	<p style="text-align: center;"><b>SVKM's NMIMS</b>  <b>Mukesh Patel School of Technology Management &amp; Engineering / School of</b>  <b>Technology Management &amp; Engineering</b></p>		
<b>B. Tech/MBA Tech</b>	<b>Workbook</b>		<b>Academic Year- 2024-25</b>
<b>Year:-First</b>	<b>Subject:-</b> Programming for Problem Solving	<b>Semester: - First</b>	

## Experiment: 9

### PART A

(PART A: TO BE REFERRED BY STUDENTS)

**Aim:** Programming using object-oriented programming concepts (using data members and member functions)

**Learning Outcomes:** The learner would be able to


1. Understand the concept of object-oriented programming
2. Solve problems using class and objects
3. Implement programming using overloading of functions

### Theory:

Note: - Theory part is continued from page number two.

### Tasks:

1. Create a class named 'Employee' with a string (char array) variable 'name' and float variable 'salary'. Assign the value of salary as 20000.67 and that of name as "Scott" in main( ) function by creating an object of the class Employee and display the same.
2. Create a class Employee having data members name, salary & department and define two member function getData( ) & showData( ) for taking input & display the same. Write a complete C++ code for displaying the information of a Employee.
3. Create a student record (name, rollno, marks of 3 subjects and score), calculate the average, store average in a score data member. If score<40, declare FAIL else PASS along with student details, maintain 10 students records. (make use of member function to read and display records)
4. Write a program to overload sum function to perform addition of two integers, three integers and two floating-point numbers.
5. Create a class named "Shapes" with data member area. Write a member function "calArea" with two float parameters to calculate the area of rectangle and overload the same function having one float parameter to calculate the area of square.

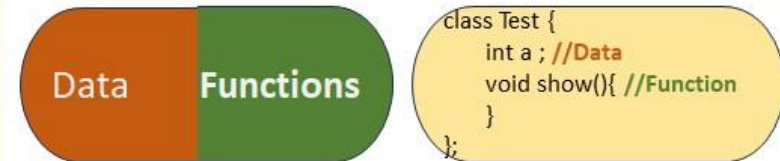
	<p align="center"><b>SVKM's NMIMS</b>  <b>Mukesh Patel School of Technology Management &amp; Engineering / School of Technology Management &amp; Engineering</b></p>		
<b>B. Tech/MBA Tech</b>	<b>Workbook</b>	<b>Academic Year- 2024-25</b>	
<b>Year:-First</b>	<b>Subject:-</b> Programming for Problem Solving	<b>Semester: - First</b>	

### What is Object Oriented Programming (OOP)

If any programming language supports **Encapsulation**, **Polymorphism**, **Inheritance** and **Abstraction** etc. is called as **OOP**

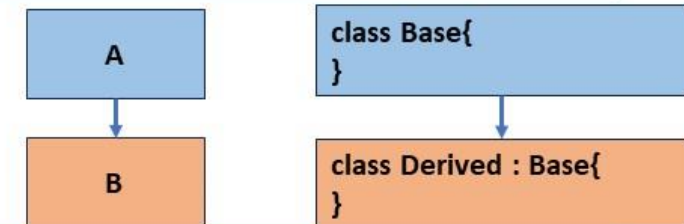
#### Encapsulation:-

- Binding/Wrapping of data to its related functions into a single unit
- In OOP we can achieve encapsulation by creating Class & Objects.



#### Inheritance:-

- Inherence is reusability (of encapsulation).
- Child/Derived/ Sub Class inherits the properties of Parent/Base/ Super Class.



#### Polymorphism (many forms):-

- Defining multiple functions with same name
- These functions with same name, will show different behaviours
- There are two types of polymorphism → Static(Overloading) & Dynamic(Overriding)

```
void area(float radius){
    cout<<3.147*radius;
}

void area(int base, int ht){
    cout<<(0.5*base*ht);
}
```


Here, area() function is showing two behaviors,  
 1. Finding Area of Circle  
 2. Finding Area of right angle triangle  
**This is function overloading**

#### Abstraction:-

- It hides the implementation details.
- A Class can decide which member will be visible to the outside of class
- cin>> & cout<< implementation details are in iostream, which is hidden

```
#include<iostream>
#include<math.h>
using namespace std;
int main(){
    cout<< sqrt(25);
    return 0;
}
```

Here, implementation details of sqrt() is hidden as it is in math.h and details of cout<< is also hidden as it is in iostream... this way abstraction is achieved

	<p align="center"><b>SVKM's NMIMS</b>  <b>Mukesh Patel School of Technology Management &amp; Engineering / School of Technology Management &amp; Engineering</b></p>		
<b>B. Tech/MBA Tech</b>	<b>Workbook</b>	<b>Academic Year- 2024-25</b>	
<b>Year:-First</b>	<b>Subject:-</b> Programming for Problem Solving	<b>Semester: - First</b>	

**Write a program to read and display Student information like roll number & name using class & Object...**

**class name** → `class Student{`

**public, private & protected are access specifier or visibility modifier. private is default** → `public:`

**Here, Student is user defined datatype** → `Student s;`

**Reading class data members** → `cin>>s.rollno>>s.name;`

**Displaying class data members** → `cout<<"Name = "<<s.name<<endl;`

```

#include <iostream>
using namespace std;
class Student{
public:
    int rollno;
    char name[10];
};

int main() {
    Student s;
    cout<<"Enter roll no and name of student"<<endl;
    cin>>s.rollno>>s.name;
    cout<<"Students Details are"<<endl;
    cout<<"Roll No = "<<s.rollno<<endl;
    cout<<"Name = "<<s.name<<endl;
    return 0;
}

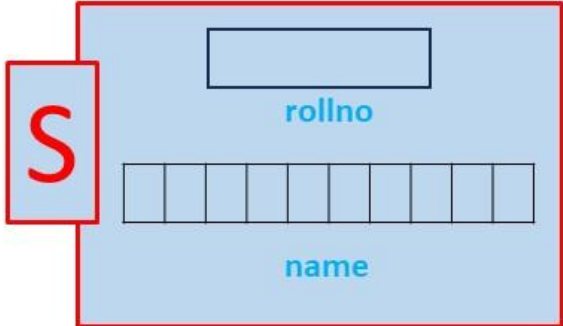
```

**public is access specifier is used to access rollno & name outside (here, in main()) of the class.**

**Class data members or fields or instance variable** → `int rollno;` and `char name[10];`


**Object Declaration or instance of a class** → `Student s;`

**Memory Allocation**



**Now, dot (.) operator is required to access class members**  
**Ex. s.rollno**



	<p style="text-align: center;"><b>SVKM's NMIMS</b>  <b>Mukesh Patel School of Technology Management &amp; Engineering / School of Technology Management &amp; Engineering</b></p>		
<b>B. Tech/MBA Tech</b>	<b>Workbook</b>		<b>Academic Year- 2024-25</b>
<b>Year:-First</b>	<b>Subject:-</b> Programming for Problem Solving	<b>Semester: - First</b>	

## Classes and Objects

### Class:

- A class is user defined data type
- A class is blueprint of an object
- One class can have many objects

### Syntax of class definition

```
class classname{
    access_specifier:
        datatype datamemberi;
        :
        datatype datamembern;
    access_specifier:
        type memberfunction1() {
            //body
        }
        :
        type memberfunctionn() {
            //body
        }
};
```

### Objects:

- object is an instance of a class
- due to objects class comes to the existence
- Object have access to members of a class.


### Syntax of object declaration

```
classname objectname;
```

Define a class Cricket with data members player\_name, team\_name & batting\_average and Member Function as read( ) and display( ). Write C++ program to read and display information of a players of a team.

```
class Cricket{
    private:
        char player_name[10];
        char team_name[10];
        float batting_average;
    public:
        void read( ){
            cout<< "Enter Player name,Team name & bat avg of player\n";
            cin>>player_name>>team_name>>batting_average;
        }
        void display( ){
            cout<< "Player Name="<<player_name;
            cout<< "Team Name"<<team_name;
            cout<< "Batting Avg="<<batting_average;
        }
};

int main( ){
    Cricket p;
    p.read( );
    p.display( );
    return 0;
}
```

	<p style="text-align: center;"><b>SVKM's NMIMS</b>  <b>Mukesh Patel School of Technology Management &amp; Engineering / School of Technology Management &amp; Engineering</b></p>		
<b>B. Tech/MBA Tech</b>	<b>Workbook</b>		<b>Academic Year- 2024-25</b>
<b>Year:-First</b>	<b>Subject:-</b> Programming for Problem Solving	<b>Semester: - First</b>	

## Function Overloading

Write a program to find area of circle and rectangle using function overloading

### Program without using class & Object

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

void area(float r){
    cout<<"Area of Circle="<<3.147*r*r;
}

void area(float l, float b){
    cout<<"\nArea of rectanlge"<<l*b;
}

int main() {
    area(40.0);
    area(10.0,20.0);
    return 0;
}
```

### Program using class & Object

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

class OverloadingTest{
    void area(float r){
        cout<<"Area of Circle="<<3.147*r*r;
    }
    void area(float l, float b){
        cout<<"\nArea of rectanlge"<<l*b;
    }
};

int main() {
    OverloadingTest ot;
    ot.area(40.0f);
    ot.area(10.0f,20.0f);
    return 0;
}
```



**SVKM's NMIMS**  
**Mukesh Patel School of Technology Management & Engineering / School of**  
**Technology Management & Engineering**

**B. Tech/MBA Tech**

**Workbook**

**Academic Year- 2024-25**

**Year:-First**

**Subject:-** Programming for Problem Solving

**Semester: - First**