G. H. Raisoni
College of Engineering & Management
NAAC Accredited A+ Grade
(An Empowered Autonomous Institute Affiliated to Savitribai Phule Pune University)
Gat No. 1200, Domkhel Road, Wagholi, Pune - 412207

RAISONI GROUP
a vision beyond

Certificate

This is to certify that

Mr. / Ms. Tanmay Prashant Dalvi

studying in Semester / year 3rd Sem. (SY)

Branch _____

Name of the Dept. CSE-AIML

Section B

has satisfactorily completed the practical work of subject Object Oriented Programming

during the academic year 2023-24

Signature of Subject Teacher

Signature of Head of Department

INDEX PAGE

Sr. No	Name of the Experiment	Page No.	Date of Performance	Sign of Teacher & Date
1)	Write a program to compute the area of triangle & circle by overloading the area fun^c	1.		
2)	Define a class to represent bank acc. Include foll. members data mem. name of depositor, acc. num. type of acc, balance amount in account mem. fun^c. To assign initial values, to deposit an amount, to withdraw an amount after checking the balance, to display name & balance. Write a main program to test prg. using an object.	4		
3)	Create two classes DM & DB which stores values of distances. DM in meters & DB in feets & inches. Write a prg. that can read values for class object and add one object of DM with another object of DB. Use a friend function to carry out addition operation	7		

INDEX PAGE

INDEX PAGE

RAISONI GR
— a vision beyon

Sr. No	Name of the Experiment	Page No.	Date of Performance	Sign of Teacher & Date	Remar
	A function display() must be defined in all classes. to display class contents. write a program using polymorphism & virtual fun^c.				
8)	Write a program to show use of 'this' pointer & 'new' and 'delete'	20			
9)	Write a function 'template' for finding the value contained in an array.	21			
10)	Write a program containing a possible exception. Use a 'try' block to throw it and 'catch' block to handle it properly.	24			

RAISONI GROUP
— a vision beyond —

* ASSIGNMENT: No. 01

Title :- Write a program to compute the area of triangle and circle by overloading the area() function.

Problem Statement :- Implement a C++ program to understand the concept of overloading.

Objective :-

1. Understand the basic principles of OOP.
2. Implement the concept of overloading.
3. Develop program using object oriented concepts.

Theory :-

- C++ Structures:-

Structure is a collection of variables of different data types under a single name. It is similar to class in that, both holds a collection of data of different datatypes.

For example- You want to store some information about a person : his/her name, citizenship number and salary. You can easily create different variables name, citNo., Salary to store these information separately.

RAISONI GROUP
— a vision beyond —

How to declare Structures in C++ ?

The 'struct' keyword defines a structure type followed by an identifier (name of the structure).
Then inside the curly braces, you can declare one or more members of that structure.
For example:

```
Struct Person
{
char name [50];
int age;
float Salary;
};
```

Difference of a Structure and a Class.

	Structure	Class
i)	A structure is a collection of variables of different data types under a single unit. It is almost similar to class because both are user defined data types and are different.	i) A class is a user defined blue print of prototype from which obj are created. A class combines the fields and methods into a single unit.
ii)	If access specifier is not declared, by default all members are public.	If access specifier not declared, by default all members are public.

iii)	Declaration of Structure:
	
	struct structure_name {
	typestruct element 1;
	typestruct_element 2;
	typestruct_element 3;
	};
		Declaration of class:
		
		class class_name {
		data member;
		member function;
		};

iv)	Instance of 'structure' is called 'structure variable'	Instance of 'class' is called 'object'
v)	Structure has limited features	Class has unlimited features
vi)	Structures are used in small programs.	Class are used in large programs.
vii)	Structure does not contain parameters less constructor or destructor, but can contain Parameterized constructor or static constructor.	Class can contain constructo or destructor.
viii)	Each variable in struct contains its own copy in data (exept in ref and out parameter variable) and any operations on one variable cannot effect another variable.	Two variables of class can contain the reference of the same object and any oper^ on one variable can affect another variable.

RAISONI GROUP
— a vision beyond —

* ASSIGNMENT NO. 02 :-

Title :- Define a class to represent a bank account. Include the following members: data members, name of depositor, account number, type of account, balance amount in the account member function. To assign initial values, to deposit an amount, to withdraw an amount after checking the balance, to display name & balance. Write a main program to test program using and object.

Problem Statement :- Implement C++ programs which co·indude a class having data members and member function and object to access these members of class.

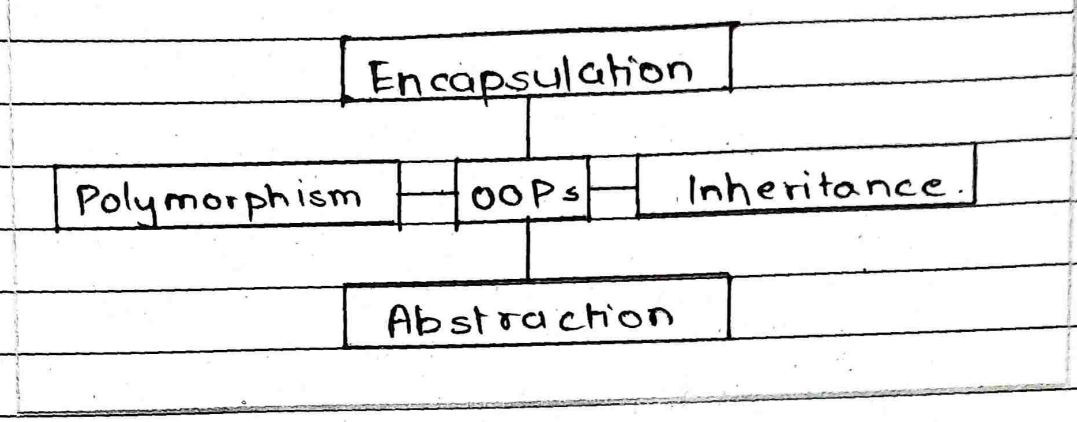
Objective :-

1. Understand the basic principles of OOP.
2. Develop programs using object oriented concepts.

Theory :-

o Object Oriented Programming :-

OOP is a programming style that is associated with the concept of class, object and various other concepts revolving around these two, like inheritence, Polymorphism, abstraction, Encapsulation, etc.



- **Class:-**

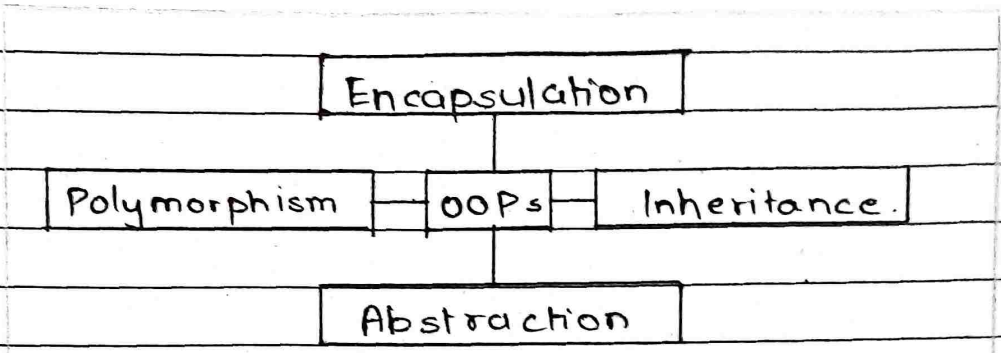
Here, we can take humanbeing as a class. A class is a blueprint for any function entity which defines its properties and its functions. like Human being, have body parts, and performing various actions.

- **Inheritance:-**

Considering Human being as a class, which has properties like hands, legs, eyes, etc. and functions like walk, talk, eat, see, etc. Male and female are also classes, but most of the properties and functions are included in Human being Hence, they can inherit everything from class Human being using the concept of Inheritance.

- **Abstraction:-**

It means, showcasing only the required things to outside world while hiding the details. Continuing our example. Human beings can talk, walk, etc. but the details are hidden from outside world.



- **Class:-**

Here, we can take humanbeing as a class. A class is a blueprint for any function entity which defines its properties and its functions. Like Human being, have body parts, and performing various actions.

- **Inheritance:-**

Considering Human being as a class, which has properties like hands, legs, eyes, etc. and functions like walk, talk, eat, see, etc. Male and female are also classes, but most of the properties and functions are included in Human being. Hence, they can inherit everything from class Human being using the concept of Inheritance.

- **Abstraction:-**

It means, showcasing only the required things to outside world while hiding the details. Continuing our example. Human beings can talk, walk, etc. but the details are hidden from outside world.

- **Encapsulation :-**

This concept is little tricky to explain with our example. Our legs are behind to help us walk. Our hands, help us to hold things. This binding of the properties to function is called encapsulation.

- **Polymorphism:-**

It is a concept, which allows us to redifine the way something works, by either changing how it is done or by changing the parts using which it is done. Both the ways have different terms of them.

RAISONI GROUP
— a vision beyond —

* ASSIGNMENT NO: 03

Title-
Create two classes DM and DB which stores values of distances. DM stores distances in meters & cm's and DB in feets and inches. Write a program that can read values for the class object and add one object of DM with another object of DB. Use a friend function to carry out addition operation.

Problem Statement-
Implement a C++ program to understand concept of Friend function.

Objective-
1. Understand the basic principles of OOP.
2. Apply the concepts of inheritance and friend function.
3. Develop program using OOP concept.

Theory:-

- C++ Friend function-
A friend fun^n of a class is defined outside the class scope but it has the right to access all private & protected members of the class. Even though the prototypes for the friend functions appear in the class definition, friends are not member functions.

Declaration of friend function.

```
class class_name {
...
friend return_type function_name (arguement);
...
}
```

- Now you can define the friend function as a normal function to access the data of the class. No friend keyword is used in the definition.

- Consider in above case:

```
class class_name {
...
friend returntypefunction_name (arguements);
...
}
return_type function name (arguements)
{
...
...
}
```

ASSIGNMENT NO. 04

Title:-
Create a class MAT of size m*n. Define all possible matrix operations for MAT type objects.

Problem Statement:-
Implement a C++ program to understand concept matrix operations.

Objective:-
1. Understand the basic principles of OOP.
2. Apply the concepts of overloading, inheritance, polymorphism
3. Develop program using OOP concepts.

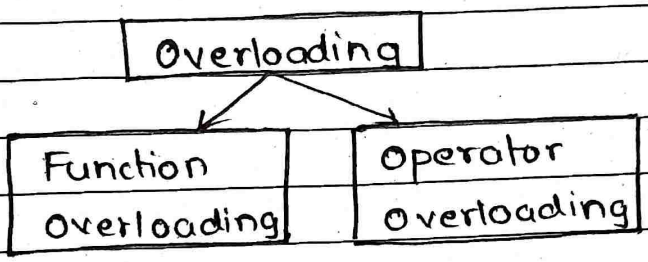
Theory:-
If we create two or more members having the same name but different in number or type of parameter, it is known as C++ overloading. In C++, we can overload.

- Methods
- Constructors
- Indexed properties.

It is because these members have parameters only.

Types of overloading in C++ are:-

- Function overloading
- Operator overloading



- C++ Function overloading -

Function overloading is defined as the process of having two or more function with the same name, but different in parameters is known as function overloading in C++

The advantage of Function overloading is that it increases the redability of the program because you don't need to use different names for the same action.

Example :-

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

class Cal {
public:
static int add (int a, int b) {
return a+b;
}
```

* ASSIGNMENT NO. 05 :-

Title :-
Create Stud class to display student information using constructor and destructor.

Problem Statement :-
Implement a C++ program to understand concept of constructor and destructor.

Objective :-
1. Understanding the basic principles of OOP
2. Apply the concept of overloading, inheritance, polymorphism
3. Develop program using OOP.

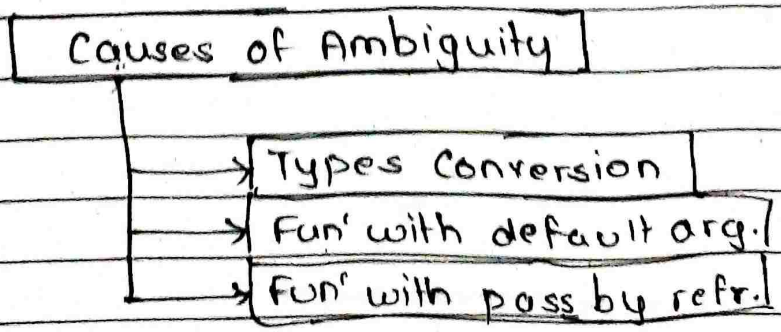
Theory :-

- C++ Constructor-
In C++, constructor is a special method which is invoked automatically at the time of object creation. It is used to initialize the data members of new object generally.
The constructor in C++ has the same name as class or structure
- Default Constructor
- Parameterized Constructor.

```
static int add (int a, int b, int c)
{
 return a+b+c;
}
};
int main (void) {
 cal C;
 cout << C.add (10,20) << endl;
 cout << C.add (12,20,23);
 return 0;
}
```

Output-
30
55

- Function overloading and Ambiguity :- When the compiler is unable to decide which function is to be added among the overloaded function, this situation is known as function overloaded.
- Causes of function overloading:-
- Type Conversion.
- Function with default arguments
- Function with pass by reference.



* C++ Default Constructor :-
A constructor which has no argument is known as default constructor. It is invoked at the time of creating object.

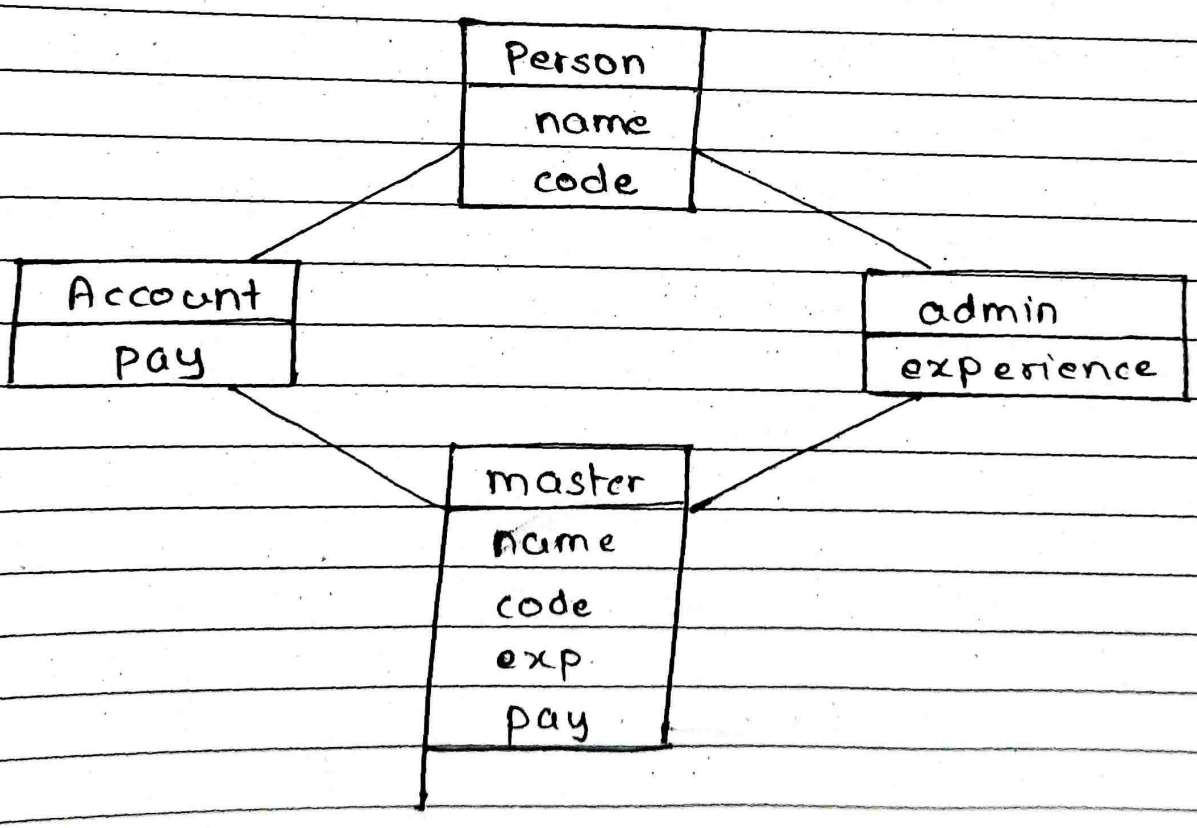
* C++ Parameterized Constructor :-
A constructor which has parameters is called para-constructor. It is used to provide different values to distinct object.

* C++ destructor
A destructor works opposite to constructor; it destructs the objects of classes. It can be defined only once in a class. Like construction, in is invoked automatically. It is defined with tidle sign (~)

* ASSIGNMENT NO.06 :-

Title -

Consider a network of given figure. The class matter derives information from both account and admin classes which is turn derive information. from the class person Define all the four classes and write a program to create, update and display the information contained in master objects.



Problem Statement :-

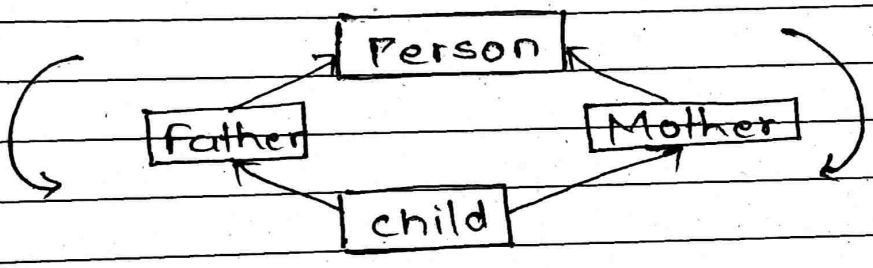
Implement a C++ program to understand concepts of multiple and multilevel inheritance using virtual fun^c

Objective :-

1. Understand the basic principle of OOP
2. Apply the concepts of inheritance
3. Develop programs using OOP concepts.

Theory :-

The Diamond problem occurs when a child class inherits from two parents classes who both share a common grandparent class. This is illustrated in diagram.



Here, we have a class child inheriting from classes, Father and Mother. These two classes, in turn, inherit the class Person because both Father & mother are Person.

As shown in Fig., class Child inherits the traits of class Person twice-once from Father and again from mother. This gives rise to ambiguity since the compiler fails to understand which way to go.

This scenario gives rise to a diamond shape inheritance graph and is famously called "The diamond Problem"

The Solution to the diamond problem is to use virtual keyword. We make the two parent classes (who inherit from the same grandparent class) into virtual classes in order to avoid two copies of grand parent in child class.

In other words, the child class will have a single instance of Person class, shared by both the Father and mother class. By having a single instance of the person class, the ambiguity is resolved.

* ASSIGNMENT NO.07 :-

Title :-

A book shop sells both books and video tapes. Create a class media that stores the title and price of the publication. Create two derived classes, one for storing number of pages in the book and another for storing playing time of tape. A function display() must be defined in all cases to display class contents. Write a program using polymorphism and virtual function.

Problem Statement :-

Implement a C++ program to understand concept of polymorphism and virtual functions.

Objective :-

1. Understand the basic principle of OOP
2. Apply the concept of overloading, inheritance, polymorphism.
3. Develop programs using OOP concepts.

Theory :-

- Static & Dynamic Binding :-

Polymorphism means „one name" - „multiple forms.

The overloaded member functions are "selected" for invoking by matching arguements, both type and number.

~~The overloaded member function~~

Early binding means that an object is bound to its function call at compile time. It would be nice if the appropriate member function could be selected while program is running. This is known as runtime polymorphism.

C++ supports a mechanism known as virtual function to achieve a run time polymorphism. At the runtime, when it is known what class objects are under consideration, this term is late binding. Also known as Dynamic Binding, because the selection of appropriate function is done dynamically at run time

- Virtual Functions :-

Polymorphism refers to the property by which the objects belonging to different classes are able to respond to the same message, but different forms. When we use the same function name in both the base and derived classes, the function in base class is declared as virtual using keyw. virtual proceeding its normal declrations. Thus, by making the base pointer to different objects, we can execute the different versions of virtual function

Rules for Virtual function:-

When virtual functions are created for implementing late binding, observe the some basic rules that satisfy the compiler requirement.

1. The virtual functions must be member of some class.
2. They cannot be static members.
3. They are accessed by using object pointers.
4. A virtual function can be a friend of another class.
5. A virtual function in a base class must be defined, even though it may not be used.
6. The prototypes of base class version of vf and all the derived class versions must be identical. C++ cons. them as overloaded functions. and vf mechanism is ignored.
7. We cannot have virtual constructors, but we can have virtual destructors.
8. While a base pointer to a derived class, incrementing or decrementing it will not make it
9. when a base pointer points to any type of derived class, the reverse is not true.
10. If a virtual function is defined in the base class, it need not be necessary redifined in the derived class. In such case, calls will invoke the base function.

RAISONI GROUP
— a vision beyond —

* **ASSIGNMENT NO. 08 :-**

Title :- Write program to show use of this pointer, new and delete.

Problem Statement :- Implement C++ program to understand concept of memory allocation using new, del

Objective :- i) Understand principles of OOP.
ii) Implement memory allocation using exp. handling, gene. prog.
iii) Develop programs using OOP concepts.

Theory :- New and Deletes are unary operators that are used for allocation & deallocation of memory. A object can be created- new, destructed- delete.

- **new operator :-** can be used to create obj. of any type. Hence new operator allocate memory to hold sufficient data and return address of allocated memory.
  - Example : int *p = new int;
    → int * p = new int [10] (for array creation)

- **delete operator** - If the variable or object is no longer required or needed is destroyed by 'delete'
  Syntax : delete pointer_variable;
  Example : delete p;
  If we want to free a dynamically allocated array :
  delete [size] pointer_variable;

* ASSIGNMENT No: 09

Title:-
Write a function template for finding the value contained in an array.

Problem Statement:-
Implement a c++ program to understand concept of template.

Objective:-
1. Understand the basic principles of OOP.
2. Implement memory allocation techniques and usage of exception handling, generic programming.
3. Develop programs using OOP concept.

Theory:-
Instead of writing different functions for diff. data types, we can define common function.
For example:
int max (int a, int b);
max (float a, float b);
max (char a, char b);
(this is called fun' overloading)
But instead of writing three different functions as above, c++ provided the facility called templates

With the help of templates you can define only one common function as follows:

T max (Ta, Tb); // This T is called generic data type.

Template functions are the way of making functions/ class abstracts by creating the behaviour of function without knowing what data will be handled by function. For example you could make a templated stack push function. This push function can handle this insertion operation to a stack on any data type rather than having to create a stack push function for each diff. type.

Syntax:

```
template < class type>
 ret_type - fun_name (parameter list)
   {
   ...
   ...
   }
```

Features of templates:-

1. It eliminates redundant code
2. It enhances reusability of code
3. It provides great flexibility to language

Function templates:-

The templates declared for functions are called as function templates. A function templates defines how an indivisual function can be constructed.

Syntax:

```
template < class type>
return_type fun_name (argument)
{
  ---  //body
  ---
}
```

* ASSIGNMENT NO.10

Title -
Write a program containing a possible exception. Use a try block to throw it and catch block to handle it properly

Problem Statement :-
Implement a C++ program to understand concept of error handling

Objective :-
1. Understand the basic principles of OOP
2. Implement memory allocation techniques and usage of exception handling, generic programming.
3. Develop program using OOP concepts.

Theory :-
- Exception Handling :-

Exceptions are runtime anomalies or unusual conditions that a program may encounter while executing. Anomalies might include conditions such as zero, accessing an array outside of its bounds or running out of memory or disk space.
Exception provide a way to transfer control from one part of program to another. C++ is built upon three keywords : try, catch & throw.

Types of exceptions:-

① Synchronous exception
② Asynchronous exception

Exception Handling mechanism:-

An exception is said to be thrown at the place where some error or abnormal condition is detected. The throwing will cause the normal program flow to be aborted, in a raised exception. The catch block can be and usually is, located in different function than the point of throwing. C++ exception handling is based upon three keywords: try, catch, and throw. Try is used to preface a block of statement which may generate exceptions. This block of statement is known as try block. When an exception is detected it is thrown by using throw statement in the try block. Catch block catches the exception thrown by throw statement in the ~~catch~~ try block and handles it appropriately.