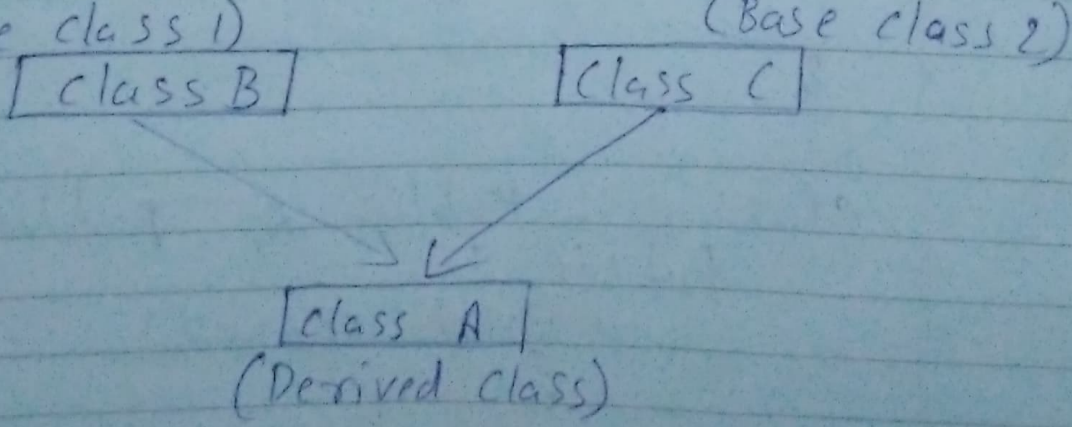
Assignment 3

Q1 What is inheritance? Explain multiple inheritance and discuss the problem with the mutiple inheritance and how do solve that?

Ans

The capability of a class to derive properties and characteristics from another class is called Inheritance. Inheritance is one of the most important feature of object oriented programming.

-> Multiple inheritance is a ~~feature of~~ feature of c++ where a class can inherit from more than one class i.e. one sub class is inherited from more than one base class.



→ The problem with multiple inheri
is Ambiguty.

Ex

```
class A
{
    public:
        void display()
        {
            cout << "class A";
        }
};

class B
{
    public:
        void display()
        {
            cout << "class B";
        }
};

class c : Public A, public B {
    Public:
        void show()
        {
            display();
        }
};
```

→ The problem is compiler cannot understand which display function is to be called of class A or class B as both the class are inherited in class C. The above problem can be solved by making display function "virtual" or using the scope resolution operator in show() function of class C.

```
Void show()
{
    A :: display();
    B :: display();
}
```

Q2 What is the difference between is-a and part of relationship

Ans

Is-a Relationship:

→ Defines a week-coupled relationship between two entities where one entity could be part of another but either can exist without the other

→ Is-a relationship doesn't necessarily own any of its aggregation.

→ Is a relationship has weaker or looser bounds with aggregation

→ Is a relationship has aggregation that lives at the outer level.

→ Eg. Car is a vehicle.

Part of Relationship:

→ Defines a strong-coupled relationship between two entities where the one entity is part of another and both need each other for their existence.

-> Part of relationship implies real ownership of its components

-> Part of relationship has stronger bounds of its components.

-> Part of relationship has components that exist at the inner level.

-> E.g. Engin is part of vehicle.

Q3 Explain derivation using different access modifiers.

Ans

Member type in base class	type of derivation	member type in derived class
private	private Protected Public	Not accessible Not accessible Not accessible
protected	private Protected Public	Private Protected Protected
public	private protected public	Private Protected Public.

Q-4 Explain Run time Polymorphism:-

Ans

-> Polymorphism means same thing having many forms. It allows the object condition. It is the ability of a single object to appear in many forms.

-> one person be a student in a college or an employer in a working place. The same person may be a son or daughter at home and a friend when sitting with a group of friends.

Run time Polymorphism:-

-> It is also known as Dynamic binding or late binding.

-> Function overriding is example of Run time Polymorphism when a child class declare a method which is already present in the parent class then it is called function overriding.

Ex:

```
class Base
{
    public:
        void display()
        {
            cout << " Base";
        }
};
class Derived : public Base
{
        public:
            void display()
            {
                cout " I'm derived"}
            }
};
main()
{
        Derived obj_B;
        Obj_B. display();
}
```

o/p => I'm derived.

Q5 Describe the use of this pointer with an example.

Ans

Every object in c++ has access to its own address through an important pointer called this pointer.

The this pointer is an implicit parameter to all member functions. Therefore inside a member function this may be used to refer to the invoking object.

Friend function do not have a this pointer because friends are not member of a class only member function have a this pointer

e.g:-

```
class Demo{
{
    Private:
        int num;
    Public: void setdata(int n)
            {
                this -> num = n;
            }
};
```

```
    void showdata()
    {
        cout << num;
    }
};

main()
{
    Demo obj;
    obj.setdata(100);
    obj.showdata();
}
```

o/p :-> 100

Q6. Explain following functions.

Ans (i) fseek() :-

It is used to move file pointer associated with a given file to a specific position.

Syntax:

fseek (File *pointer, long int offset, int position)

~~Pointer~~ =>

(ii) feof() :-

The function takes a file stream argument and returns an integer value which specifies if the end of file has been reached.

It returns non zero if the End has been reached, zero otherwise

Syntax:

feof (File * pointer)

(iii) fread():-

It reads the block of data from stream. This function first reads count number of object, each one with a size of size bytes from the given input streams.

- The total argument of bytes read if successful is (size * count)

Syntax:-

size_t fread (void *butter, sizet, size_t count, File * stream).

(iv) fopen():-

It takes a two argument and return a file stream associated with that files specified by the argument filename.

Syntax:-

File *fopen(const char * file name, const char * mode)

(v) fclose() :-

- It takes a single argument file stream which is to be closed. All the data that are buffered but not written are flused to the OS and all unread buffered data are discarded.

Syntax:

```
int fclose( file * stream);
```

Page No.
Date :

Explain namespace in detail.

A namespace is a declaration region that provides a scope to the identifiers inside it.

A namespace can be declared inside another namespace. It is also possible to declare a namespace as an alias of another namespace.

Userdefined namespace is declared using the namespace keyword.

Syntax :-

```
namespace namespace_name {
    // variables;
    // function();
}
```

eg.

```
namespace ns1 {
    int num;
    string name;
}
```

```
void main()
{
    cin >> ns1::num;
    cin >> ns1::name;

    cout << ns2::num;
    cout << ns1::name;
}
```

Q8 Explain I/O modes in files.

Ans

I/O modes in files:-

(i) ios :: in → file opens in input mode
(ii) ios :: out → file opens in output mode
(iii) ios :: app → file opens in append mode, we can add records at the end existing file.
(iv) ios :: ate → file is opend the file pointer move at the end of file. We can read and write anywhere in the file depending on other modes providing with mode.

(v) ios :: trunc :- when the file is opened, the contents are erased.

(vi) ios :: noreplace :- checks if the file exists, if file does not exist, the call to open fails.

(vii) ios :: nocreate :- check if file exists if file exists the call to open fails.

(viii) ios :: binary :- The file is opened in binary rather than default text mode

Q9 What are the advantages of storing the data in binary form?

Ans binary files are faster and easier for a program to read and write than the text files.

As long as the file doesn't need to be read by people or need to be ported to a different type of system binary files are the best way to store program information.

Advantages

(i) Stores the file in binary form

(ii) files can be either processed sequentially or randomly

(iii) no delimiter are used for a line.

(iv) no internal translation.

(v) It takes less space to store data. for example the integer value occupies 2 bytes in memory but as in text file it takes 4 bytes

Q10 Write Program to develop an applicat of college management System using all Obsect oriented concepts.

Ans

```
Class Institute
{
    protected:
        int member_id;
        String type;
        String ins_name;
        String member_name;

    public:

        Institute()
        {
            ins_name = "Rollwala";
        }

        void Add_member(){
            cin >> member_id;
            cin >> member_name;
        }

        void display(){

            cout << "Id:" <<
            member_id;
```

```
        cout << "Name : " << member_
             name;
        cout << "Type : " << type;
    }
};
class student : public Institute {
public:
    student()
    {
        member_type = "studen
    }
};
int main()
{
    Teacher
class Teacher : public Institute {
public:
    Teacher()
    {
        member_Type = "Teache
    }
};
```

```
int main()
{
    Teacher t[10];
    Student s[10];
    Institute i;

    int choice, choice2, i;
    static int count = 0;

    do {

        cout << " 1. Add member "
             << endl << " 2. Display"
        cout << endl << " 3. Exit ";

        cin >> choice;

        switch (choice)
        {
          case (1):
              cout << " 1. Add studen"
                   << endl << " 2. Add
                   teacher ";
              cin >> choice2;

              if (choice2 == 1)
              {
                  s[count].add member();
              }
              else
              {
                  t[count].add member();
```

```
        }
        count++;
        break;

      case(2):

        cout << "1. Display student"
             << endl << "2. Display
                Teacher";
        cin >> choice2;

        if (choice2 == 1)
        {
            for(i=0; i<count; i++)
            {
                s[i].display();
            }
        }
        else
        {
            for(i=0; i<count; i++)
            {
                t[i].display();
            }
        }
        break;

        case(3):
            exit(0);

        Default:
            cout << "Please enter correct
    }
}
```

Q11. Explain STL in detail

Ans

- STL stands for standard template library
- It is a collection of generic software components and generic algorithms glued by objects called iterators.
- STL function are known as generic algorithms.
- STL contains so many useful algorithms such as find(), sort() etc.

* Components of STL

1. Generic Containers.
2. Generic Algorithms.
3. Iterators.