson Brooks /Cole,2001.

2. Ellis Horowitz &amp;SartajSahani, Fundamentals of Data Structures, GalGotia

publications,2006.

3.Mark Allen Weiss, Data Structures and Algorithm Analysis in C++.

E resources:

- https://www.geeksforgeeks.org/binary-search-tree-data-structure/
- https://www.programiz.com/dsa/binary-search-tree

**9. Verified by Subject Expert:**

S. Gnanathayan

**Course In-charge**

S. Gnanathayan

**Approved by HoD.**

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

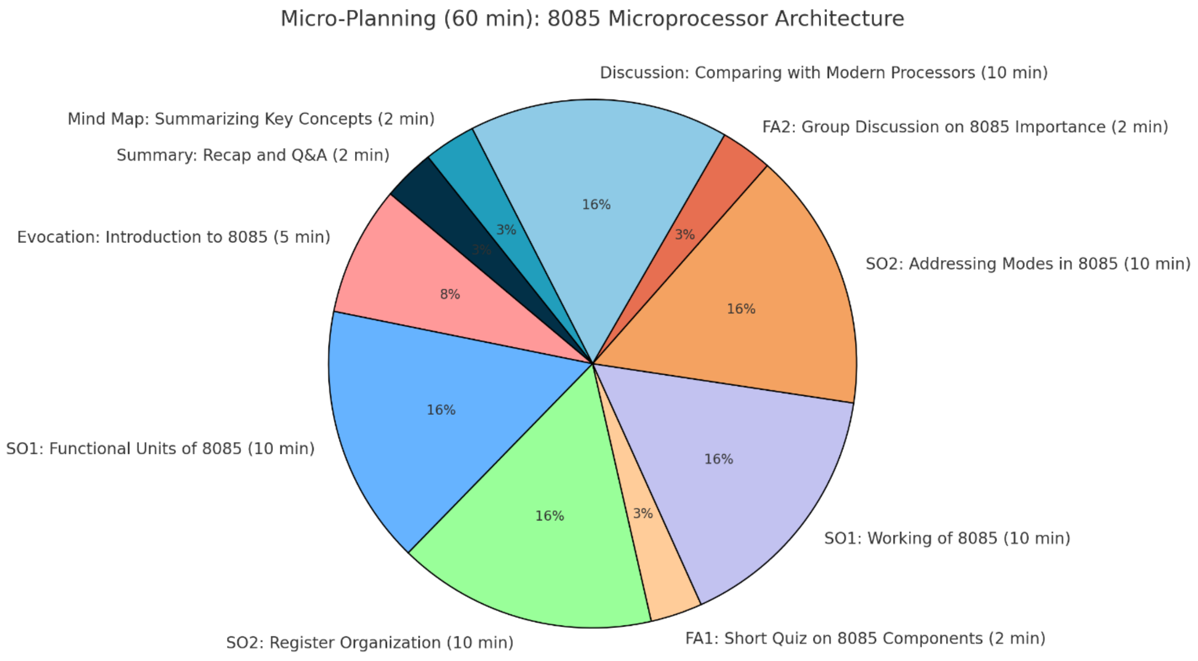
## Lesson Plan

| | |
|---|---|
| Programme | B. Sc. Computer Science |
| Semester | III |
| Course Title | Microprocessors |
| Code | 21UCSS31 |
| Hours | 2 |
| Total Hours | 30 |
| Credits | 2 |
| Max Marks | 50 |
| Unit & Title | Unit I: Architecture of 8085 |
| Name of the Faculty | S.Gnanathangam |
| T-L tools | Mind Maps, Video, PowerPoint Presentation, Microprocessor Simulator |

**Pre-requisite Knowledge:**

- basic understanding of computer architecture and digital electronics
- mnemonics of assembly language

**Micro-Planning: (60 minutes)**



| Phase | Activity | Duration |
|---|---|---|
| Evocation | Introduction to 8085 Microprocessor Architecture | 5 min |
| SO1 | Explain the Functional Units of 8085 Microprocessor | 10 min |

|  |  |  |
|---|---|---|
| SO2 | Explain the Register Organization of 8085 | 10 min |
| FA1 | Short Quiz on 8085 Components | 2 min |
| SO1 | Discuss the Working of the 8085 Microprocessor | 10 min |
| SO2 | Addressing Modes in 8085 | 10 min |
| FA2 | Group Discussion: Importance of 8085 in Early Computing | 2 min |
| Discussion | Comparing 8085 with Modern Processors | 10 min |
| Mind Map | Summarizing Key Concepts | 2 min |
| Summary | Recap and Q&A | 2 min |

1. **Evocation (5 min)**

- Ask students if they have heard about microprocessors in computers or embedded systems.
- Introduce the 8085 Microprocessor as the foundation of early computing.
- Show an image of the 8085 Microprocessor.

**2. Topic Introduction:**

**2.1 General Objective:**

- To understand the architecture of 8085 microprocessor.
- To uunderstand the working principles of 8085 microprocessor.

**2.2** Specific Outcomes (SO1 & SO2):

- Explain the functional units and working of 8085 Microprocessor.
- Explain the register organization and addressing modes of 8085.

**First Phase:** Explanation (20 min)

SO1 (10 min): Explain Arithmetic and Logic Unit (ALU), Timing and Control Unit, Instruction Register and Decoder, Interrupt Control

SO2 (10 min) : Describe the Register Organization such as Accumulator , General-Purpose Registers, stack Pointer and Special Purpose Registers

**Second Phase:** Application (20 min)

SO1 (10 min): Discuss the Working of the 8085 Microprocessor:

- How instructions are fetched, decoded, and executed.
- The role of buses (Address Bus, Data Bus, Control Bus).

SO2 (10 min): Discuss the addressing modes of 8085

- Immediate Addressing
- Direct Addressing
- Register Addressing
- Indirect Addressing
- Implicit Addressing

**Mind Map** (2 min)

Draw a mind map that summarize the architecture, working, and key components of the 8085 Microprocessor

Summary (2 min)

Summarize the functional units, importance and applications of 8085.

### 2.3. Taxonomy of Objectives:

| Taxonomy of objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | 1 | 1 | | | | |
| B. Conceptual Knowledge | | 2 | | | | |
| C. Procedural Knowledge | | | | 1 | | |

### 2.4. Key Words:

Microprocessor

8085 Architecture

Register Organization

Addressing Modes

Instruction Cycle

### 2.5 Powerpoint Presentation

https://docs.google.com/presentation/d/1_MKWflL-ZOMeujp8VLj2VmZNPqR9UGfJ/edit?usp=sharing&ouid=115208675943936491706&rtpof=true&sd=true

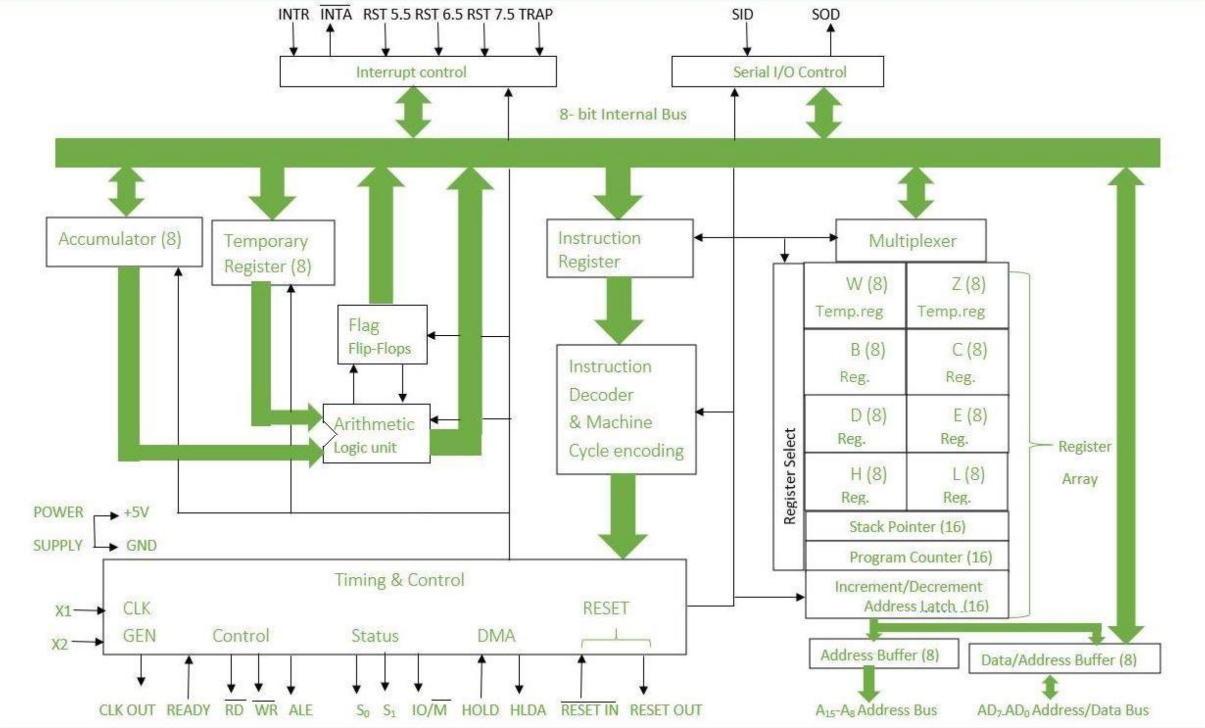
### 3. Discussion:

- Compare addressing modes.
- Discuss the limitations of 8085.

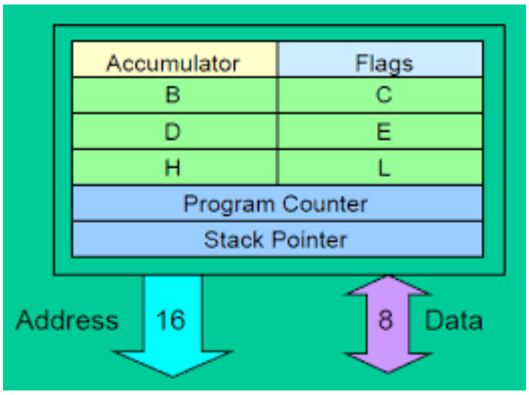
- Compare 8085 with Modern Processors (e.g., Intel, ARM).
- Discuss how 8085 is still used in academic and embedded systems projects.

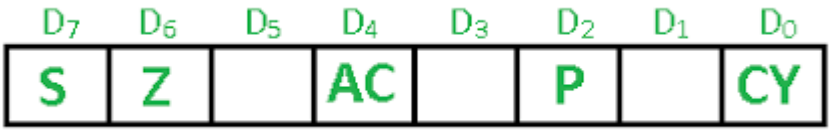
## 4. Mind Map:

Draw a mind map summarizing the architecture, working, and key components of the 8085 Microprocessor.



## 5. Summary: 8085 Architecture Overview





| $D_7$ | $D_6$ | $D_5$ | $D_4$ | $D_3$ | $D_2$ | $D_1$ | $D_0$ |
|---|---|---|---|---|---|---|---|
| S | Z |  | AC |  | P |  | CY |

Show the internal block diagram of the 8085 Microprocessor. Break down the diagram and explain the key components. Arithmetic Logic Unit (ALU) Accumulator (A):General-purpose registers , Program Counter (PC), Stack Pointer (SP), Instruction Register (IR), Control unit, Clock Generator

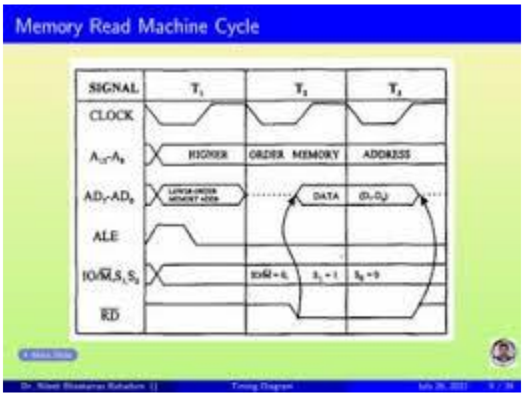
Data, Address, and Control Buses

Data Bus: 8-bit bi-directional bus used to transfer data between the microprocessor and memory or I/O devices.

Address Bus: 16-bit unidirectional bus used to specify the address in memory or I/O devices where data is to be read from or written to.

Control Bus: Used to carry control signals like READ, WRITE, I/O, and memory operations that govern the interaction with external devices.

Instruction Cycle and Machine Cycle



Memory Read Machine Cycle

Instruction Cycle: The cycle that the 8085 goes through to fetch, decode, and execute an instruction.

Machine Cycle: A machine cycle consists of a sequence of clock pulses for a specific operation (like reading from or writing to memory). Discuss the different types of machine cycles (Memory Read, Memory Write, I/O Read, I/O Write).

Practical/Hands-on Session

If available, use an 8085 simulator or a simple 8085 emulator to demonstrate the fetching and execution of instructions.

Students can experiment with writing simple programs (like moving data between registers) and simulate how the microprocessor behaves.

**6. Assessment through questions/analogy/new ideas:**

Formative Assessment 1 (FA1) (2 min)

Short quiz with 3-5 multiple-choice questions on 8085 Microprocessor.

1. What is ALE?

2.Explain the flag register.

3.Write a program to move a block of data from one location to another.

4.Compare addressing modes with example.

Formative Assessment 2 (FA2) (2 min)

Group Activity**:** "Why was the 8085 important in the development of modern processors?"

**7. FAQ's: MCQ's/ Descriptive questions:**

1.Compare 8085 with Modern Processors (e.g., Intel, ARM).

2.Discuss how 8085 is still used in academic and embedded systems projects.

3.Describe instruction and machine cycle with timing diagram.

**8. References:**

1. Ramesh Gaonkar, Microprocessor Architecture, Programming, and Applications with 8085.

2.Douglas V. Hall, Microprocessors and Interfacing.

3.IEEE Research Papers on Microprocessor Evolution.

**9.Verified by Subject Expert:**

S. Gnanathayan S. Gnanathayan

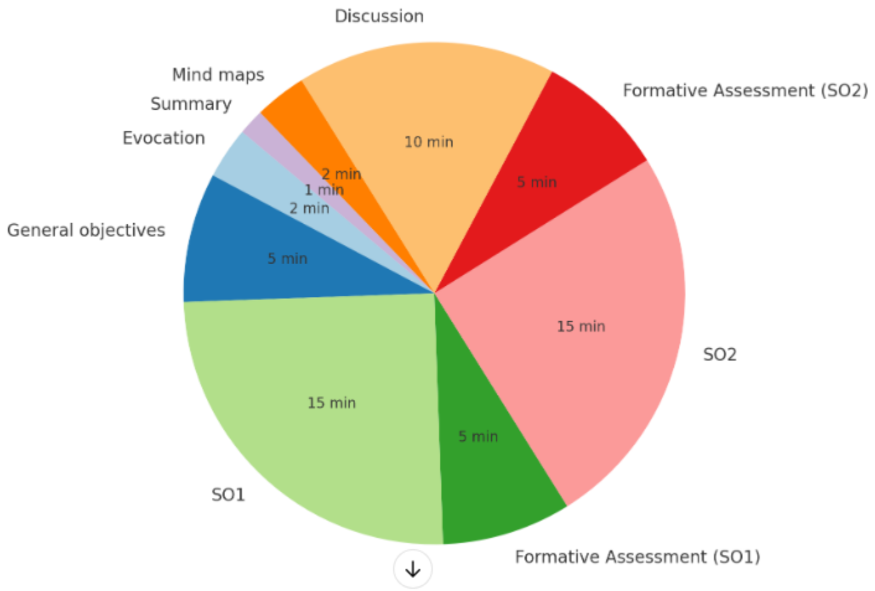
**Course In-charge** **Approved by HoD.**

S. Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

## LESSON PLAN
## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | **III** |
| **Subject Title** | **Introduction to Computers** |
| **Code** | **21UCSN31** |
| **Hours** | **2** |
| **Total Hours** | **30** |
| **Credits** | **2** |
| **Max Marks** | **50** |
| **Unit & Title** | Unit: 1II – Different types Printer |
| **Name of the Faculty** | **Sr.K.Reetha** |
| **T-L tools** | Lecture method, **Visual aid**: Picture showing different types of printer |

## Micro –planning



### 1. Evocation (2 minutes):

- **Objective**: Activate prior knowledge and engage students.
- **Activity**:

    - Ask: “What types of printers do you know or use at home/school/work?”
    - Display images of common printers (inkjet, laser, dot matrix, 3D printer) and ask students to identify them.

**2. General Objectives (5 minutes):**

- **Objective**: Set the context for the lesson.
- **Activity**:
    - Introduce the key objective: To learn about the different types of printers, their features, uses, and advantages/disadvantages.
    - Explain the significance of choosing the right printer for different needs (home, office, industrial).

**3. SO1: Overview of Printer Types (15 minutes):**

- **Objective**: Provide an overview of the various printer types.
- **Activity**:
    - **Lecture/Discussion**: Explain the basic types of printers:
        - **Inkjet Printers**: Popular for home use, good for color prints (photos, documents).
        - **Laser Printers**: Efficient for high-volume printing, faster, and cost-effective for text-based documents.
        - **Dot Matrix Printers**: Older technology, used for multipart forms, and noisy.
        - **3D Printers**: New technology used for creating three-dimensional objects.
    - Compare and contrast their features, uses, and costs.

**4. Formative Assessment (5 minutes):**

- **Objective**: Check understanding of printer types.
- **Activity**:
    - Quick interactive poll or quiz: “Which printer type would you choose for printing high-quality photos at home?”
    - Students respond and briefly justify their choice.

**5. SO2: Printer Features and Use Cases (15 minutes):**

- **Objective**: Dive into the specific features and use cases of each printer type.
- **Activity**:
    - **Group Work**: Assign each group a printer type (Inkjet, Laser, Dot Matrix, 3D) and have them research the following:
        - Key features of the printer type (speed, print quality, cost per page).
        - Common uses (home, business, industrial).
        - Advantages and disadvantages.
    - Groups present their findings to the class.

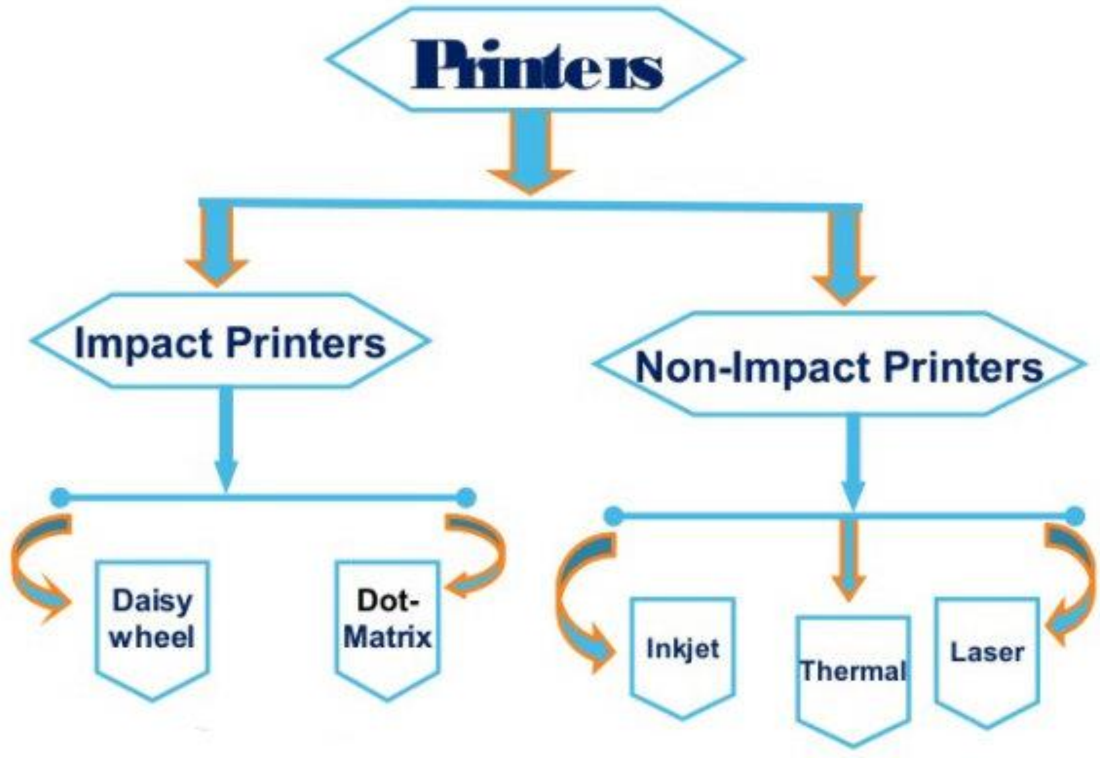
**6. Formative Assessment (5 minutes):**

- **Objective**: Evaluate understanding of printer features and uses.
- **Activity**:
  - Quick "exit ticket" or quiz: Students write down one advantage and one disadvantage of a specific printer type.

**7. Discussion (10 minutes):**

- **Objective**: Facilitate a discussion on real-world choices of printers.
- **Activity**:
  - Ask students: "If you were setting up an office or home printing system, which printer would you choose and why?"
  - Discuss factors like cost, speed, volume, and print quality in the decision-making process.

**8. Mind Maps (2 minutes):**



- **Objective**: Help students visualize and organize the types of printers and their features.
- **Activity**:
  - Ask students to create a simple mind map summarizing the different printer types and their key features (e.g., print quality, use cases, speed).

**9. Summary (1 minute):**

- **Objective**: Recap the key points of the lesson.
- **Activity**:

- Summarize the main types of printers (Inkjet, Laser, Dot Matrix, 3D) and their specific use cases.
- Reinforce the idea that the choice of printer depends on factors like purpose, cost, and volume of printing.

**10.Discussion:**

The students will be asked questions regarding the types of printers and its applications. One of the students will be asked to explain any one application studied during the class.

**11.FAQ's:**

**1. Which type of printer uses a ribbon and print head to strike an image onto paper?**

A) Inkjet Printer
B) Laser Printer
C) Dot Matrix Printer
D) Thermal Printer

**Answer:** C) Dot Matrix Printer

---

**2. Which printer is best suited for high-speed, high-volume printing in an office environment?**

A) Inkjet Printer
B) Laser Printer
C) Thermal Printer
D) 3D Printer

**Answer:** B) Laser Printer

---

**3. What is a major disadvantage of an inkjet printer?**

A) Slow printing speed and expensive ink cartridges
B) Cannot print in color
C) Very large in size
D) Uses a lot of electricity

**Answer:** A) Slow printing speed and expensive ink cartridges

---

**4. Thermal printers are commonly used for which of the following?**

A) Printing official documents
B) Printing receipts and labels
C) Printing high-quality photos
D) Printing multi-part forms

**Answer:** B) Printing receipts and labels

---

**5. Which of the following printers creates objects layer by layer using additive manufacturing?**

A) Laser Printer
B) Inkjet Printer
C) 3D Printer
D) Dot Matrix Printer

**Answer:** C) 3D Printer

**12.References: (Books/Periodicals/Journals)**

1. Alexis Leon & Mathews Leon. *Introduction To Computers*. India: McGraw Hill Education Private Limited. Fifth Reprint, Edition 2008.

2. Dr.P.Velmani.,(Assistant Professor),M.C.A.,M.Phil.,Ph.D. *Computers Bascis to Advancements*. India: Chess Educational Publishers. First Edition.

3. Peter Norton's. *Introduction to computers* .India: New Delhi: Tata McGraw-Hill. Edition 2004

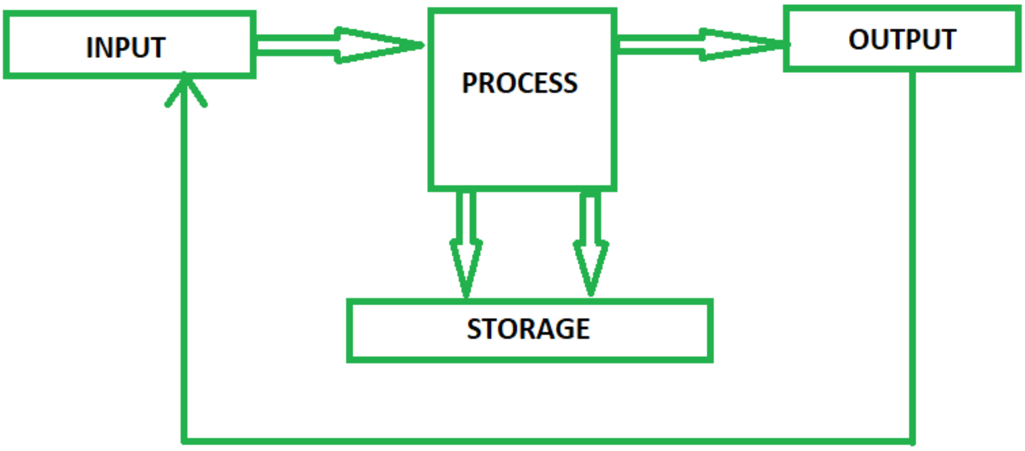
**13.1Taxonomy of objectives:**

| **Taxonomy of Objectives** | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyze** | **Evaluate** | **Create** |
| A. Factual Knowledge | **1,2** | | | **1,2** | **1,2** | |
| B. Conceptual Knowledge | | **1,2** | **1,2** | | **1,2** | |
| C. Procedural Knowledge | | | | **1,2** | | |

**13.2 Key words:**

Data mining, neural network, supervised learning, unsupervised learning, reinforcement learning, clustering, feature engineering, classification, regression, anomaly detection, dimensionality reduction, convolutional neural network (CNN), natural language processing (NLP), overfitting, underfitting, and deep learning.

**13.3 Key diagrams (if any):**



**Types of Printers**

Use slides, videos, or real-life examples to explain the following types of printers:

**A. Impact Printers (Older Technology)**

1. **Dot Matrix Printer**
   1. Works by striking an ink ribbon against paper.
   2. Used for invoices, receipts, and industrial applications.
   3. **Pros:** Low cost, can print multi-part forms (carbon copies).
   4. **Cons:** Noisy, low-quality output, slow speed.

**B. Non-Impact Printers (Modern Technology)**

**Inkjet Printer**

1. Sprays tiny ink droplets onto paper.
2. Common for home and office use.

3. **Pros:** High-quality color printing, affordable.
4. **Cons:** Ink cartridges are expensive, slower than laser printers.

### Laser Printer

1. Uses laser beams and toner to print.
2. Used in offices for high-speed, high-quality prints.
3. **Pros:** Fast, cost-effective for bulk printing, sharp text.
4. **Cons:** Expensive initial cost, large size.

### Thermal Printer

1. Uses heat to print on special paper (used in receipts, labels).
2. **Pros:** No ink required, fast, quiet.
3. **Cons:** Prints fade over time, limited color options.

### 3D Printer

1. Uses additive manufacturing to create three-dimensional objects.
2. Used in engineering, medical, and prototyping industries.
3. **Pros:** Customization, reduced material waste.
4. **Cons:** Expensive, slow process.

**14.Drive Link:**
**https://docs.google.com/presentation/d/17OZkbLhjjQ8_nQhMgBXzRClIFfnQeieV/edit?usp=sharing&ouid=104648840415631957301&rtpof=true&sd=true**

S. Gnanathayan

Reetha

**Course In-charge**

**Approved by HoD.**

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

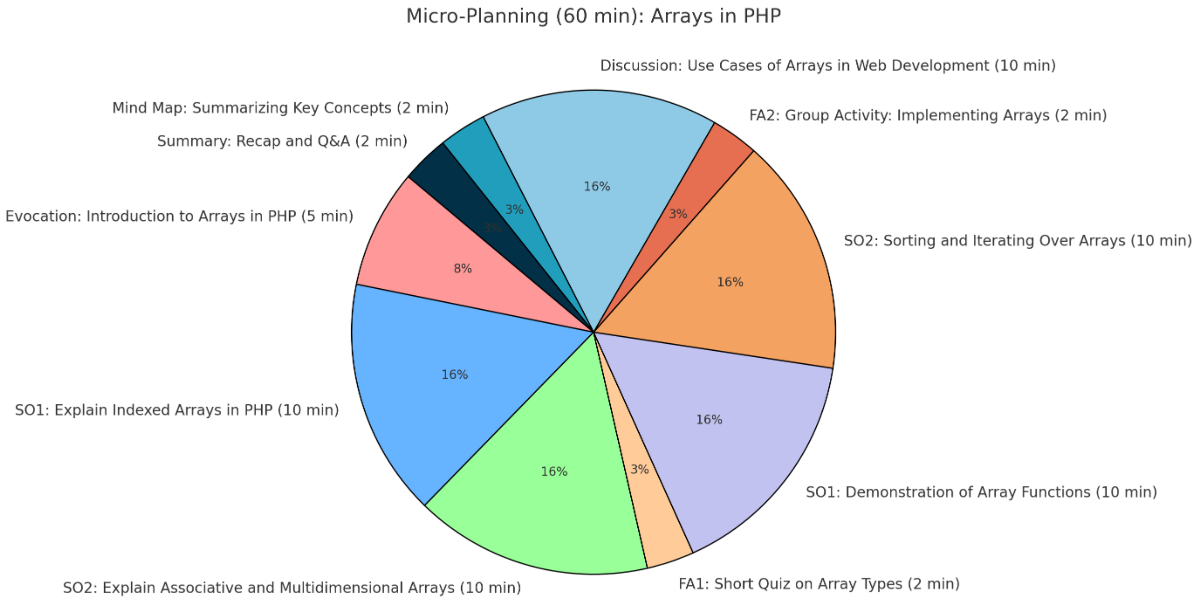
## Lesson Plan

| | |
|---|---|
| Programme | B. Sc. Computer Science |
| Semester | IV |
| Course Title | RDBMS with PHP and MySQL |
| Code | 21UCSC41 |
| Hours | 4 |
| Total Hours | 60 |
| Credits | 4 |
| Max Marks | 100 |
| Unit & Title | Unit III: Arrays in PHP |
| Name of the Faculty | S.Gnanathangam |
| T-L tools | Mind Maps, Video, PowerPoint Presentation, Live Coding Examples |

### Pre-requisite Knowledge:

- Basic knowledge of PHP syntax and Semantics.
- Logical thinking

### Micro-Planning: (60 minutes)



| Phase | Activity | Duration |
|---|---|---|
| Evocation | Introduction to Arrays in PHP | 5 min |
| SO1 | Explain Indexed Arrays in PHP | 10 min |
| SO2 | Explain Associative and Multidimensional Arrays | 10 min |

|  |  |  |
|---|---|---|
| FA1 | Short Quiz on Array Types | 2 min |
| SO1 | Demonstration of Array Functions | 10 min |
| SO2 | Sorting and Iterating Over Arrays | 10 min |
| FA2 | Group Activity: Implementing Arrays | 2 min |
| Discussion | Use Cases of Arrays in Web Development | 10 min |
| Mind Map | Summarizing Key Concepts | 2 min |
| Summary | Recap and Q&A | 2 min |

1**. Evocation (5 min)**

- Ask students where they store multiple pieces of related data in real-world applications.
- Explain how arrays help in managing structured data in PHP.

.

**2. Topic Introduction:**

**2.1 General Objective:**

- To understand the types and usage of arrays in PHP.
- To learn the syntax of arrays.

**2.2** Specific Outcomes (SO1 & SO2):

- Explain the concept of Indexed, Associative, and Multidimensional Arrays.
- Demonstrate the use of array functions and sorting techniques in PHP.

**First Phase:** Explanation (20 min)

SO1 (10 min):

Indexed Arrays:

- Syntax and declaration ($arr = array ("Apple", "Banana", "Cherry");).
- Accessing elements using index numbers.

Associative Arrays:

- Key-value pairs ($student = array ("name" => "John", "age" => 21);).
- Accessing elements using keys.

Multidimensional Arrays:

Nested arrays ($matrix = array (array (1,2,3), array (4,5,6)) ;).

SO2 (10 min):

1. Common Array Functions:

   count (), array_push(), array_pop(), array_merge().

2. Sorting Arrays:

   sort (), rsort(), asort(), ksort().

**Second Phase:** Implementations and Application (20 min)

SO1 (10 min): Live Coding Demo:

- Create an indexed array of colors and print them using foreach().
- Use array_merge() to combine two arrays.

SO2 (10 min): Sorting and Iteration:

- Demonstrate asort() for sorting associative arrays.
- Explain how foreach () loops work with arrays.

Mind Map (2 min)

Draw a mind map that summarizing the types of arrays and key functions.

Summary (2 min)

Recap key points:

- Indexed, Associative, and Multidimensional Arrays.
- Useful array functions like count(), array_merge(), and sort().

Open Q&A session.

**2.3. Taxonomy of Objectives:**

| Taxonomy of objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | 1 | 1 | | | | |
| B. Conceptual Knowledge | | 2 | | | | |
| C. Procedural Knowledge | | | | | 1 | |

**2.4. Key Words:**

- Indexed Arrays
- Associative Arrays
- Multidimensional Arrays
- Array Functions
- Sorting Arrays

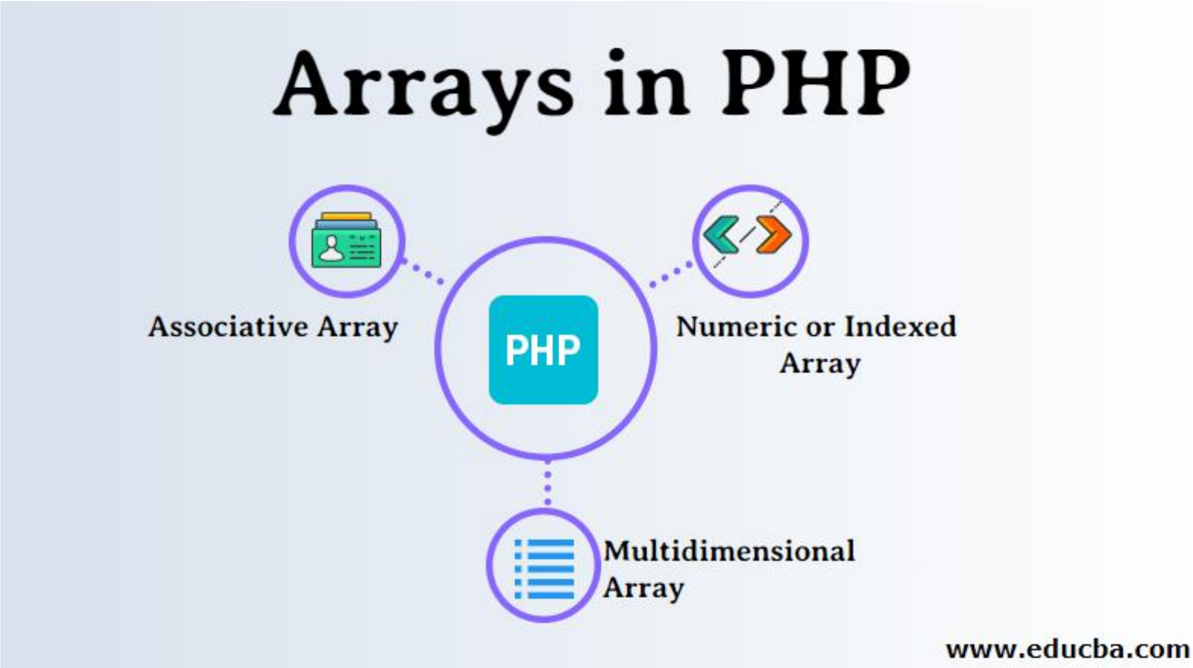
### 2.5 Powerpoint Presentation

https://docs.google.com/presentation/d/1No1pJeKy4Y_mBEDlGttHGtMlJCA-VUfZ/edit?usp=sharing&ouid=115208675943936491706&rtpof=true&sd=true

## 3. Discussion (10 min)

- Where are arrays used in real-world web applications?
- How do arrays help in storing and managing form data?
- Why are associative arrays useful in PHP?

## 4. Mind Map:



Here's a simple mind map:

```
        PHP Arrays
            |
  ----------------------
  |        |        |
Indexed   Associative   Multidimensional
  |        |        |
[1,2,3]   ["name"=>"John"]  [["John", 25], ["Jane", 30]]
```

## 5. Summary:

### 1. Introduction to Arrays

- Array Definition
- Usage

- Real-world examples (e.g., storing a list of students' names).

**Example:**

```
$fruits = array ("Apple", "Banana", "Cherry");

echo $fruits [0]; // Outputs: Apple
```

## 2. Types of Arrays in PHP

### 1. Indexed Arrays

Simple array with numerical indexes.

Example

```
$colors = ["Red", "Green", "Blue"];

echo $colors[1]; // Green
```

### 2. Associative Arrays

Arrays with named keys.

```
$ages = ["John" => 25, "Jane" => 30];

echo $ages["Jane"]; // 30
```

### 3.Multidimensional Arrays

```
php
CopyEdit
$students = [
   ["John", 25, "A"],
   ["Jane", 30, "B"]
];
echo $students[1][0]; // Jane
```

### 4.Array Functions:

1. count($array): Get number of elements.
2. array_push($array, $value): Add elements.
3. array_pop($array): Remove last element.
4. array_merge($array1, $array2): Merge arrays.
5. sort($array): Sorting arrays.

   **Example:**

```
$numbers = [3, 1, 4, 2];
sort($numbers);
print_r($numbers); // Outputs: [1, 2, 3, 4]
```

**Hands-on Practice**

- **Exercise 1:** Create an indexed array of your top 5 favourite movies and print them.
- **Exercise 2:** Create an associative array with countries and capitals, then display a specific capital.

- **Exercise 3:** Create a multidimensional array representing students (name, age, grade) and display specific values.

**Q&A and Recap (5 min)**

- Recap key points:
  - Indexed, Associative, and Multidimensional arrays.
  - Common array functions.
- Address students' doubts.

**6. Assessment through questions/analogy/new ideas:**

Formative Assessment 1 (FA1) (2 min)

Short quiz with 3-5 multiple-choice questions on PHP arrays

1. Where are arrays used in real-world web applications?
2. How do arrays help in storing and managing form data?
3. What are associative arrays ?

Formative Assessment 2 (FA2) (2 min)

Group Activity: "Create and display a list of students with names and grades using an associative array."

**7. FAQ's: MCQ's/ Descriptive questions:**

1.Implement PHP Program using arrays.

2.Compare associative and indexed array.

**8. References:**

1. W3Schools PHP Arrays: https://www.w3schools.com/php/php_arrays.asp
2. PHP Official Documentation: https://www.php.net/manual/en/language.types.array.php
3. Learning PHP, MySQL & JavaScript – Robin Nixon

**9.Verified by Subject Expert:**

S. Gnanathayan

**Course In-charge**

S. Gnanathayan

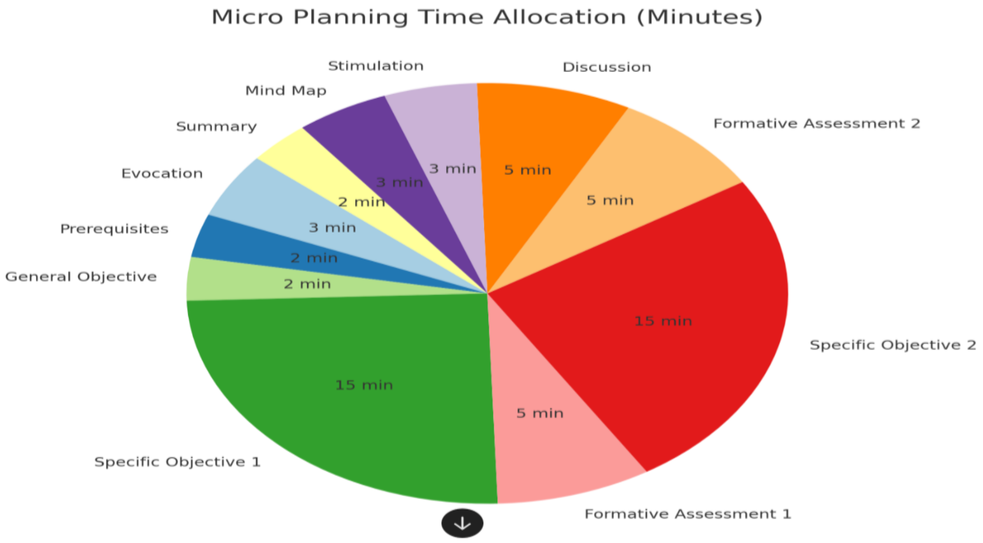
**Approved by HoD**

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

# LESSON PLAN
## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | **IV** |
| **Subject Title** | **Big Data Analytics** |
| **Code** | **21UCSA41** |
| **Hours** | **3** |
| **Total Hours** | **45** |
| **Credits** | **3** |
| **Max Marks** | **100** |
| **Unit & Title** | Unit: 1 – Different types of Topology |
| **Name of the Faculty** | **Dr.Anila Maily** |
| **T-L tools** | Lecture method, **Visual aid**: Picture showing application of machine learning |

Micro -planning



|  |  |
|---|---|
| Evocation | 3min |
| Prerequisties | 2min |
| General Objective | 2min |
| Specific objective1 | 15min |
| Formative assessment1 | 5min |
| Specific objective2 | 15min |
| Formative assessment2 | 5min |
| Discussion | 5min |
| Stimulation | 3min |
| Mind map | 3min |
| Summary | 2min |

**1. Evocation (3 min)**

- **Objective:** Engage students and activate prior knowledge.
- **Activity:** Ask students where they have encountered ML in real life (e.g., recommendations on YouTube, Siri, chatbots).
- **ML Tool:** AI-based sentiment analysis of student responses.

**2. Prerequisites (2 min)**

- **Objective:** Ensure students understand key ML concepts.
- **Activity:** Quick poll or quiz (e.g., What is ML? Difference between AI and ML?).
- **ML Tool:** AI-powered quiz (e.g., Kahoot, Google Forms with AI-based suggestions).

**3. General Objective (2 min)**

- **Objective:** Define the learning goal.
- **Activity:** Explain that students will explore ML applications in different domains.

**4. Specific Objective 1 (15 min)**

- **Objective:** Discuss ML applications in **healthcare, finance, and marketing**.
- **Activity:** Present real-world case studies:
  - **Healthcare:** AI-powered diagnosis (e.g., IBM Watson, medical image analysis).
  - **Finance:** Fraud detection and algorithmic trading.
  - **Marketing:** Recommendation systems (e.g., Netflix, Amazon).
- **ML Tool:** AI-powered chatbot to answer students' queries.

**5. Formative Assessment 1 (5 min)**

- **Objective:** Check understanding.
    - **Activity:** Students match ML applications with correct industries in a gamified quiz.

**6. Specific Objective 2 (15 min)**

- **Objective:** Explore ML applications in **education, robotics, and transportation**.
- **Activity:**
    - **Education:** Adaptive learning systems (e.g., Coursera AI tutors).
    - **Robotics:** Self-learning robots (e.g., Tesla's humanoid robot).
    - **Transportation:** Self-driving cars, traffic prediction.
- **ML Tool:** AI-driven simulation of an ML model in action.

**7. Formative Assessment 2 (5 min)**

- **Objective:** Assess knowledge through student participation.
- **Activity:** Case study discussion – students predict how ML will transform a new industry.

**8. Discussion (5 min)**

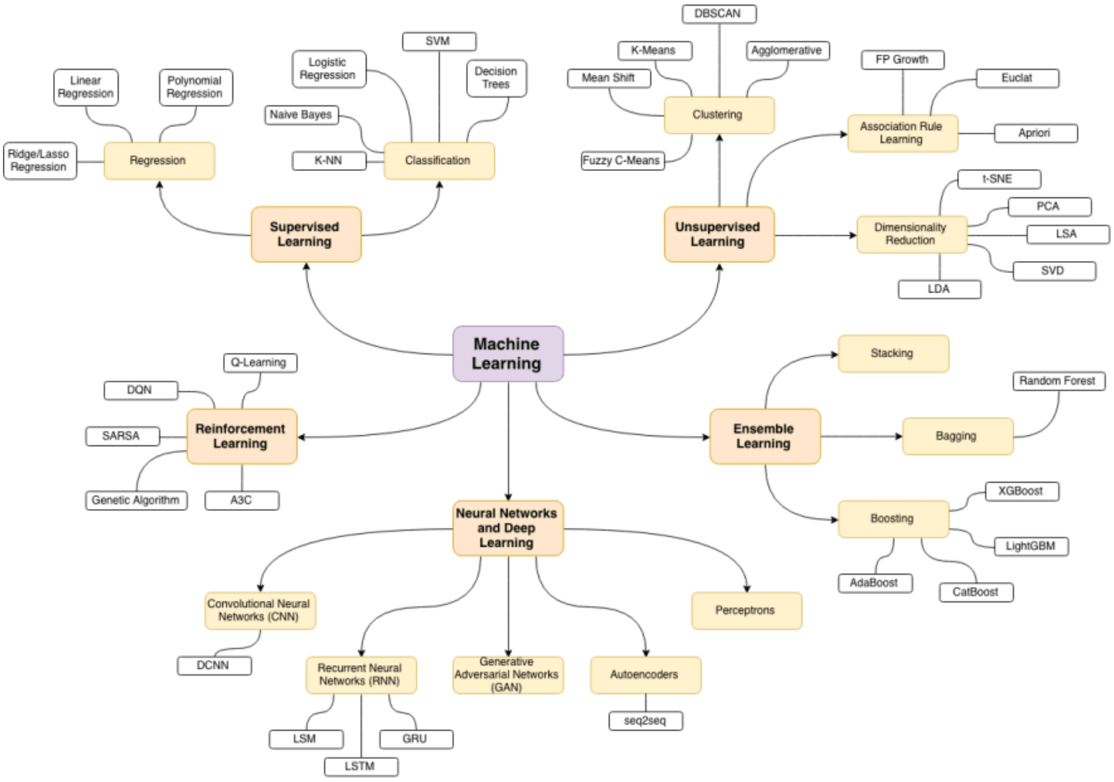
- **Objective:** Encourage critical thinking.
- **Activity:** Debate on ML’s ethical concerns (e.g., privacy, bias in AI).

**9. Stimulation (3 min)**

- **Objective:** Inspire future learning.
- **Activity:** Show a short video on **the future of AI and ML**.

**10. Mind Map (3 min)**

- **Objective:** Visualize knowledge.
- **Activity:** Students create a mind map of ML applications.
- **ML Tool:** AI-generated mind map software (e.g., MindMeister).



## 11. Summary (2 min)

- **Objective:** Reinforce key takeaways.
- **Activity:** AI-generated summary of key points.

## 12.FAQ’s:

**1. What is the main difference between supervised and unsupervised learning?**

A) Supervised learning requires labeled data, while unsupervised learning does not.
B) Supervised learning is faster than unsupervised learning.
C) Unsupervised learning requires labeled data, while supervised learning does not.
D) Supervised learning is used for clustering tasks.

**2. What is overfitting in machine learning?**

A) When a model performs well on training data but poorly on new, unseen data.
B) When a model performs well on new data but poorly on training data.
C) When the model has too few parameters.
D) When the model is trained on too little data.

**3. What is a feature in machine learning?**

A) A type of algorithm used to train models.
B) A part of the model used to calculate errors.

C) An input variable used to make predictions.
D) The output of the machine learning model.

**4. Which of the following is an example of a regression problem?**

A) Classifying emails as spam or not spam.
B) Predicting the price of a house based on features like size and location.
C) Grouping customers based on purchasing behavior.
D) Identifying whether an image contains a dog or a cat.

**5. What is a decision tree?**

A) A data structure used to organize training data.
B) A model used for decision-making that splits data into smaller subsets.
C) A method to visualize how data points are clustered.
D) A technique to reduce the dimensionality of data.

## 13.References: (Books/Periodicals/Journals)

1.Soraya Sedkaoui *Data Analytics and Big Data* -, Wiley – ISTE 2018.
2. Michael Minelli, Michele Chamboss, Ambiga Dhiraj , "*Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for today's businesses*" John Wiley , 2014.
3. *Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data*,EMC Education Services.
4. Avid Loshin, "*Big data analytics: From Strategic planning to enterprise integration with tools, techniques, NoSQL, and Graph*, Elsevier,2013

## 14.1Taxonomy of objectives:

| **Taxonomy of Objectives** | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyze** | **Evaluate** | **Create** |
| A. Factual Knowledge | **1** | | | **1** | **1** | |
| B. Conceptual Knowledge | | **1** | **1** | | **1** | |
| C. Procedural Knowledge | | | | **1** | | |
| D. Meta Cognitive Knowledge | | | | | **1** | |

## 14.2: Key words:

Data mining, neural network, supervised learning, unsupervised learning, reinforcement learning, clustering, feature engineering, classification, regression, anomaly detection, dimensionality reduction, convolutional neural network (CNN), natural language processing (NLP), overfitting, underfitting, and deep learning.

## 14.3: Key diagrams (if any):

### Types of Machine Learning

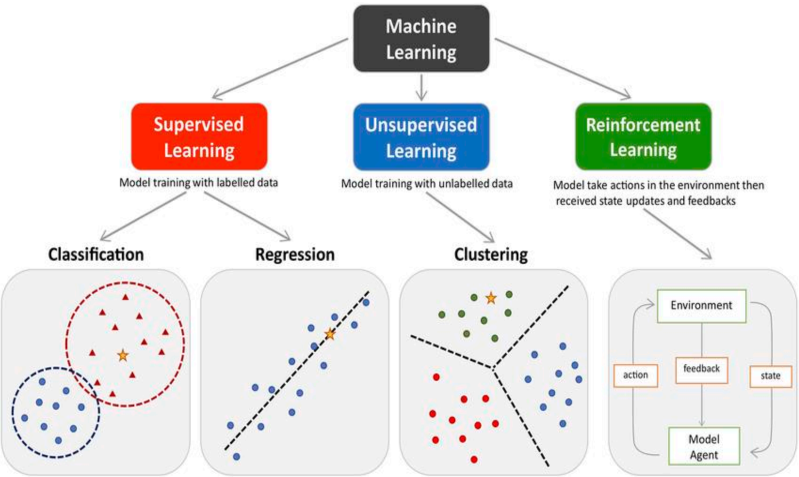
There are several types of machine learning, each with special characteristics and applications. Some of the main types of machine learning algorithms are as follows:

1.Supervised Machine Learning

2.Unsupervised Machine Learning

3.Reinforcement Learning

Additionally, there is a more specific category called semi-supervised learning, which combines elements of both supervised and unsupervised learning.



### Applications of Machine learning

### 1. Image Recognition:

Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, **Automatic friend tagging suggestion**:

### 2. Speech Recognition

Speech recognition is a process of converting voice instructions into text, and it is also known as "**Speech to text**", or "**Computer speech recognition**." At present, machine learning algorithms are widely used by various applications of speech recognition. **Google assistant**, **Siri**, **Cortana**, and **Alexa** are using speech recognition technology to follow the voice instructions.

### 3. Traffic prediction:

It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:

- Real Time location of the vehicle form Google Map app and sensors
- Average time has taken on past days at the same time.

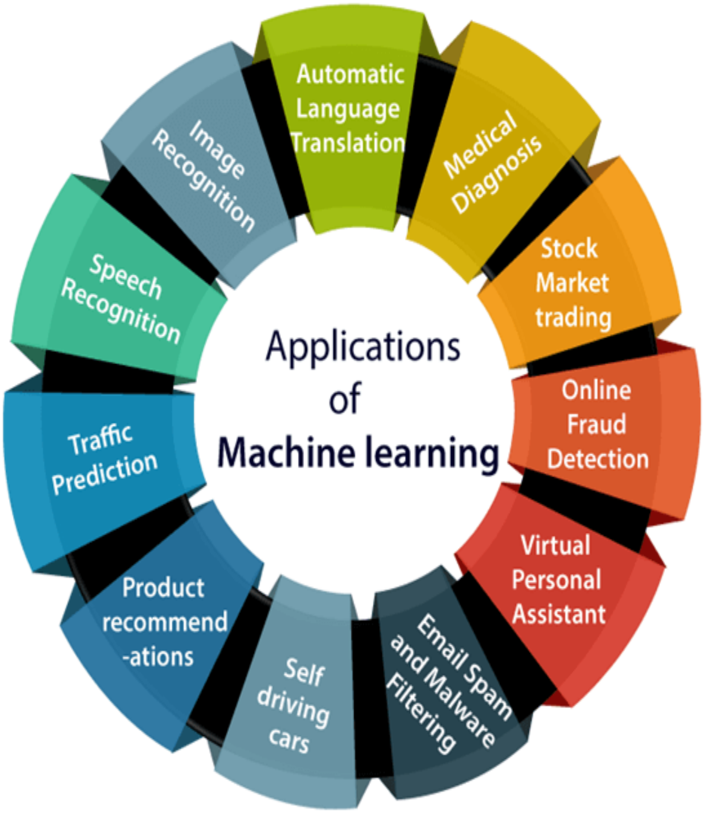
**4. Self-driving cars:**

One of the most exciting applications of machine learning is self-driving cars. Machine learning plays a significant role in self-driving cars. Tesla, the most popular car manufacturing company is working on self-driving car. It is using unsupervised learning method to train the car models to detect people and objects while driving.

**5. Email Spam and Malware Filtering:**

Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning. Below are some spam filters used by Gmail:

- Content Filter
- Header filter
- General blacklists filter
- Rules-based filters
- Permission filters


Applications of Machine learning
Automatic Language Translation
Medical Diagnosis
Stock Market trading
Online Fraud Detection
Virtual Personal Assistant
Email Spam and Malware Filtering
Self driving cars
Product recommend -ations
Traffic Prediction
Speech Recognition
Image Recognition

**15.Drive Link:**

https://docs.google.com/presentation/d/18O-DaGBVWUeFmo_NiKd_qdiJMLNHjWyn/edit?usp=sharing&ouid=104648840415631957301&rtpof=true&sd=true

S. Gnanathayan

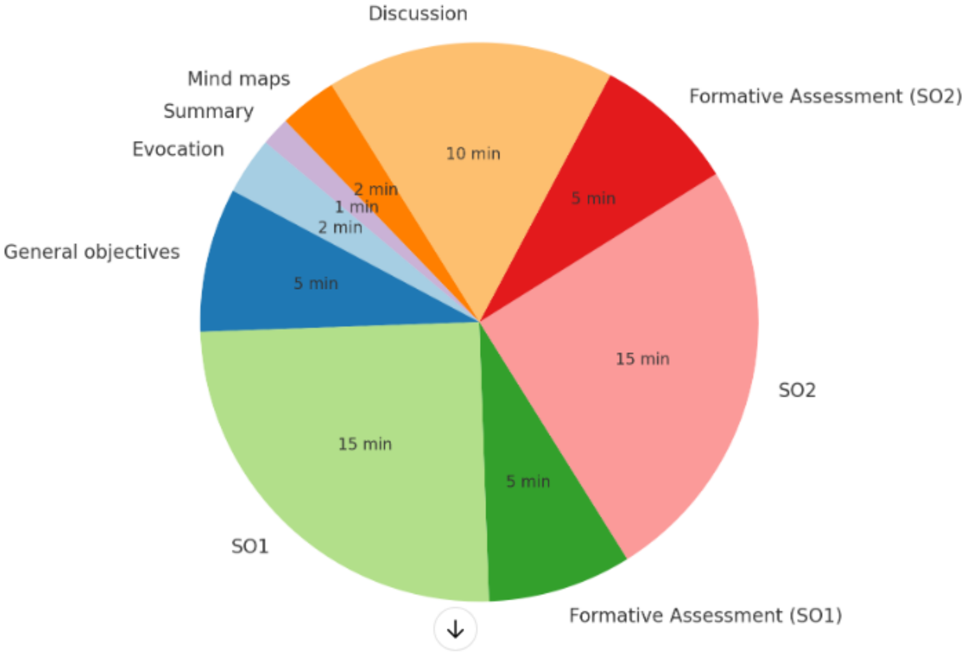
**Course In-charge** **Approved by HoD.**

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

## LESSON PLAN
## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | **III** |
| **Subject Title** | **Introduction to Internet** |
| **Code** | **21UCSN41** |
| **Hours** | **2** |
| **Total Hours** | **30** |
| **Credits** | **2** |
| **Max Marks** | **50** |
| **Unit & Title** | Unit: 1II – Different types Browsers |
| **Name of the Faculty** | **Ms.S.Gnanathangam** |
| **T-L tools** | Lecture method, **Visual aid**: Picture showing different types of printer,Mind Maps |

## Micro –planning



**1. Evocation (2 minutes):**

- **Objective**: Engage students with a quick question or visual to activate prior knowledge about browsers.
- **Activity**:
    - Ask: “What browsers do you use regularly, and why?”
    - Display logos of popular browsers (Chrome, Firefox, Safari, Edge) and ask students to identify them.

**2. General Objectives (5 minutes):**

- **Objective**: Set the stage for the lesson by explaining the purpose of understanding different browsers.
- **Activity**:
    - Introduce the key points: The different types of browsers (e.g., Chrome, Firefox, Safari, Opera, Edge) and their features.
    - Explain the significance of choosing the right browser for different needs (speed, security, privacy, extensions).

**3. SO1: Overview of Browsers (15 minutes):**

- **Objective**: Present an overview of different browsers and their basic functionalities.
- **Activity**:
    - **Lecture/Discussion**: Describe popular browsers (Chrome, Firefox, Safari, Edge, Opera).
    - Highlight unique features of each, such as:
        - **Google Chrome**: Speed, Google integration, extensive extension support.
        - **Mozilla Firefox**: Open-source, privacy-focused, customization options.
        - **Safari**: Optimized for Apple devices, energy-efficient.
        - **Microsoft Edge**: Integrated with Windows, new Chromium-based features.
        - **Opera**: Built-in VPN, ad blocker, and customization.

**4. Formative Assessment (5 minutes):**

- **Objective**: Check students’ understanding of browser types.
- **Activity**:
    - Quick quiz or poll: “Which browser would you choose for speed, privacy, or security, and why?”
    - Allow students to share responses briefly.

**5. SO2: Browser Features Comparison (15 minutes):**

- **Objective**: Dive deeper into the specific features of different browsers.
- **Activity**:
    - **Group Work**: Assign each group a browser and ask them to create a list of features and benefits.

  - Discuss the importance of speed, security, extensions, and user-friendliness for each browser.

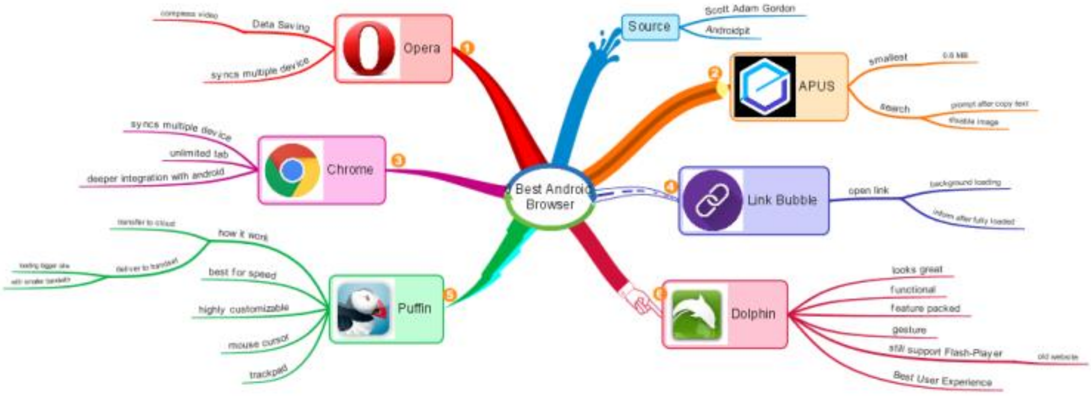
**6. Formative Assessment (5 minutes):**

- **Objective**: Assess understanding of the detailed browser features.
- **Activity**:
  - Ask students to identify which features of a specific browser (from their group activity) they find most useful and why.

**7. Discussion (10 minutes):**

- **Objective**: Engage students in a deeper conversation about their browser preferences.
- **Activity**:
  - Discuss which browsers students prefer and why.
  - Encourage them to consider factors such as security, compatibility with devices, and personal habits.

**8. Mind Maps (2 minutes):**



- **Objective**: Help students organize and visualize the different browsers and their features.
- **Activity**:
  - Ask students to create a quick mind map in their notes, showing the different browsers and their unique features (e.g., speed, security, compatibility).

**9. Summary (1 minute):**

- **Objective**: Recap the key points.
- **Activity**:
  - Summarize the different browsers, their strengths, and considerations for choosing one based on user needs.

**10.FAQ's:**

**1. Which of the following is an example of a graphical web browser?**

A) Lynx
B) Google Chrome
C) w3m
D) Telnet

**Answer:** B) Google Chrome

---

**2. What is the primary function of a web browser?**

A) Sending emails
B) Storing data permanently
C) Accessing and displaying web pages
D) Creating web applications

**Answer:** C) Accessing and displaying web pages

---

**3. Which browser is known for prioritizing privacy by using the Tor network?**

A) Mozilla Firefox
B) Microsoft Edge
C) Tor Browser
D) Opera

**Answer:** C) Tor Browser

---

**4. Which protocol is commonly used by web browsers to retrieve web pages?**

A) FTP
B) HTTP/HTTPS
C) SMTP
D) TCP/IP

**Answer:** B) HTTP/HTTPS

---

**5. What is the purpose of the incognito or private browsing mode in web browsers?**

A) To browse the internet faster
B) To prevent the browser from saving history, cookies, and cache

C) To block all pop-up ads
D) To download files anonymously

**Answer:** B) To prevent the browser from saving history, cookies, and cache

## 11.References: (Books/Periodicals/Journals)

1. Alexis Leon & Mathews Leon. *Internet for Everyone*. India: Leon Press.15th Anniversary Edition.

2.*Computer Literacy*, Department of Foundation Courses in collaboration with School of Computing Sciences

3.Vikas Gupta. *Internet and Web design*, India: Rematch Press I. Edition 2003.

4.Rajeev Gupta B.Tech. *Internet Guide,* India: Copyright reserved Nipun Publications. First Edition November 2000.

## 12.1Taxonomy of objectives:

| **Taxonomy of Objectives** | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyze** | **Evaluate** | **Create** |
| A. Factual Knowledge | **1,2** | | | **1,2** | **1,2** | |
| B. Conceptual Knowledge | | **1,2** | **1,2** | | **1,2** | |
| C. Procedural Knowledge | | | | **1,2** | | |

## 12.2 Key words:

Web Browser, URL (Uniform Resource Locator) , HTTP (Hypertext Transfer Protocol) ,HTTPS (Hypertext Transfer Protocol Secure) ,Search Engine, Tabs

## 12.3 Key diagrams (if any):

### 1. Graphical Web Browsers

These browsers display text, images, videos, and interactive content using a graphical user interface (GUI).

**Examples:**

- **Google Chrome** – Fast, widely used, integrates with Google services.
- **Mozilla Firefox** – Open-source, focuses on privacy and customization.
- **Microsoft Edge** – Built-in with Windows, AI-powered features.
- **Apple Safari** – Optimized for macOS and iOS devices.
- **Opera** – Has a built-in VPN and ad blocker.

### 2. Text-Based Browsers

These browsers display only text and do not support images or multimedia, mainly used in command-line environments.

**Examples:**

- **Lynx** – A text-only browser used for fast browsing and accessibility.
- **w3m** – Supports tables and some text formatting in terminal environments.

## 14.Drive Link

https://docs.google.com/presentation/d/1a_2YrP9F1RTm3_IrXzrvkheLgKQ8hKlD/edit?usp=sharing&ouid=104648840415631957301&rtpof=true&sd=true

S. Gnanathayan

S. Gnanathayan

**Course In-charge**

**Approved by HoD.**

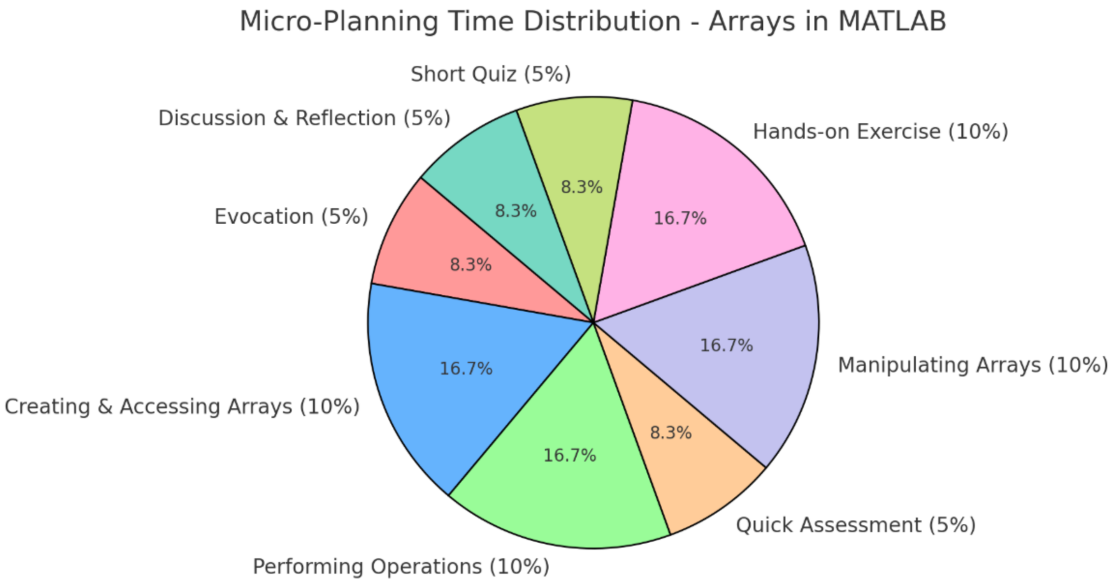
Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

| | |
|---|---|
| Programme | B.Sc |
| Semester | V |
| Course Title | Computer Oriented Numerical Methods |
| Code | 21UCMC51 |
| Hours | 6 |
| Total Hours | 90 |
| Credits | 5 |
| Max Marks | 100 |
| Unit & Title | IV / Introduction to MATLAB |
| Name of the Faculty | Sr.K.Reeta |
| T-L tools | PowerPoint, MATLAB Software ,Hands-on Lab Exercises |

**Pre-requisite Knowledge:**

- Basic understanding of MATLAB environment
- Familiarity with variables and basic mathematical operations

**Micro Planning:** 60 minutes



| Phase | Time |
|---|---|
| **Evocation** – Introduction to Arrays | 5 min |
| **SO1** – Creating and Accessing Arrays | 10 min |
| **SO2** – Performing Operations on Arrays | 10 min |
| **FA1** – Quick Assessment (Q&A) | 5 min |
| **SO1** – Manipulating and Modifying Arrays | 10 min |
| **SO2** – Practical Hands-on Exercise | 10 min |
| **FA2** – Short Quiz | 5 min |
| **Discussion & Reflection** | 5 min |

### 1. Evocation (5 min) – Activating Prior Knowledge

- **Ask students:**
  - Have you worked with arrays in any programming language before?
  - Why do we use arrays in programming?
- **Discussion:**
  - MATLAB is optimized for numerical computing, and arrays are its fundamental data structure.
- **Real-World Applications:**
  - Image processing, signal processing, data analysis, and simulations.

### 2. SO1 – Creating and Accessing Arrays (10 min)

- **Concepts Covered:**
  - Row vectors: A = [1 2 3 4]
  - Column vectors: B = [1; 2; 3; 4]
  - Matrices: C = [1 2; 3 4]
  - Special Function in arrays**: zeros(n,m), ones(n,m), rand(n,m), eye(n)**
- **Demonstration:** Creating a **3×3 identity matrix** in MATLAB.
- **Activity:** Students create their own **row and column vectors**.

### 3. SO2 – Performing Operations on Arrays (10 min)

- **Element-wise Operations:**

  - `C = A + B` (Addition)
  - `C = A .* B` (Multiplication)
  - `C = A .^ 2` (Power)

- **Matrix Operations:**

  - `C = A * B` (Matrix Multiplication)
  - `A'` (Transpose)

- **Activity:** Students perform element-wise multiplication on two matrices.

### 4. FA1 – Quick Assessment (Q&A) (5 min)

- **Ask students:**
  - What is the difference between A * B and A .* B?
  - How do you extract a subarray from a matrix?

### 5. SO1 – Manipulating and Modifying Arrays (10 min)

- **Indexing and Slicing:**
  - Accessing specific elements: A(2,3)
  - Extracting rows/columns: A(:,2), A(1,:)
  - Modifying elements: A(1,2) = 99
- **Hands-on:** Students extract a submatrix from a given matrix.

### 6. SO2 – Practical Hands-on Exercise (10 min)

- **Task:** Write a MATLAB script that:
  - Creates a 4×4 random matrix
  - Finds the sum of all elements
  - Extracts the second row and third column separately.

### 7. FA2 – Short Quiz (5 min)

- **Multiple-choice or short-answer questions on:**
  - Difference between matrix and element-wise operations
  - Extracting subarrays
  - Common MATLAB functions (zeros(), ones(), etc.)

### 8. Discussion & Reflection (5 min)

- **Ask students:**
  - How does MATLAB simplify array operations?
  - Where can they see themselves using arrays in future projects?
- **Summarize key takeaways:**
  - Arrays are the core data structure in MATLAB.
  - MATLAB provides built-in functions for array creation and manipulation.
  - Operations on arrays are optimized for numerical computin

**Assessment through Questions & Exercises**

1. Describe the difference between row and column vectors in MATLAB.
2. Write a MATLAB command to extract the second column from a matrix.
3. Explain the difference between **A * B** and **A .* B**.
4. Write a MATLAB script that creates a **5×5** matrix of random values and finds the maximum value.

**Mind Map:**

```
Arrays in MATLAB
│
├── 1. Definition
│   ├── Fundamental data structure
│   ├── Used for numerical computations
│
├── 2. Types of Arrays
│   ├── Row Vector → [1 2 3 4]
│   ├── Column Vector → [1; 2; 3; 4]
│   ├── Matrix → [1 2; 3 4]
│   ├── Special Matrices
│   │   ├── zeros(n,m) → Matrix of zeros
│   │   ├── ones(n,m) → Matrix of ones
│   │   ├── rand(n,m) → Matrix of random numbers
│   │   ├── eye(n) → Identity matrix
│
├── 3. Accessing and Modifying Arrays
│   ├── Indexing → A(row, column)
│   ├── Slicing → A(:,2) (Extract second column)
│   ├── Modifying → A(1,2) = 99
│
├── 4. Operations on Arrays
│   ├── Element-wise Operations
│   │   ├── Addition → C = A + B
│   │   ├── Multiplication → C = A .* B
│   │   ├── Power → C = A .^ 2
│   ├── Matrix Operations
│   │   ├── Matrix Multiplication → C = A * B
│   │   ├── Transpose → A'
│
├── 5. Built-in Functions
│   ├── sum(A) → Sum of elements
│   ├── mean(A) → Mean of elements
│   ├── max(A) → Maximum value
│   ├── min(A) → Minimum value
│
└── 6. Applications
├── Data Analysis → Statistical calculations
├── Image Processing → Pixel manipulations
├── Engineering Simulations → Matrix computations
├── Machine Learning → Handling datasets
```

**Taxonomy of objectives:**

| Taxonomy of objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | | 1 | | | | |
| B. Conceptual Knowledge | | 2 | | | | |
| C. Procedural Knowledge | | | | | | 1 |

**Reference:**

1. MATLAB and its Applications in Engineering ,Raj Kumar Bansal, Ashok Kumar Goel, Mano Kumar Sharma-Pearson publication.
2. https://www.mathworks.com/help

**PPT Link :** **https://docs.google.com/presentation/d/1v-XYVr4XvUB5sKcnKcLnNDC4llCyrmH0/edit?usp=sharing&ouid=113456501758187936448&rtpof=true&sd=true**

**Verified by Subject Expert:**

Reetha

**Course In-charge**

Reetha

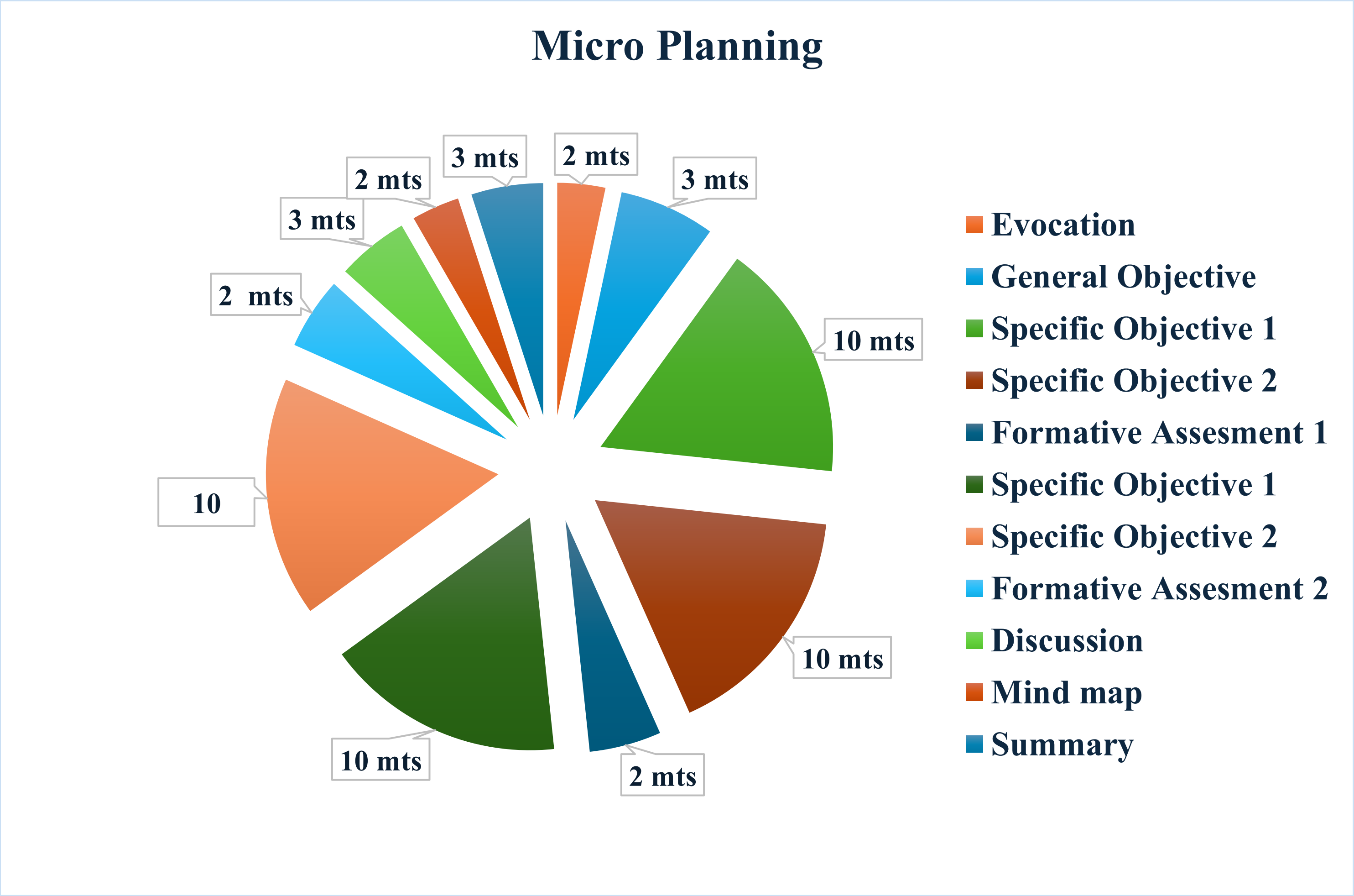
**Approved by HoD**

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

# LESSON PLAN

## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | V |
| **Course Title** | Core : Operating Systems |
| **Course Code** | 21UCSC51 |
| **Hours** | 4 |
| **Total Hours** | 60 |
| **Credits** | 4 |
| **Max Marks** | 100 |
| **Unit & Title** | Unit: II Process Management |
| **Name of the Faculty** | Ms. J. Anila Maily |
| **T-L tools** | Lecture method, **Visual aid**: PPT |



1. **Learning through evocation**

Learners are introduced to process and its various states.

## 2. Topic Introduction:

Students are introduced to the concept of CPU scheduling

### 2.1 General Objective:

Enables the students to evaluate various scheduling algorithms and choose the best one.

### 2.2 Specific Objectives:

Enables the students to:

1. To calculate the average waiting time of FCFS and SJFS
2. To evaluate various scheduling algorithm

First Phase:

SO1 : How CPU utilization can be maximised using multiprogramming.

SO2 : Types of scheduling algorithms

Second Phase:

SO1 : Describe FCFS and SJFS algorithms

SO2 : Calculate average waiting time of Round Robin scheduling with various time quantum and describe priority scheduling.

### 2.3: Taxonomy of objectives:

| Taxonomy of Objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyse** | **Evaluate** | **Create** |
| A. Factual Knowledge | 1,2 | | | | | |
| B. Conceptual Knowledge | | **1,2** | **1,2** | | | |
| C. Procedural Knowledge | | | 1,2 | 1,2 | | |
| D. Meta Cognitive Knowledge | | | | | **1,2** | |

### 2.4: Key words:

CPU scheduling, FCFS, SJFS, Round Robin Scheduling

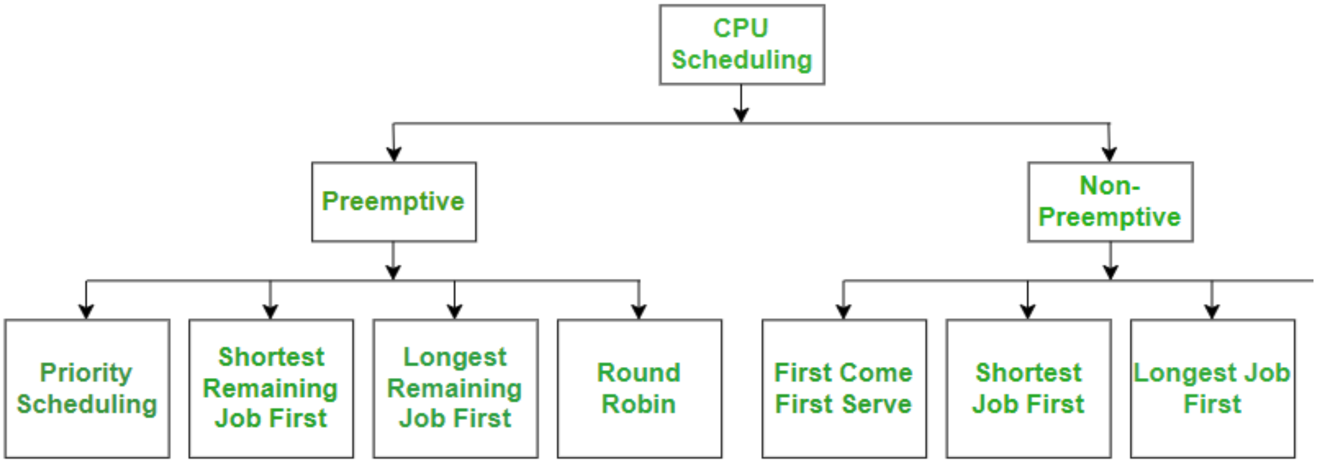
### 2.5: Key diagrams :

**PowerPoint Presentation:** https://docs.google.com/presentation/d/12YMor5R5VUGtgsuNZOMwp-LW4PHNjpXfzPSPmcPKJ9Q/edit?usp=share_link

3. **Discussion:**

The students will be asked to calculate average waiting time of round robin algorithm with various time quantum.

### 4. Mind Map:



### 5. Summary:

Students will be asked to evaluate average waiting time of various scheduling algorithms.

### 6. Assessment

**Formative assessment1 :** Describe FCFS and SJFS scheduling.

**Formative assessment2 :** Evaluate average waiting time of Round robin algorithm with time quantum =1, 4,10,20

### 7. FAQ's:

1. The processes that are residing in main memory and are ready and waiting to execute are kept on a list called _______________

a) job queue b) ready queue c) execution queue d) process queue

2. The interval from the time of submission of a process to the time of completion is termed as

a) waiting time b) turnaround time. c) response time d) throughput

3. Which scheduling algorithm allocates the CPU first to the process that requests the CPU first?

a) first-come, first-served scheduling b) shortest job scheduling
c) priority scheduling d) Round Robin scheduling

4. . Round robin scheduling falls under the category of _____________ scheduling

a) Pre-emptive b) Non Pre-emptive c) Priority d) Shortest Job First

5. If the time quantum of round robin scheduling is very large then it is equivalent to

a) FCFS b) SJFS c) Longest Job First d) Longest Job Last

## 8. References

1. Silberschatz, P.B. Galvin and G. Gagne. *Operating System Concepts*. New Delhi: Wiley IndiaPrivate Ltd.. 8th Edition 2011.
2. Stalling William, *Operating Systems: Internals and Design Principle.* New Delhi : Prentice Hall India.7th Edition 2011.
3. Dietel, *Operating Systems*. New Delhi :Pearson Education.3$^{rd}$Edition 2007.
4. A.S. Tanenbaum. *Modern Operating Systems*. New Delhi : Prentice Hall India. 3rd Edition 2007

Reetha

**Verified by Subject Expert**
J. Anila Maily
Associate Professor of Computer Science

**Approved by HOD**
Sr. K. Reetha

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

# LESSON PLAN

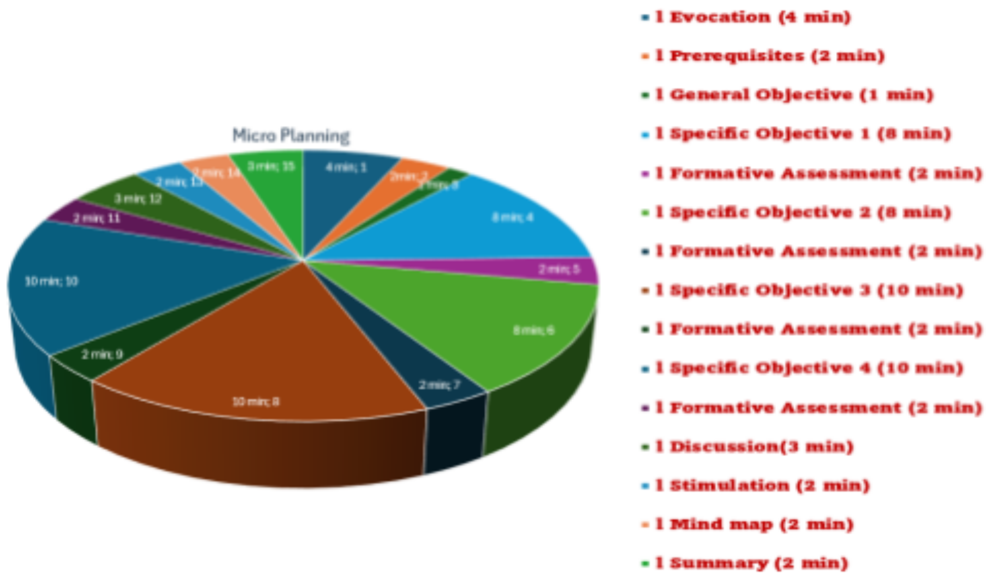
## Objective Oriented Learning Process RBT

| | |
|---|---|
| Programme | B.Sc. Computer Science |
| Semester | V |
| Subject title | Major Core: Python Programming |
| Code | 21UCSC52 |
| Hours | 4 |
| Total Hours | 60 |
| Credits | 4 |
| Max Marks | 100 |
| Unit & Title | Unit: III – Tuples |
| Name of the Faculty | M.Jothimani |
| T-L tools | Mind Maps, Lecture method, Visual aid: PPT, online references, and real-time execution |

### Prerequisite Knowledge

- Basic understanding of data types in Python (integers, floats, strings, lists).
- Knowledge of functions and their usage in Python.
- Understanding of lists and their operations in Python.

### Micro-Planning



## 1. Topic for Learning through Evocation:

designing a program to store student information, such as name, roll number, and marks. Instead of using a list, which allows modification, you want to ensure that once the details are stored, they cannot be changed. This introduces the concept of tuples in Python.

**Relating the concept to programming:**

- Tuples in Python are immutable sequences used to store heterogeneous data.
- They provide better performance compared to lists when immutability is required.

Discussion:

- Why would we need a data structure that cannot be modified?
- How do tuples differ from lists in Python?

---

## 2. Topic Introduction

**Definition:**
A tuple is a collection of ordered elements that are immutable (cannot be changed after creation).

**Characteristics of Tuples in Python:**

- Tuples are defined using parentheses ().
- They allow heterogeneous data types.
- Elements cannot be modified after assignment.
- Tuples support indexing and slicing.
- More memory-efficient than lists.

---

### 2.1. General Objective

Enable students to understand the concept of tuples and their importance in handling immutable data in Python.

### 2.2.Specific Objectives

- Define tuples and explain their key properties.
- Differentiate between tuples and lists.
- Develop Python programs utilizing tuples.
- Evaluate the performance benefits and immutability of tuples.

**First Phase :**

- SO1: Define tuples and their structure.
- SO2: Explain the key characteristics of tuples (e.g., immutability, ordering).
- SO3: Create Python code examples that demonstrate the use of tuples.
- SO4: Analyze how tuples behave in different programming scenarios.

**Second Phase :**

- SO1: Highlight the differences between tuples and lists.
- SO2: Demonstrate common operations that can be performed on tuples.
- SO3: Develop real-world Python programs that leverage tuples.
- SO4: Evaluate tuple-based programs about efficiency and immutability.

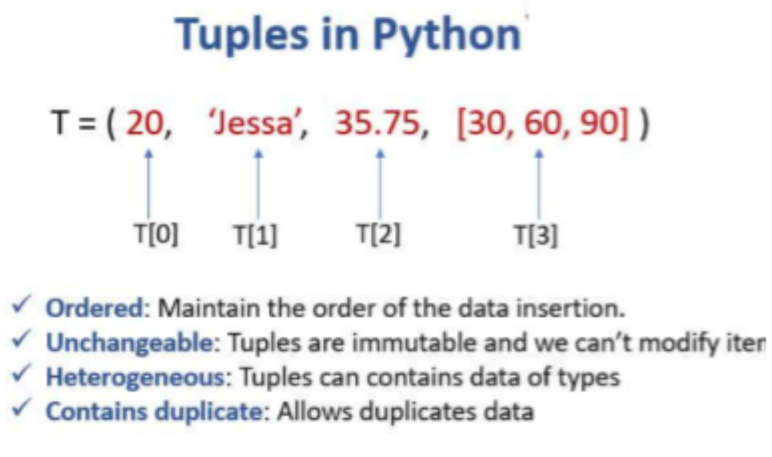
### 2.3. Taxonomy of Objectives

| Taxonomy of Objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | 1 | | | | | |
| B.Conceptual Knowledge | | 1,2 | | | | |
| C. Procedural Knowledge | | | 3,4 | 2 | | |
| D. Meta Cognitive Knowledge | | 2,3,4 | | | 4 | |

### 2.4. Keywords

Tuple, Immutability, Indexing, Slicing, Packing and Unpacking, Tuple Operations

### 2.5. Key Diagrams



Tuples in Python

T = ( 20, 'Jessa', 35.75, [30, 60, 90] )

T[0] T[1] T[2] T[3]

- ✓ Ordered: Maintain the order of the data insertion.
- ✓ Unchangeable: Tuples are immutable and we can't modify iter
- ✓ Heterogeneous: Tuples can contains data of types
- ✓ Contains duplicate: Allows duplicates data

### 3.Discussion:

**Code Examples**

**Creating a Tuple:**

```
# Defining a tuple
student = (101, "John Doe", 85.5)
print(student)
```

**Accessing Elements in a Tuple:**

```
print(student[0])  # Output: 101
print(student[1])  # Output: John Doe
```

**Tuple Packing and Unpacking:**

```
name, roll_no, marks = student
print(name)  # Output: John Doe
print(marks)  # Output: 85.5
```

**Tuple Immutability Demonstration:**

```
student[0] = 102  # TypeError: 'tuple' object does not support item assignment
```

**Tuple Slicing:**

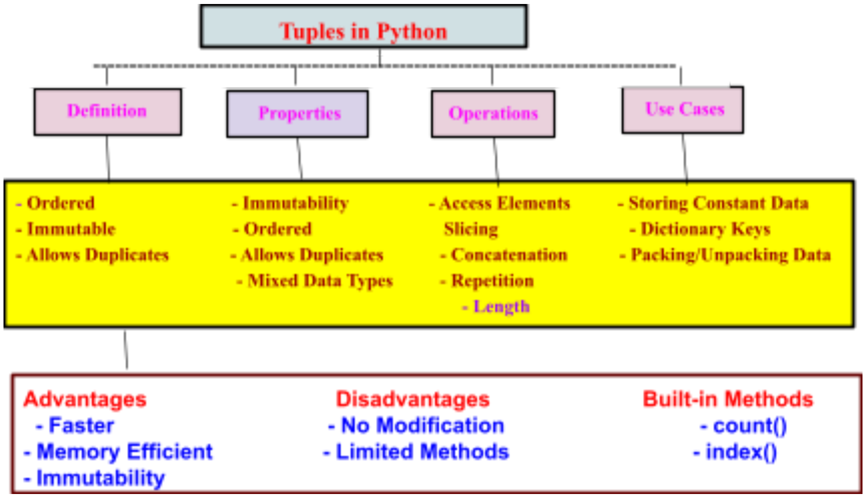
```
print(student[1:])  # Output: ('John Doe', 85.5)
```

**Tuple vs List Performance Comparison:**

```
import timeit
print(timeit.timeit("x = (1,2,3,4,5,6,7,8,9,10)", number=1000000))
print(timeit.timeit("x = [1,2,3,4,5,6,7,8,9,10]", number=1000000))
```

**PPT Link:**
[https://docs.google.com/presentation/d/1AkMXR9phfIH_R3vY3yMsdthdnUaiUA0Sq86wojfpEEs/edit?usp=sharing](https://docs.google.com/presentation/d/1AkMXR9phfIH_R3vY3yMsdthdnUaiUA0Sq86wojfpEEs/edit?usp=sharing)

## 4. Mind Map



## 5. Summary

- A tuple is an immutable, ordered collection of elements.
- Tuples are more memory-efficient than lists and useful for fixed data.
- Common operations include indexing, slicing, and unpacking.
- Tuples improve performance where immutability is required.
- They support indexing, slicing, and packing/unpacking.
- Useful for storing read-only data and for performance-sensitive tasks.
- Tuples can be used as dictionary keys (unlike lists).
- Tuples are a versatile tool that can be used when you need ordered, immutable collections.

## 6. Assessment through Stimulating Questions/Analogy/New Ideas and Concepts

### Formative Assessments (FA)

- Formative Assessment 1 (FA1) (2 minutes)
  Quiz on basic tuple properties.
- Formative Assessment 2 (FA2) (2 minutes)
  **:** Discuss differences between tuples and lists.

- Formative Assessment 3 (FA1) (2 minutes
  Explain the output of given tuple-related Python snippets.
  Execution of coding and debugging
- Formative Assessment 4 (FA4) (2 minutes):Implement a tuple-based program for storing product details.

**7. FAQs**

1. How is a tuple different from a list?
   - A tuple is immutable, whereas a list is mutable.
   - Tuples have better performance compared to lists for fixed data.
2. Can a tuple contain elements of different data types?
   - Yes, tuples support heterogeneous data storage.
3. What happens if I try to modify a tuple?
   - A TypeError is raised, indicating immutability.
4. Why use tuples instead of lists?
   - When data should remain unchanged for security or performance reasons.
5. Can a tuple be empty or contain only one element?
   - Yes, an empty tuple is defined as empty_tuple = ().
   - A single-element tuple requires a comma: single_tuple = (5,).
6. A tuple be empty or contain only one element?
   - Yes, an empty tuple is defined as empty_tuple = ().
   - A single-element tuple requires a comma: single_tuple = (5,).

**8. References**

1. Balagurusamy, E. (2020). *Programming in Python*. New Delhi, India: McGraw-Hill Education.
2. Kanetkar, Y. (2021). *Let Us Python* (3rd Edition). New Delhi, India: BPB Publications.
3. Malhotra, S., & Choudhary, S. (2018). *Python Programming* (2nd Edition). New Delhi, India: Oxford University Press.
4. Lutz, M. (2013). *Learning Python* (5th Edition). Sebastopol, CA: O'Reilly Media.

M. Jothimani

**Course Co Ordinator/s**

M.Jothimani
Associate Professor in computer Science

Reetha

**HOD**

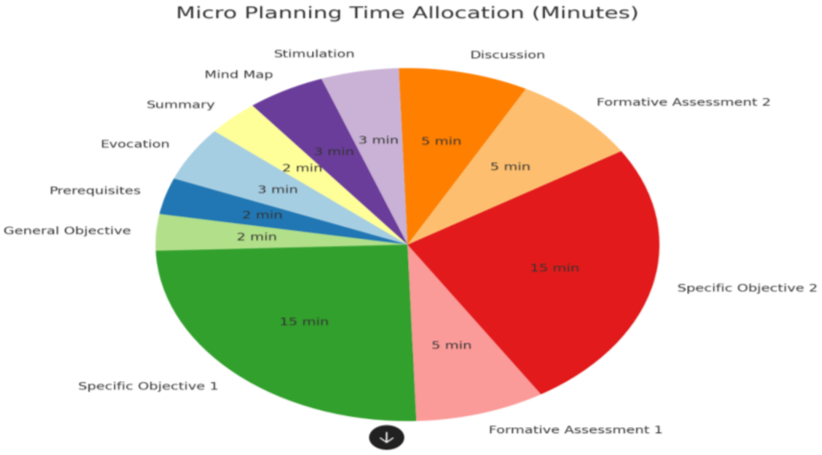
Sr.Reetha

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

## LESSON PLAN
## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | **V** |
| **Subject Title** | **Introduction to IoT** |
| **Code** | **21UCSE52** |
| **Hours** | **4** |
| **Total Hours** | **60** |
| **Credits** | **4** |
| **Max Marks** | **100** |
| **Unit & Title** | Unit:V – Application Building with IoT |
| **Name of the Faculty** | **Dr.J.Anila Maily** |
| **T-L tools** | Lecture method, **Visual aid**: Picture showing the applications buiding with IOT. |

### Micro -planning



| | |
|---|---|
| Evocation | 3min |
| Prerequisties | 2min |
| General Objective | 2min |
| specifici objective1 | 15min |
| Formative assessment1 | 5min |
| specifici objective2 | 15min |

| | |
|---|---|
| Formative assessment2 | 5min |
| Discussion | 5min |
| Stimulation | 3min |
| Mind map | 3min |
| Summary | 2min |

**1. Evocation (3 min)**

- **Objective:** Engage students and activate prior knowledge.
- **Activity:** Ask students to name IoT devices they have used (e.g., smart watches, Alexa, smart TVs).
- **Tool:** Live poll using Mentimeter or Kahoot.

**2. Prerequisites (2 min)**

- **Objective:** Ensure students understand basic IoT concepts.
- **Activity:** Quick Q&A on IoT fundamentals (e.g., What is IoT? How does it connect devices?).
- **Tool:** AI-powered quiz (Google Forms, Kahoot).

**3. General Objective (2 min)**

- **Objective:** Explain that students will explore real-world IoT applications across industries.
- **Activity:** Brief introduction to IoT benefits and challenges.

**4. Specific Objective 1 (15 min)**

- **Objective:** Learn about IoT applications in **Smart Homes, Healthcare, and Agriculture**.
- **Activity:** Discuss real-world case studies:
    - o **Smart Homes:** Alexa, Google Nest, IoT-based home automation.
    - o **Healthcare:** Wearable health trackers (e.g., Fitbit), remote patient monitoring.
    - o **Agriculture:** IoT-enabled smart irrigation systems.
- **Tool:** Video demonstration of IoT applications in action.

**5. Formative Assessment 1 (5 min)**

- **Objective:** Test understanding of IoT applications.
- **Activity:** Students match IoT applications to their industries in an interactive quiz.
- **Tool:** LMS quiz (Moodle, Google Forms).

**6. Specific Objective 2 (15 min)**

- **Objective:** Explore IoT applications in **Smart Cities, Industrial IoT, and Transportation**.
- **Activity:** Case study discussion:
  - **Smart Cities:** Smart streetlights, connected waste management.
  - **Industrial IoT (IIoT):** Predictive maintenance in factories.
  - **Transportation:** IoT-enabled traffic monitoring, connected vehicles.
- **Tool:** Live demo of an IoT dashboard (e.g., real-time air quality monitoring).

### 7. Formative Assessment 2 (5 min)

- **Objective:** Apply IoT knowledge to real-world problem-solving.
- **Activity:** Students analyze an IoT scenario and propose a solution.
- **Example:** How can IoT reduce traffic congestion in a city?

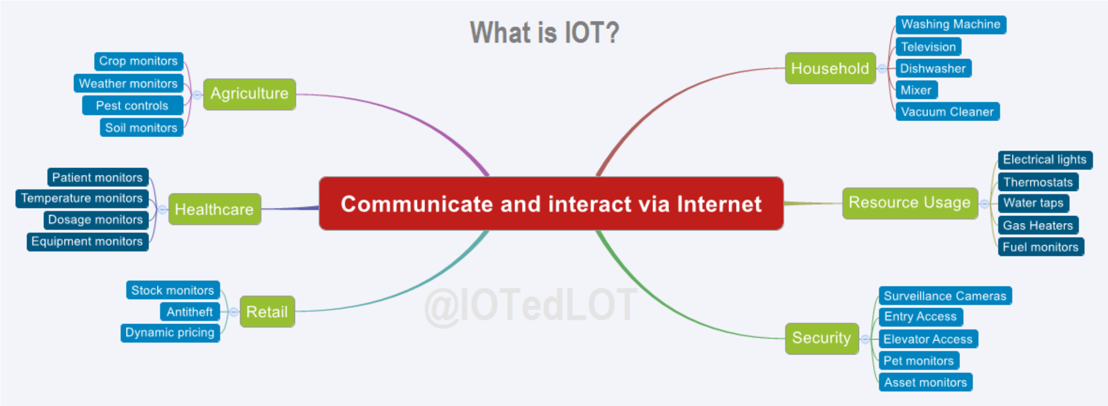
### 8. Discussion (5 min)

- **Objective:** Encourage critical thinking on IoT challenges.
- **Activity:** Debate: "What are the biggest security risks of IoT?"

### 9. Stimulation (3 min)

- **Objective:** Spark curiosity about the future of IoT.
- **Activity:** Watch a video on **next-gen IoT trends (AI-powered IoT, 6G, Edge Computing).**

### 10. Mind Map (3 min)



- **Objective:** Organize IoT knowledge visually.
- **Activity:** Students collaboratively create a **mind map** of IoT applications and their use cases.
- **Tool:** Online tools like MindMeister or Miro.

### 11. Summary (2 min)

- **Objective:** Reinforce key takeaways.
- **Activity:** AI-generated summary + students share their biggest takeaway from the session.

## 12.FAQ's:

**1. Which of the following is an example of an IoT application in healthcare?**
A. Smart thermostats controlling room temperature
B. Wearable devices that monitor heart rate and send data to doctors
C. GPS systems used for navigation
D. Online banking apps

**2. How does IoT contribute to smart agriculture?**
A. By automating crop planting without human involvement
B. By providing real-time weather forecasts
C. By using sensors to monitor soil moisture and optimize irrigation
D. By reducing the need for farmworkers entirely

**3. Which IoT application is commonly used in smart cities to manage traffic?**
A. Virtual Reality (VR) systems
B. Intelligent traffic lights that adjust to real-time conditions
C. High-speed internet in public spaces
D. Automated toll collection systems only

**4. In which sector is IoT used for predictive maintenance?**
A. Retail
B. Manufacturing
C. Education
D. Tourism

**5. What is the primary role of IoT in home automation?**
A. Providing faster internet speeds
B. Enabling devices like lights, thermostats, and appliances to be controlled remotely
C. Increasing the resale value of homes
D. Enhancing home security through traditional locks and keys

## 13.References: (Books/Periodicals/Journals)

1. ArshdeepBahga and Vijay Madisetti.*Internet of Things- A Hands-on Approach*. India: Universities Press Private Limited. 2015
2. Hanes, David, Gonzalo Salgueiro, Patrick Grossetete, Robert Barton and Jerome Henry. *IoT fundamentals: Networking technologies, protocols, and use cases for the Internet of Things*. Cisco Press. 2017.
3. Qusay F. Hassan.*Internet of Things A to Z: Technologies and Applications*. Wiley Publication IEEE Press. 2018.

## 14.1.1Taxonomy of objectives:

| Taxonomy of Objectives |
|---|

| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
|---|---|---|---|---|---|---|
| | **Remember** | **Understand** | **Apply** | **Analyze** | **Evaluate** | **Create** |
| A. Factual Knowledge | **1** | | | **1,2** | **1** | |
| B. Conceptual Knowledge | | **1, 2** | **1,2** | | **1** | |
| C. Procedural Knowledge | | | | | | |

**14.2 Key words:**

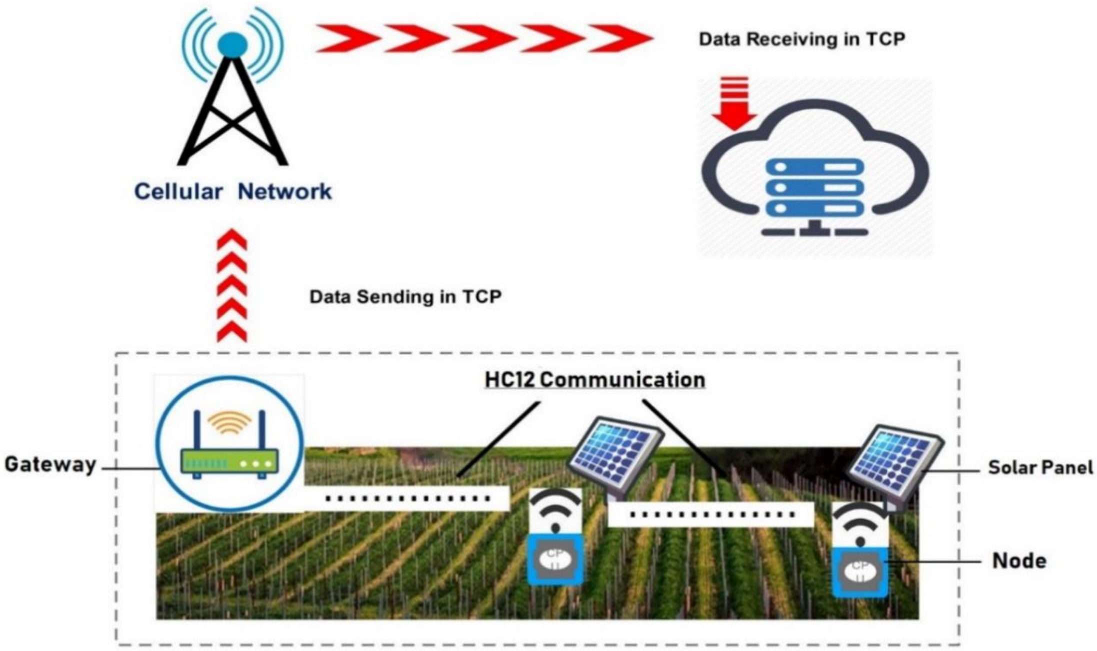
sensor", "connectivity", "firmware", "access control", "wearable", "GNSS", "radio frequency identification (RFID)", "cloud computing", "repeater", "health monitoring sensors", "flood detection sensors", and "API" (Application Programming Interface"

**14.3 Key diagrams (if any):**

# Applications of IOT

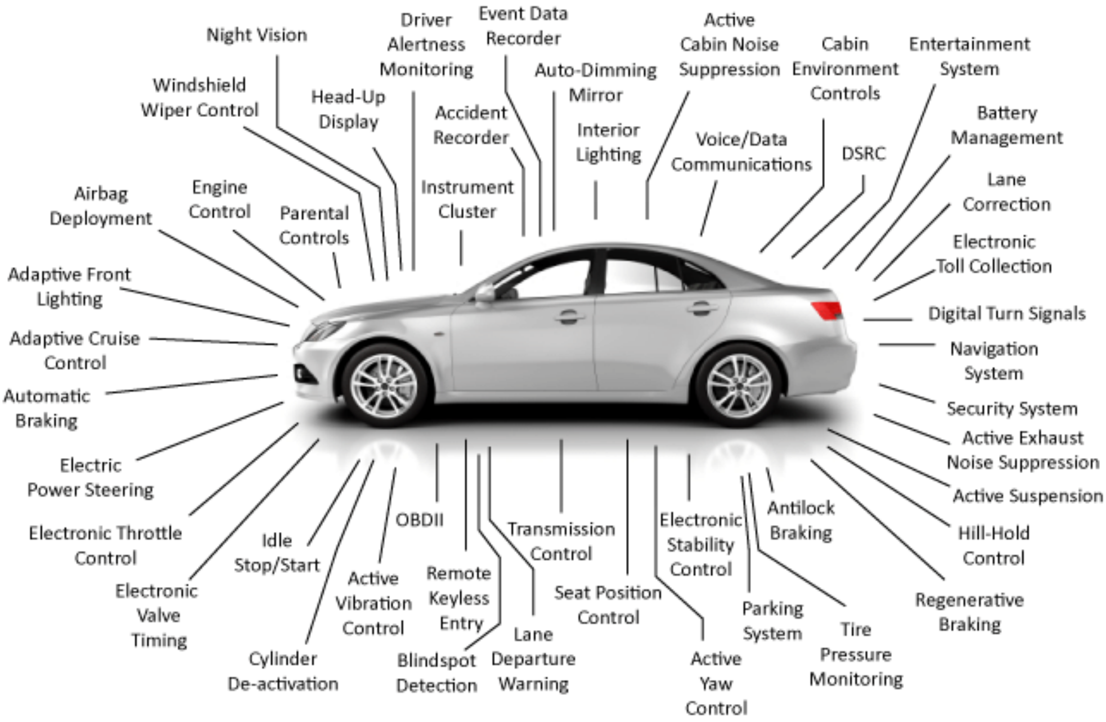
### 1. Smart Agriculture

Food is an integral part of life without which we cannot survive. However, it is an unfortunate fact that a lot of food is wasted in developed countries like America while people starve in poorer countries like Chad, Sudan, etc. One way to feed everyone is through better agricultural practices which can be enhanced using IoT applications. This can be done by first collecting data for a farm such as soil quality, sunlight levels, seed type, and rainfall density from various sources like farm sensors, satellites, local weather stations, etc. and then using this data with Machine Learning and IoT to create custom recommendations for each farm that will optimize the planting procedure, irrigation levels required, fertilizer amount, etc. All this will result in better yield or crops with a focus on reducing world hunger in the future. This is done very efficiently by SunCulture, a top IoT application, which is an initiative by Microsoft AI for Earth.



### 2. Smart Vehicles

Smart vehicles or self-driving cars are Iot applications as they can be called are pretty dependent on IoT. These cars have a lot of features that are integrated with each other and need to communicate such as the sensors that handle navigation, various antennas, controls for speeding or slowing down, etc. Here the Internet of Things technology is critical, especially in the sense that self-driving cars need to be extremely accurate and all the parts need to communicate with each other in milliseconds on the road. Tesla Cars are quite popular and working on their self-driving cars. Tesla Motors' cars use the latest advancements in Artificial Intelligence and the Internet of Things. And they are quite popular as well!!! Tesla Model 3 was the most sold plug-in electric car in the U.S. in 2018 with a total yearly sales of around 140,000 cars. This top IoT application has gained a lot of advancement in recent years



### 3. Smart Home

Maybe one of the most famous applications of IoT is in Smart Homes. After all, who hasn't heard about connecting all the home applications like lighting, air conditioners, locks, thermostat, etc. into a single system that can be controlled from your smartphone? These IoT devices are applications of IoT and becoming more and more popular these days because they allow you complete freedom to personalize your home as you want. In fact, these IoT devices are so popular that every second there are 127 new devices connected to the internet. Some popular ones that you might have heard have, or even have in your home, include Google Home, Amazon Echo Plus, Philips Hue Lighting System, etc. There are also all sorts of other inventions that you can install in your home including Nest Smoke Alarm and Thermostat, Foobot Air Quality Monitor, August Smart Lock, etc. These applications of IoT are getting famous nowadays.

HOME SMART HOME
SECURITY CAMERAS
BLINDS
THERMOSTAT
LIGHTING
TV/SPEAKERS
APPLIANCES
LOCKS
IRRIGATION

## 15.Drive Link

https://docs.google.com/presentation/d/1yW5ogQs56kS3JzbORyLWG8VQE8O1-1ne/edit?usp=sharing&ouid=104648840415631957301&rtpof=true&sd=true

Reetha

**Course In-charge**

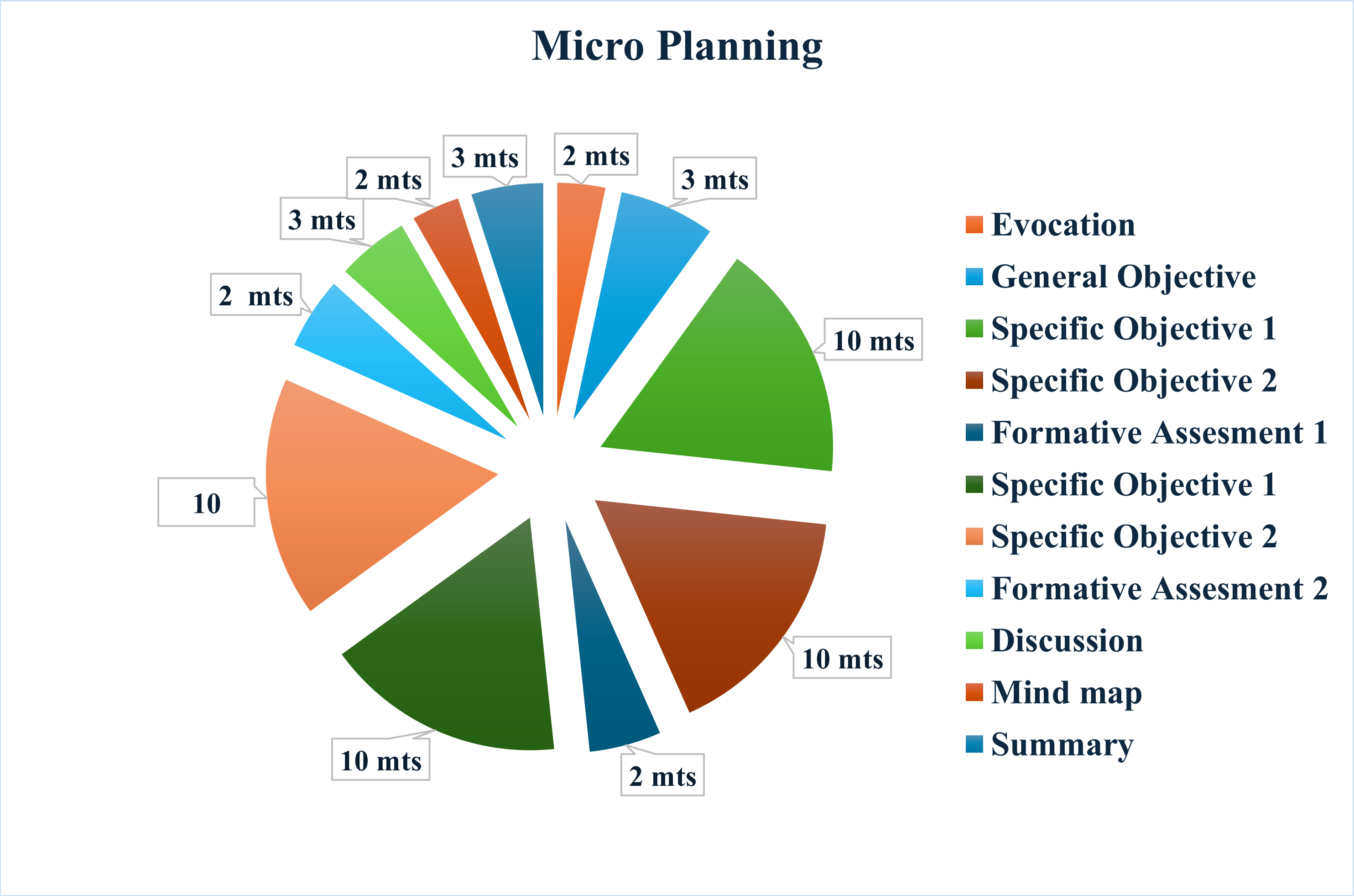
**Approved by HoD.**

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

# LESSON PLAN

## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | VI |
| **Course Title** | Core : .NET Programming |
| **Course Code** | 21UCSC61 |
| **Hours** | 5 |
| **Total Hours** | 75 |
| **Credits** | 4 |
| **Max Marks** | 100 |
| **Unit & Title** | Unit III : Validation Controls |
| **Name of the Faculty** | Ms. J. Anila Maily |
| **T-L tools** | Lecture method, **Visual aid**: PPT |



1. **Learning through evocation**

Students are asked to identify the errors that may happen while submitting forms.

2. **Topic Introduction:**

Students are introduced to the concept of validation.

**2.1 General Objective:**

Enables the students to implement validation of forms submitted by users.

**2.2 Specific Objectives:**

Enables the students to:

1. To understand client side and server side validation.
2. To apply various validation techniques.

First Phase:

SO1 : Why validation is essential.

SO2 : Types of validation controls.

Second Phase:

SO1 : Apply validation controls to ensure error free form submission.

SO2 : Apply custom validation control.

**2.3 : Taxonomy of objectives:**

| **Taxonomy of Objectives** | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyse** | **Evaluate** | **Create** |
| A. Factual Knowledge | 1,2 | | | | | |
| B. Conceptual Knowledge | | **1,2** | **1,2** | | | |
| C. Procedural Knowledge | | | 1,2 | | | |
| D. Meta Cognitive Knowledge | | | | 1,2 | | 2 |

**2.4 : Key words:**

Required Field validator, Compare validator, Range validator, Custom Validator

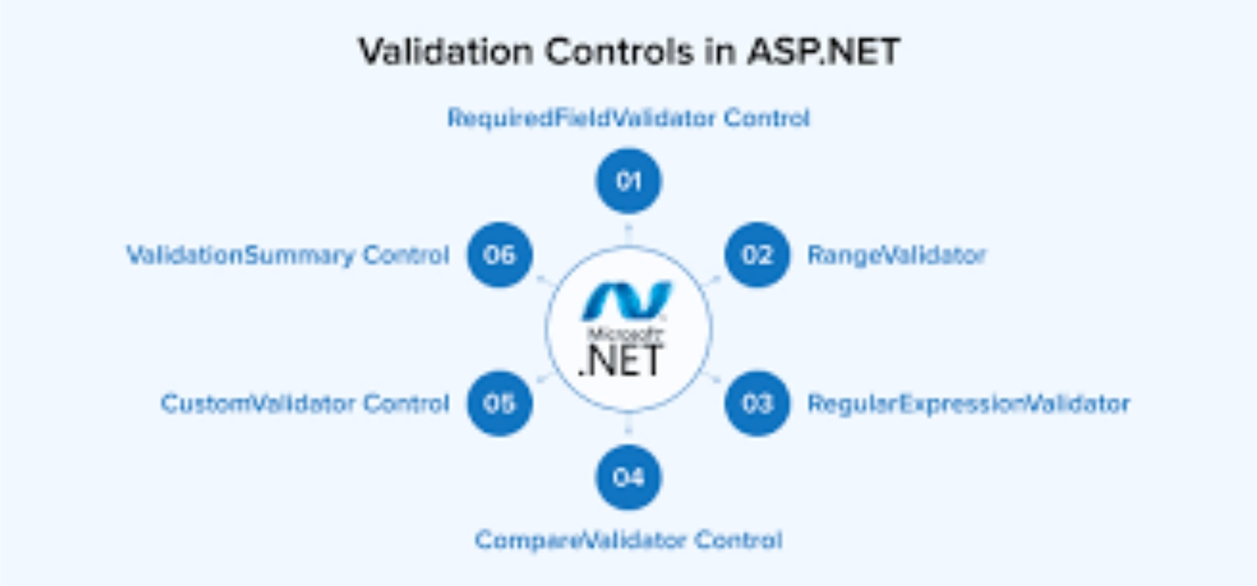
**2.5 : Key diagrams :**

**PowerPoint Presentation:** DOTNET.pptx

3. **Discussion:**

The students will be asked to identify the possible errors that may occur during a form submission and how to use the various validation controls to overcome them.

**4. Mind Map:**


Validation Controls in ASP.NET
RequiredFieldValidator Control
01
06
ValidationSummary Control
02
RangeValidator
Microsoft .NET
05
CustomValidator Control
03
RegularExpressionValidator
04
CompareValidator Control

**5. Summary:**

Students will be asked to use create their custom validation.

**6. Assessment**

**Formative assessment1 :** Discuss about the various validation control and why it is essential.

**Formative assessment2 :** Apply custom validation and group validation controls.

**7. FAQ's:**

| |
|---|
| 1. ASP.NET validation controls works |
| a) Only on client side b) Only on server side c)both on client and server side d) only on text |
| 2. Mandatory property of all validation controls |
| a) Message b) client script c) server script d) control to validate |
| 3. Which validation is used to ensure user enters a number from 20 – 120? |

| |
|---|
| a) RequiredField b) Compare. c) Range d) validationSummary |
| 4. Type of validation used to check password and confirm password in a login form. |
| a) RequiredField b) Compare. c) Range d) Regular expression |
| 5. Validation used to check the email entered by the user follows the email pattern |
| a) RequiredField b) Compare. c) Range d) Regular expression |

## 8. References

1. Christian Nagel, Bill Evjen, Jay Glynn, Karli Watson, Morgan Skinner. *Professional C# 2012 and .NET 4.5*. New Delhi: Wiley India Private Ltd. First Edition 2012Harsh Bhasin. *Programming in C#*, New Delhi: Oxford University Press. First Edition 2014.
2. MridulaParihar, YeshSingal and Nitin Pandey. *“Visual Studio .Net Programming*”. New Delhi: Prentice Hall India. First Edition 2002
3. Black Book. Kogent Learning Solutions Inc,*NET 4.0 Programming (6-in-1)*.New Delhi : Dream Tech Press.
4. Paul Deitel and Harvey Deitel, *C# 2010 for Programmers*, New Delhi: Pearson Education 4th Edition
5. G. Andrew Duthie. *Microsoft ASP.NET Step by step*. Microsoft Press, 2003

Reetha

**Verified by Subject Expert**
J. Anila Maily
Associate Professor of Computer Science

**Approved by HOD**
Sr. K. Reetha

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.

**Lesson Plan: Introduction to the Agile Model in Software Engineering**

<u>LESSON PLAN</u>

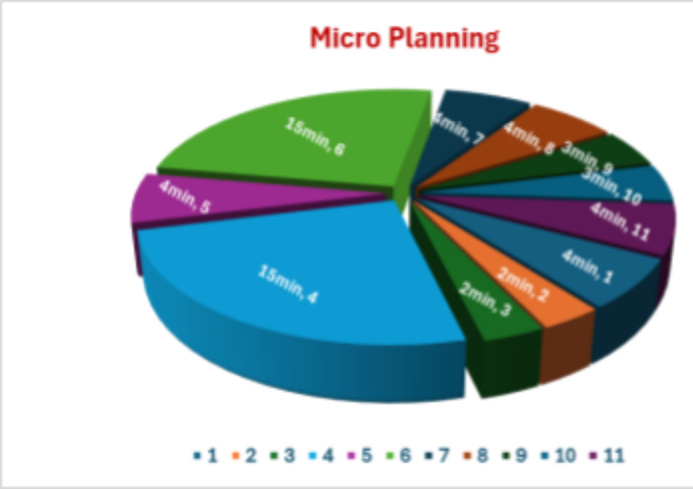
<u>Objective Oriented Learning Process RBT</u>

| | |
|---|---|
| Programme | B.Sc. Computer Science |
| Semester | VI |
| Subject title | Major Core: Software Engineering |
| Code | 21UCSC62 |
| Hours | 4 |
| Total Hours | 60 |
| Credits | 4 |
| Max Marks | 100 |
| Unit & Title | Unit: I – Agile Model |
| Name of the Faculty | M.Jothimani |
| T-L tools | Mind Maps, Lecture method, Visual aid: PPT , group discussion |

## 1. Prerequisite Knowledge:

- Basic understanding of software development life cycles (SDLC).
- Knowledge of iterative development and project management principles.
- Basic familiarity with team-based project work and collaboration.

Micro Planning:



1. Evocation (4 min)
2. Prerequisites (2 min)
3. General Objective (1 min)
4. Specific Objective 1 ( 15min)
5. Formative Assessment (4 min)
6. Specific Objective 2 (15min)
7. Formative Assessment (4min)
12. Discussion(3 min)
13. Stimulation (2 min)
14. Mind map (2 min)
15. Summary (2 min)

### 1. Topic for Learning through Evocation:

Designing a project to develop software (e.g., a student management system) using Agile methods. Instead of following a rigid waterfall approach, the development process will be broken into iterations or "sprints" for frequent feedback and improvements.

**Relating the concept to programming:**

- **The agile model** emphasizes flexibility, iterative development, and constant collaboration with the client.
- It promotes an adaptive approach to development, where the software evolves based on feedback from the client and team.

**Discussion:**

- Why do we need a flexible and adaptive approach to software development?
- How does Agile differ from traditional Waterfall models in handling changes and feedback?

### 2. Topic Introduction

**Definition:** The Agile model is a software development methodology that focuses on iterative development, customer collaboration, and flexibility in responding to changes throughout the development process.

**Characteristics of the Agile Model:**

- Emphasizes small, iterative releases (sprints).
- Close collaboration with stakeholders and clients.
- Focus on delivering functional software frequently.
- Embraces changes in requirements, even late in development.
- Teams are self-organizing and cross-functional.

### 2.1. General Objective

Enable students to understand the Agile model and how it can be applied to software engineering projects, focusing on flexibility, collaboration, and iterative development.

### 2.2. Specific Objectives

- Define the Agile model and explain its core principles.
- Demonstrate how Agile can be applied to real-world projects using Scrum or Kanban.

**First Phase:**

**SO1** - Define the Agile model
**SO2** - Explain the core principles of Agile

**Second Phase:**

**SO2** - Demonstrate how Agile can be applied to real-world projects using Kanban

**SO3** - Implement Agile practices (e.g., sprints, stand-ups) in a simulated project

### 2.3. Taxonomy of Objectives

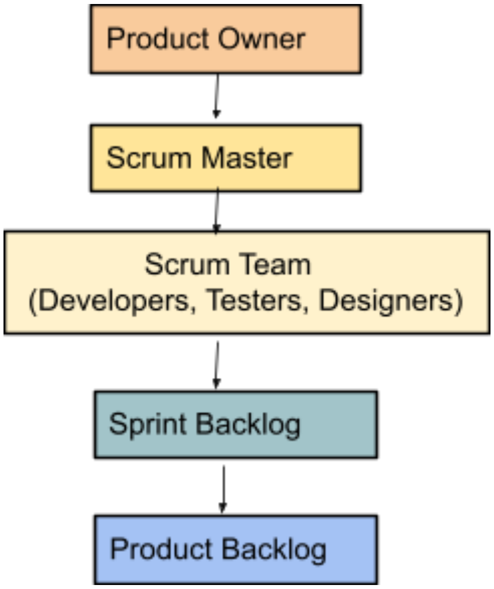
| Taxonomy of Objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | 1 | | | | | |
| B.Conceptual Knowledge | | 2 | | | | |
| C. Procedural Knowledge | | | 3 | | | |
| D. Meta Cognitive Knowledge | | | | 3 | | |

### 2.4. Keywords

- Agile, Scrum, Sprint, Product Backlog, Iterative Development, Scrum Master, Product Owner, Kanban, Continuous Integration, Retrospectives, Standups

### 2.5. Key Diagrams

The Scrum framework is one of the most common Agile methodologies. This diagram will help explain the roles, ceremonies, and artifacts in Scrum.



[**Plan**] ⟷ [**Develop**] ⟷ [Test] ⟷ [**Feedback**]

↑ ↓

[**Iterate**] ⟷ [**Refactor**] ⟷ [**Release**]

**Explanation**:

- Agile emphasizes the **feedback loop**, where teams develop features, test them, gather feedback, and make necessary improvements before the next cycle.
- This allows the software to evolve based on stakeholder feedback and changing requirements.

### 3. Discussion

The Agile model is a popular and flexible approach to software development that emphasizes iterative progress, collaboration, and the ability to adapt to changing requirements. Unlike traditional software development models (like Waterfall), Agile focuses on delivering small, incremental improvements over time rather than a large, final product after a long development period. This approach helps teams deliver software more frequently, with better alignment to user needs and market changes.

Core Features of Agile:

- Iterative Process: The project is divided into small, manageable units called iterations or sprints, each lasting a few weeks. At the end of each sprint, a working increment of the product is delivered.
- Collaboration: Close collaboration between the development team and stakeholders, such as customers and business representatives, is central to Agile.
- Flexibility: Agile allows for changes and new requirements to be introduced even late in the development process, adapting to evolving market needs or user feedback.
- Continuous Improvement: Teams regularly reflect on their performance and make adjustments to improve their processes in future sprints.

Some popular Agile frameworks include:

1. Scrum:
    - Scrum is a well-known Agile framework that organizes work into sprints (typically 1-4 weeks).
    - Teams follow specific roles: Product Owner, Scrum Master, and Development Team.

    - Scrum includes regular ceremonies like Sprint Planning, Daily Standups, Sprint Review, and Sprint Retrospectives to ensure communication, transparency, and continuous improvement.

2. Kanban:
    - Kanban focuses on visualizing the workflow and limiting work in progress (WIP).
    - It uses a Kanban board to manage tasks through various stages (To-Do, In Progress, Done), making it easy to identify bottlenecks and ensure smooth flow.
3. Extreme Programming (XP):
    - XP emphasizes technical excellence and practices such as pair programming, test-driven development (TDD), and continuous integration.
    - It encourages close collaboration between developers and end-users, and aims to deliver high-quality code frequently.
4. Lean Software Development:
    - Lean focuses on reducing waste and increasing efficiency. It encourages teams to build the minimum viable product (MVP) and iterate quickly.

## Key Practices in Agile

- Sprint Planning: At the beginning of each sprint, the team plans what features or tasks will be completed during that sprint.
- Daily Standups: A short, daily meeting where team members discuss what they worked on yesterday, what they will work on today, and any obstacles they face.
- Sprint Review: At the end of each sprint, the team demonstrates the working product to stakeholders and gathers feedback.
- Sprint Retrospective: The team reflects on the sprint, discussing what went well and what can be improved for the next sprint.

## Benefits of the Agile Model

1. Faster Time to Market: Agile allows teams to release product increments regulrly, enabling faster delivery of usable software.
2. Flexibility: Requirements can evolve as the project progresses, and teams can pivot when necessary.
3. Increased Customer Satisfaction: Since Agile prioritizes customer feedback, the product better aligns with user needs and expectations.
4. Improved Quality: Continuous testing and integration help detect and fix issues early in the development process.
5. Better Collaboration: Agile promotes open communication between team members and stakeholders, improving overall teamwork.

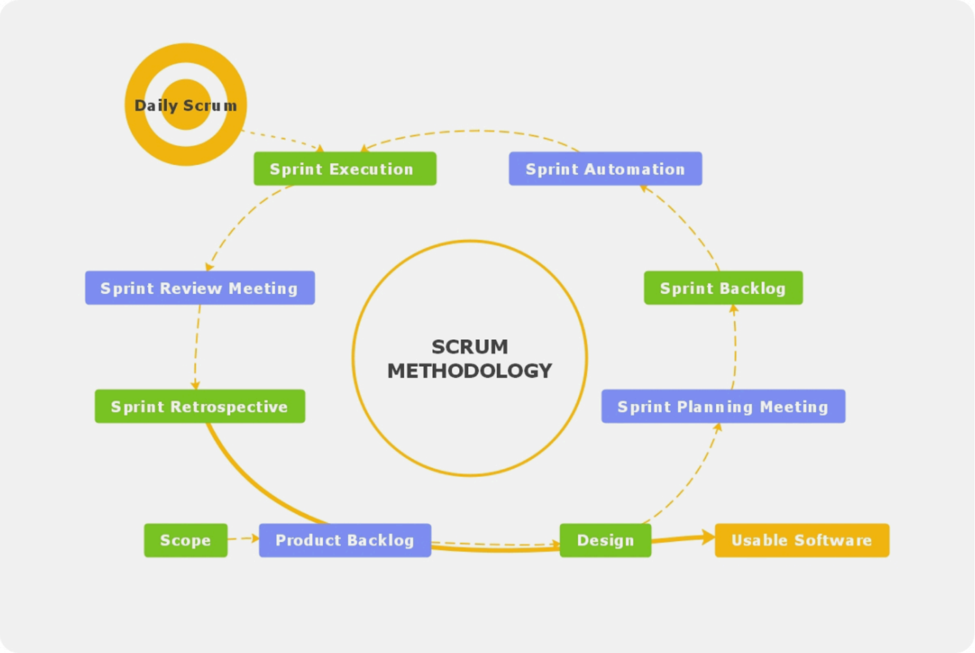
Challenges of the Agile Model

1. Scope Creep: Due to the flexibility of Agile, there is a risk of continuously changing requirements, leading to scope creep.
2. Requires Strong Team Collaboration: Agile depends heavily on communication and collaboration, which may be challenging for some teams or organizations.
3. Resource Intensity: Agile requires consistent effort, commitment, and involvement from all team members, including stakeholders.
4. Difficulty in Scaling: Implementing Agile at scale in large, complex organizations can be challenging, especially with cross-team collaboration.

**PPT Link:**

https://docs.google.com/presentation/d/1apPiN2CyJKKP7HETovr3aCSuQXnM1P-ZQ8-kFN7bQ1U/edit?usp=sharing

**4.Mind Map**



**5. Summary**

- **Agile is an iterative and flexible methodology** that encourages constant feedback and collaboration.
- Agile frameworks like **Scrum** and **Kanban** support teamwork, adaptability, and continuous improvement.
- **Frequent deliveries** of working software help clients and stakeholders adjust their requirements as the project progresses.
- The Agile model has revolutionized software development by promoting flexibility, collaboration, and continuous delivery. Its iterative nature allows teams to respond quickly to changing requirements and deliver high-quality products efficiently.

However, Agile does require a cultural shift towards transparency, frequent communication, and adaptability. It is best suited for projects with evolving requirements or where continuous improvement is needed.

- To fully embrace Agile, organizations must ensure they have the right mindset, processes, and tools in place to support Agile practices. Teams should focus on delivering value to the customer, while also remaining flexible and improving their workflow over time.

**6. Assessment through Stimulating Questions/Analogy/New Ideas and Concepts**

**Formative Assessments (FA):**

- **FA1 :**

  **Quick Quiz and Discussion:**How does Agile encourage better collaboration between developers and clients?on Agile principles and the differences between Agile and Waterfall models.
- **FA2:**

  **Activity:** Create a simple Scrum board for a hypothetical software project (e.g., a to-do list app)..**Real-World Application:** Given a project scenario, design an Agile process (Scrum or Kanban) and explain how iterative feedback can be used.

  **Group Activity:** Let students form small groups and simulate a short sprint (e.g., 10 minutes) to complete a simple task using Agile principles, then have them present their experiences.

**7. FAQs**

- What is the difference between Agile and Waterfall?
  - Agile is flexible, iterative, and encourages continuous feedback, whereas Waterfall follows a linear, sequential approach with little room for changes after the project begins.
- Can Agile work for all types of projects?
  - Agile is ideal for projects where requirements can change or evolve over time. However, it may not be suitable for projects with fixed requirements from the start, like regulatory software.
- What are the roles in Scrum?
  - Scrum Master: Facilitates the Scrum process.
  - Product Owner: Represents the customer and manages the product backlog.
  - Development Team: Builds the product incrementally in sprints.
- How do Agile teams measure progress?
  - Burn-down charts, sprint reviews, and retrospectives are common ways to track progress and ensure continuous improvement.

## 8. References

- **Pressman, R. S. (2014).** *Software Engineering: A Practitioner's Approach* (10th ed.). McGraw-Hill Education.
- **Schwaber, K., & Sutherland, J. (2017).** *The Scrum Guide: The Definitive Guide to Scrum: The Rules of the Game.* Scrum.org.
- **Highsmith, J. (2010).** *Agile Project Management: Creating Innovative Products (2nd Edition).* Addison-Wesley Professional.

M. Jothimani

**Course Co Ordinator/s**

M.Jothimani
Associate Professor in computer Science

Reetha

**HOD**

Sr.Reetha

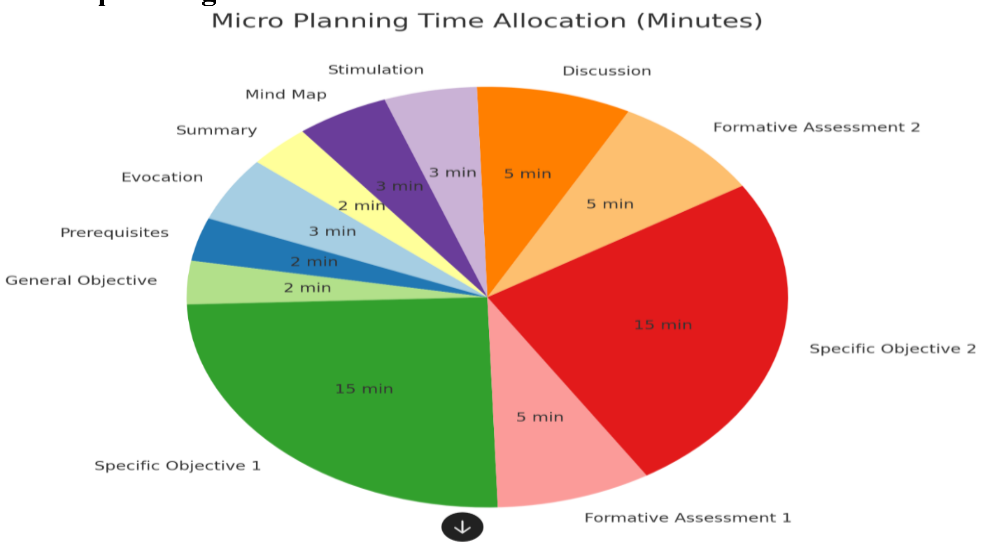
*Head*
*Department of Computer Science*
*St. Mary's College (Autonomous)*
*Thoothukudi.*

# LESSON PLAN

## Objective Oriented Learning Process RBT

| | |
|---|---|
| **Programme** | B.Sc. Computer Science |
| **Semester** | **VI** |
| **Subject Title** | **Computer Networks** |
| **Code** | **21UCSC63** |
| **Hours** | **5** |
| **Total Hours** | **75** |
| **Credits** | **4** |
| **Max Marks** | **100** |
| **Unit & Title** | Unit: 1 – Different types of Topology |
| **Name of the Faculty** | **Sr.K.Reetha** |
| **T-L tools** | Lecture method, **Visual aid**: Picture showing the types of Network Topology. |

## Micro -planning



| | |
|---|---|
| Evocation | 3min |
| Prerequisties | 2min |
| General Objective | 2min |
| specifici objective1 | 15min |
| Formative assessment1 | 5min |
| specifici objective2 | 15min |

| | |
|---|---|
| Formative assessment2 | 5min |
| Discussion | 5min |
| Stimulation | 3min |
| Mind map | 3min |
| Summary | 2min |

**1. Evocation (3 min)**

- **Objective:** Activate prior knowledge and engage students.
- **Activity:** Ask students where they have seen networks in daily life (e.g., Wi-Fi at home, office LAN, mobile networks).
- **Tool:** Quick word cloud generation from student responses using Mentimeter.

**2. Prerequisites (2 min)**

- **Objective:** Ensure students have basic knowledge of networks.
- **Activity:** Quick Q&A on what a network is and why topology matters.
- **Tool:** AI-powered quiz (e.g., Kahoot).

**3. General Objective (2 min)**

- **Objective:** Explain that students will learn about different network topologies and their advantages/disadvantages.
- **Activity:** Provide a one-sentence summary of what topology means in networking.

**4. Specific Objective 1 (15 min)**

- **Objective:** Understand common network topologies.
- **Activity:** Explain and compare the following topologies using diagrams and real-life examples:
    - **Bus Topology** (e.g., legacy Ethernet networks)
    - **Star Topology** (e.g., modern office LANs)
    - **Ring Topology** (e.g., older fiber optic networks)
    - **Mesh Topology** (e.g., military and high-redundancy networks)
- **Tool:** Network simulation software (e.g., Cisco Packet Tracer).

**5. Formative Assessment 1 (5 min)**

- **Objective:** Test understanding of topologies.
- **Activity:** Drag-and-drop matching activity (students match topology names with their diagrams).
- **Tool:** Google Forms or an LMS quiz.

**6. Specific Objective 2 (15 min)**

- **Objective:** Explore advanced and hybrid topologies.

- **Activity:** Explain:
    - **Tree Topology** (used in hierarchical networking)
    - **Hybrid Topology** (combining multiple topologies for scalability)
    - **Wireless Topology** (mesh networks in smart cities)
- **Tool:** Live demo using an interactive network builder tool.

### 7. Formative Assessment 2 (5 min)

- **Objective:** Apply knowledge to real-world scenarios.
- **Activity:** Students analyze a given network diagram and identify the topology used.

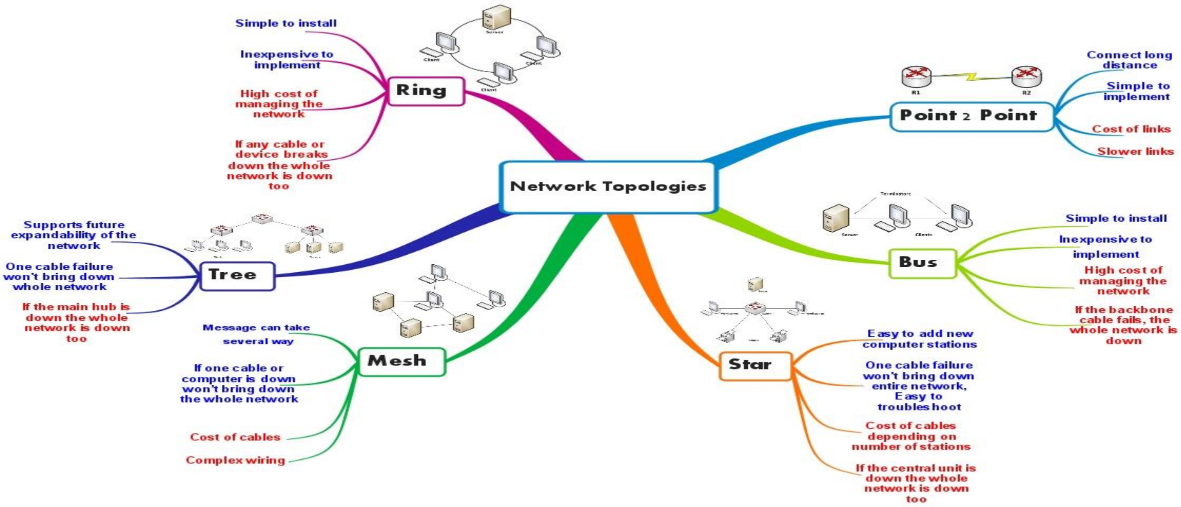
### 8. Discussion (5 min)

- **Objective:** Encourage critical thinking about network design.
- **Activity:** Debate: "Which topology is best for a smart home, a bank, and a data center?"

### 9. Stimulation (3 min)

- **Objective:** Spark curiosity about future network advancements.
- **Activity:** Show a short video on **next-gen network topologies (5G, IoT, Quantum Networks).**

### 10. Mind Map (3 min)



Network Topologies
Ring
Simple to install
Inexpensive to implement
High cost of managing the network
If any cable or device breaks down the whole network is down too
Point 2 Point
R1
R2
Connect long distance
Simple to implement
Cost of links
Slower links
Bus
Simple to install
Inexpensive to implement
High cost of managing the network
If the backbone cable fails, the whole network is down
Star
Easy to add new computer stations
One cable failure won't bring down entire network, Easy to troubleshoot
Cost of cables depending on number of stations
If the central unit is down the whole network is down too
Mesh
Message can take several way
If one cable or computer is down won't bring down the whole network
Cost of cables
Complex wiring
Tree
Supports future expandability of the network
One cable failure won't bring down whole network
If the main hub is down the whole network is down too

- **Objective:** Organize learning visually.
- **Activity:** Students collaboratively create a mind map of network topologies and their use cases.
- **Tool:** MindMeister or Miro board.

### 11. Summary (2 min)

- **Objective:** Reinforce key takeaways.
- **Activity:** AI-generated summary of key points and a takeaway quiz.

**12.FAQ's:**

1. Physical or logical arrangement of network is __________
   a) Topology b) Routing c) Networking d) Control
2. Which network topology requires a central controller or hub?
   a) Star b) Mesh c) Ringd) Bus
3. _______ topology requires a multipoint connection.
   a) Star b) Mesh c) Ringd) Bus
4. Data communication system spanning states, countries, or the whole world is
   a) LAN b) WAN c) MAN d) PAN
5. Data communication system within a building or campus is________
   a) LAN b) WAN c) MAN d) PAN

**13.References: (Books/Periodicals/Journals)**

1. Andrew S. Tanenbaum David J. Wetherall. *Computer Networks* .New Delhi: Pearson. 5th Edition 2010.
2. Stallings W. *Data and Computer Communications*. New Delhi: Prentice Hall. Ninth Edition 2010.
3. Peterson. L.L. and Davie. S.B. *Computer Networks*. San Fransisco: Morgan Kaufmann Publishers. Fifth Edition 2011.

**14.1Taxonomy of objectives:**

| Taxonomy of Objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | **The Cognitive Process Dimension** | | | | | |
| | **Remember** | **Understand** | **Apply** | **Analyze** | **Evaluate** | **Create** |
| A. Factual Knowledge | **1** | | | **1,2** | **1** | |
| B. Conceptual Knowledge | | **1, 2** | **1,2** | | | |
| C. Procedural Knowledge | | | | **1,2** | | |

**14.1Key words:**

bus, star, ring, mesh, tree, hybrid, node, central hub, point-to-point, link, cable, data flow, redundancy, fault tolerance, MAC address, access point, switch, router

**14.2Key diagrams (if any):**

**Network Topology**

Network topology is the structure of the connections between computers and other network components. There are numerous physical network topologies from which an organization can choose based on network size, suitability, and business objectives. Each network topology type comprises of various node and link configurations and has its own benefits and drawbacks. Below is a list of the six different types of physical network topologies:

1. Star Topology
2. Bus Topology
3. Ring Topology
4. Mesh Topology
5. Tree Topology
6. Hybrid Topology

**15.Drive Link:**

https://docs.google.com/presentation/d/1Y0BrivpuUKRkx6o9n-m5pzPBhCbkL6Nt/edit?usp=sharing&ouid=104648840415631957301&rtpof=true&sd=true

Reetha

**Course In-charge**

Reetha

**Approved by HoD.**

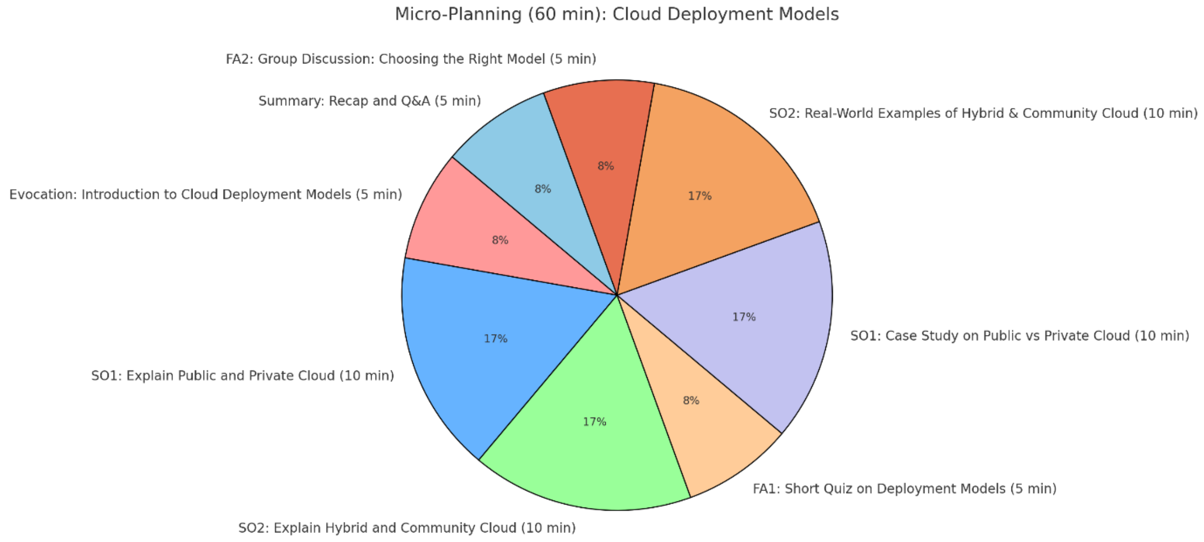
*Head*
*Department of Computer Science*
*St. Mary's College (Autonomous)*
*Thoothukudi.*

# Lesson Plan

| | |
|---|---|
| Programme | B. Sc. Computer Science |
| Semester | VI |
| Course Title | Cloud Computing |
| Code | 21UCSE61 |
| Hours | 4 |
| Total Hours | 60 |
| Credits | 4 |
| Max Marks | 100 |
| Unit & Title | Unit I: Deployment model |
| Name of the Faculty | S.Gnanathangam |
| T-L tools | Mind Maps, Video, PowerPoint Presentation |

**Pre-requisite Knowledge:** basic understanding of recent trends of computing and its benefits.

**Micro-Planning: (60 minutes)**



| Phase | Activity | Duration |
|---|---|---|
| Evocation | Introduction to Cloud Deployment Models | 5 min |
| SO1 | Explain Public and Private Cloud | 10 min |
| SO2 | Explain Hybrid and Community Cloud | 10 min |
| FA1 | Short Quiz on Deployment Models | 5 min |
| SO1 | Case Study on Public vs Private Cloud | 10 min |

|  |  |  |
|---|---|---|
| SO2 | Real-World Examples of Hybrid & Community Cloud | 10 min |
| FA2 | Group Discussion: Choosing the Right Model | 5 min |
| Summary | Recap and Q&A | 5 min |

1**. Evocation (5 min)**

- Ask students where they store their personal data (e.g., Google Drive, Dropbox, iCloud).
- Discuss the need for cloud storage and different ways organizations deploy cloud services.
- Introduce the concept of cloud computing

**2. Topic Introduction:**

**2.1 General Objective:**

- To understand the cloud deployment models and their use cases.
- To learn the functions of cloud deployment models .

**2.2** Specific Outcomes (SO1 & SO2):

- To explain Public and Private Cloud model
- To demonstrate deployment models

**First Phase**: Explanation (20 min)

SO1 (10 min): Explain the Features, Pros & Cons (Example: AWS, Google Cloud) of Public cloud and Private Cloud (Example: Banks, Government Agencies)

SO2 (10 min): Explain the Features, Pros & Cons of Hybrid cloud (Example: Large Enterprises) and Community cloud (Example: Healthcare, Research Institutes)

**Second Phase**: Implementations and Applications (20 min)

SO1 (10 min): Case Study - Compare AWS (Public Cloud) vs. On-Premise Data Centers (Private Cloud)

SO2 (10 min): Real-World Scenarios - Discuss Netflix (Hybrid Cloud) and Educational Institutes (Community Cloud).

Mind Map (2 min)

Draw a mind map that illustrate deployment models and their functions

Summary (2 min)

Summarize the deployment models with service providers

### 2.3. Taxonomy of Objectives:

| Taxonomy of objectives | | | | | | |
|---|---|---|---|---|---|---|
| Knowledge Dimension | The Cognitive Process Dimension | | | | | |
| | Remember | Understand | Apply | Analyse | Evaluate | Create |
| A. Factual Knowledge | 1 | 1 | | | | |
| B. Conceptual Knowledge | | 2 | | | | |
| C. Procedural Knowledge | | | | | 1 | |

### 2.4. Key Words:

Public Cloud

Private Cloud

Hybrid Cloud

Community Cloud

### 2.5 Powerpoint Presentation

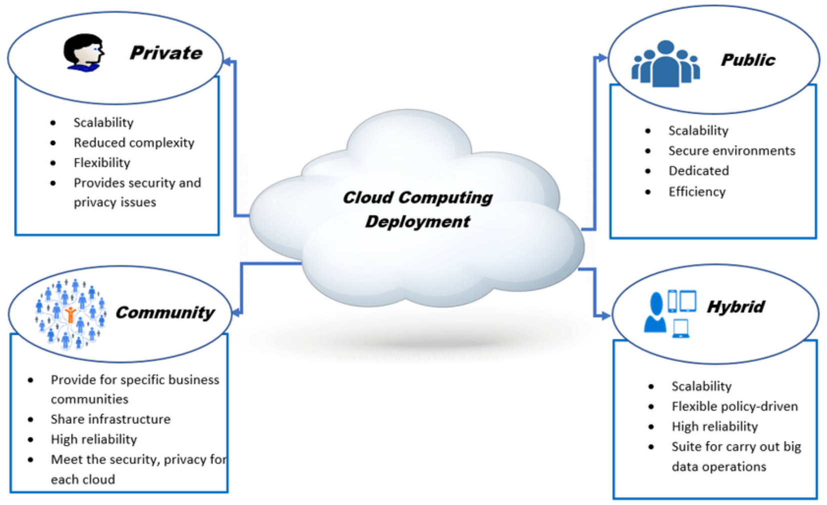
https://docs.google.com/presentation/d/1fSwC0uAlLEYico7PNX657pQu6oaRHXGb/edit?usp=sharing&ouid=115208675943936491706&rtpof=true&sd=true

### 3. Discussion:

- Compare the models in deployment.
- Discuss the limitations of Deployment models.

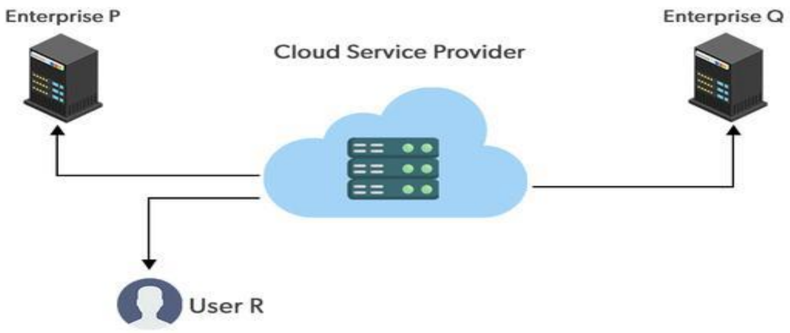
### 4. Mind Map:

Deployment Models in Cloud Computing



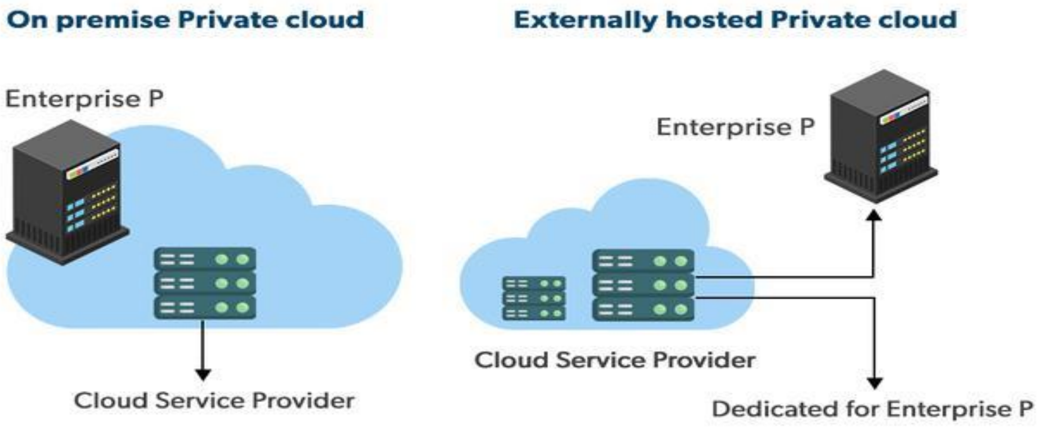
**5. Summary:** Walk through the models of deployment

Public Cloud



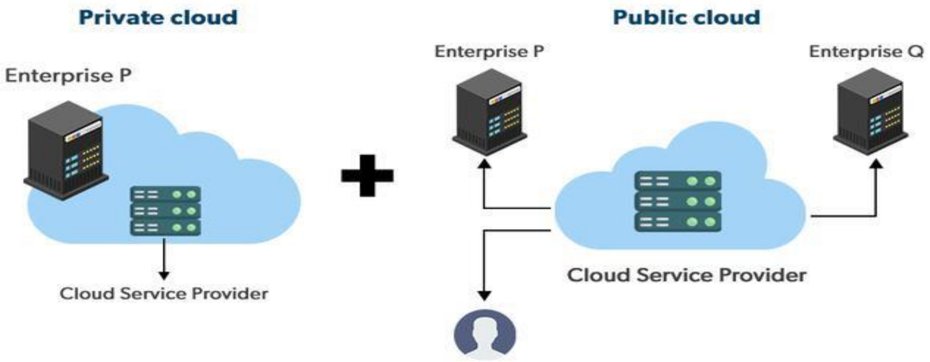
The public cloud makes it possible for anybody to access systems and services. The public cloud may be less secure as it is open to everyone. The public cloud is one in which cloud infrastructure services are provided over the internet to the general people or major industry groups. The infrastructure in this cloud model is owned by the entity that delivers the cloud services, not by the consumer. It is a type of cloud hosting that allows customers and users to easily access systems and services. This form of cloud computing is an excellent example of cloud hosting, in which service providers supply services to a variety of customers. In this arrangement, storage backup and retrieval services are given for free, as a subscription, or on a per-user basis. For example, Google App Engine etc.

Private Cloud



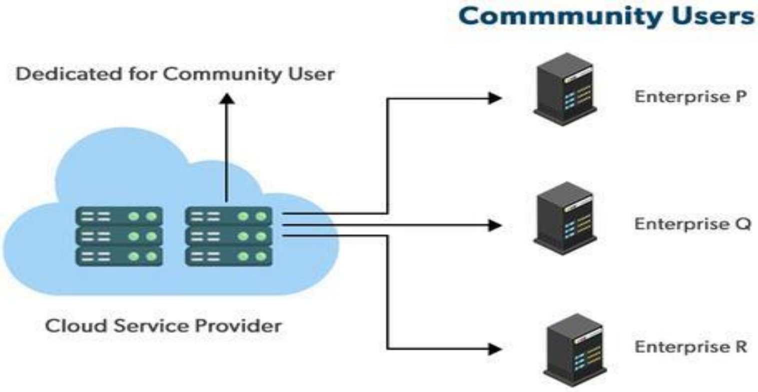
The private cloud deployment model is the exact opposite of the public cloud deployment model. It's a one-on-one environment for a single user (customer). There is no need to share your hardware with anyone else. The distinction between private and public clouds is in how you handle all of the hardware. It is also called the “internal cloud” & it refers to the ability to access systems and services within a given border or organization. The cloud platform is implemented in a cloud-based secure environment that is protected by powerful firewalls and under the supervision of an organization’s IT department. The private cloud gives greater flexibility of control over cloud resources.

Hybrid Cloud



By bridging the public and private worlds with a layer of proprietary software, hybrid cloud computing gives the best of both worlds. With a hybrid solution, you may host the app in a safe environment while taking advantage of the public cloud’s cost savings. Organizations can move data and applications between different clouds using a combination of two or more cloud deployment methods, depending on their needs.

Community Cloud



It allows systems and services to be accessible by a group of organizations. It is a distributed system that is created by integrating the services of different clouds to address the specific needs of a community, industry, or business. The infrastructure of the community could be shared between the organization which has shared concerns or tasks. It is generally managed by a third party or by the combination of one or more organizations in the community.

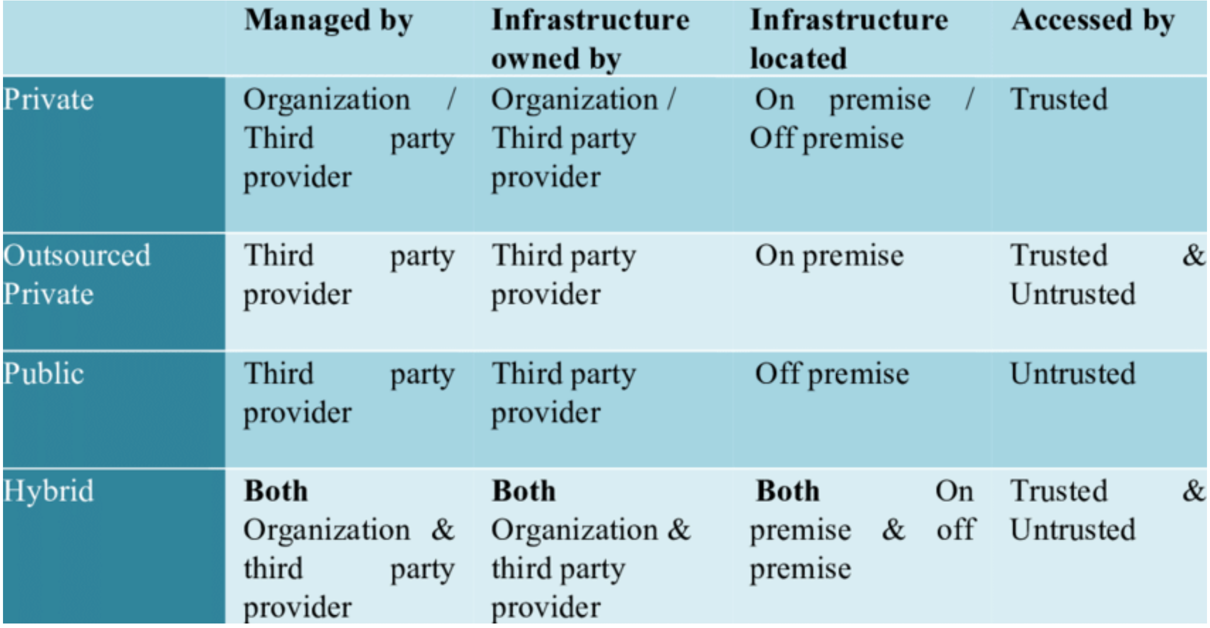
What is the Right Choice for Cloud Deployment Model?

As of now, no such approach fits picking a cloud deployment model. We will always consider the best cloud deployment model as per our requirements. Here are some factors which should be considered before choosing the best deployment model.

- Cost: Cost is an important factor for the cloud deployment model as it tells how much amount you want to pay for these things.
- Scalability: Scalability tells about the current activity status and how much we can scale it.
- Easy to use: It tells how much your resources are trained and how easily can you manage these models.
- Compliance: Compliance tells about the laws and regulations which impact the implementation of the model.
- Privacy: Privacy tells about what data you gather for the model.

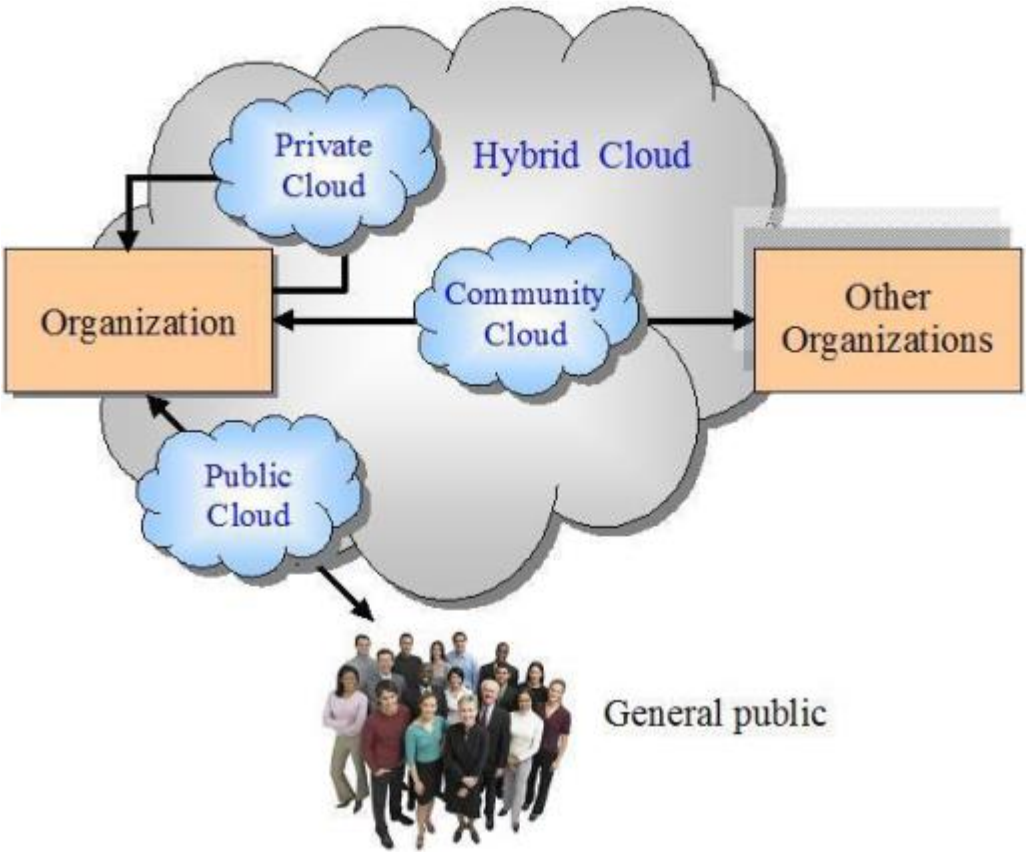
Each model has some advantages and some disadvantages, and the selection of the best is only done on the basis of your requirement. If your requirement changes, you can switch to any other model.

Overall Analysis of Cloud Deployment Models



| | **Managed by** | **Infrastructure owned by** | **Infrastructure located** | **Accessed by** |
|---|---|---|---|---|
| Private | Organization / Third party provider | Organization / Third party provider | On premise / Off premise | Trusted |
| Outsourced Private | Third party provider | Third party provider | On premise | Trusted & Untrusted |
| Public | Third party provider | Third party provider | Off premise | Untrusted |
| Hybrid | **Both** Organization & third party provider | **Both** Organization & third party provider | **Both** On premise & off premise | Trusted & Untrusted |

The overall Analysis of these models with respect to different factors is described below.



**6. Assessment through questions/analogy/new ideas:**

Formative Assessment 1 (FA1) (2 min)

Short quiz with 3-5 multiple-choice questions on Deployment models.

1. What are the key differences between Public and Private Cloud?

2.Why do some companies prefer a Hybrid Cloud model?

3.Give an example of an industry where a Community Cloud would be useful.

Formative Assessment 2 (FA2) (2 min)

Group discussion: "Which Cloud Deployment Model is best for a Startup vs. Large Enterprise?"

**7. FAQ’s: MCQ’s/ Descriptive questions:**

1.Examine the relationship between Amazon and Azure.

2.Anayse the pros and cons of cloud computing

**8. References:**

1. Mell, P., & Grance, T. (2011). The NIST Definition of Cloud Computing.

2.Cloud Computing Principles and Paradigms – Wiley Publications.

3. Barrie Sosinsky. "Cloud Computing Bible". New Delhi: Wiley India Pvt. Ltd. 2012.

**9. Verified by Subject Expert:**

S. Gnanathayan

**Course In-charge**

Reetha

**Approved by HoD.**

Head
Department of Computer Science
St. Mary's College (Autonomous)
Thoothukudi.