Assignment – 2

Course code: CAP 444

Course Title: C++

Section : D2112
Roll No : RD2112A103
Reg. No: 12111670
Name : Jayshri Lal Pandit

[illegible]

(1.) Define pure vitual functions. write a c++ program to illustrate pure virtual functions.

Ans:→ A pure virtual function is a function virtual function in C++ for which we need not to write any function definition and only we have to declare it. It is declared by assigning 0 (zero) in the declaration.

A pure virtual function is "do nothing" function. It can be considered as an empty function means that the ~~virtual~~ pure virtual function does not have any definition relative to the base class. It just provides the template and derived class implements function.

Syntax :-

~~virtual void~~

```
virtual return_type function_name()=0;
```

```
/* Demo program to illustrate the used
   of pure virtual function */

#include <iostream>
using namespace std;

class shape // base class
{
```

```
public:
  virtual float calculateArea() =0; /* pure virtual
                                      function */
};
class Square : public Shape //Desired class 1
{
    float a ;
 public:
    Square (float l)   // paramatesized constructor
    {
        a = l;
    }
    float calcwateArea() //  function
    {
        return a*a ;
    }
};

class Circle : public Shape // Desired class 2
{
    float r;
 public:
    Circle (float x) //parametrized constructor
    {
        r = x;
    }
    float calculateArea () // function
    {
        return 3.14*r*r ;
    }
};
```

```
class Rectangle : public Shape // Desieved class 3
{
    float l;
    float b;
  public :
  Rectangle (float x , float y) // Parametrized const.
  {
        l = x;
        b = y;
  }
  float calculateArea ( ) // Function
  {
      return l*b ;
  }
};

int main ( )
{
    Shape *shape ;   /* Here object name small
                              shape written */
  Square s(3.4);
  Rectangle r (5,6);
  Circle c (7.8);

  Shape = &S ;
  int a1 = shape -> calculateArea ();
  shape = &r ;
  int a2 = Shape. -> calculateArea ();
  shape = &C;
  int a3 = Shape -> calculateArea ();
```

16/11/2021 21:25

```
std:: cout << "Area of the square is" << a1
                                          << std::endl;
std:: cout << "Area of the rectangle is "<< a2
                                          << std:: endl;
std:: cout << "Area of the circle is" << a3
                                          << std:: endl;
 return 0;
}
```

Output :-

```
Area of the square is 11
Area of the rectangle is 30
Area of the circle is 191
```

(2.) Discuss with example the following:-

(a) how virtual functions are hierarchical
(b) different Stream classes to use various file modes.

Ans:- (a) virtual function is a member function in the base class that redefine in a desired class. It is declared by using 'virtual' keyword. It is used to tell the compiler to perform late binding on the function. When an object of desiered class

~~that function then~~ call that function ~~by using~~ then instead of base class derived class function is called.

Virtual functions are hierachical in C++. This means that when a derived class fails to override a virtual function. In this case the ~~first redefinition found in reverse order and used it~~. base class function is used.

*Demo program to illustrate virtual functions are hierarchical */

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

class base
{
    public:
      virtual void fun():
      {
         cout<<" This base class virtual function."<<endl;
      }
};
class derived1 : public base
{
  public:
```

```
void fun ()
  {
  cout<<"This derived1 class virtual function."<<endl;

  }

};
class derived2 : public base
{
   public:

   /* Here, virtual function are not overridden by
      derived2class and base class virtual function
      is used. */
};
int main ()
{
   base *p , b;
   derived1 d1;
   derived2 d2;

   P = &b;
   P-> fun(); //call base class virtual function.

   P = &d1;
   P-> fun(); //call derived class virtual function.

   P = &d2;
   P-> fun(); // use base class virtual function.
```

16/11/2021 21:25

```
return 0;
}
```

The output of this program is:-

```
This base class virtual function.
This derived1 class virtual function.
This base class virtual function.
```

(b) There three types of stream classes which uses various file modes. These are given below :-

(i) ofstream :- It represent the output file stream and used to create files and write information to files.

(ii) ifstream :- represent input stream and used to read information from files.

(iii) fstream :- represent the both file stream and has capabilities to create files, write files and read information from files.

The various type of file modes are given below:-

Modes	Description
ios::app	Append mode. All output to that file to be appended to the end.
ios:: in	Open a file for reading.
ios:: out	Open a file for writing
ios:: binary	Open in binary mode.
ios:: ate	Open a file for output and move the read/write control to the end of the file.

~~#includ~~
/* Demo program to illustrate the different Stream classes to use various file modes */

```
#include< iostream>
#include< fstream>
using namespace std;
int main ()
{
    ofstream fout ;
    ifstream fin ;
    String Str, content ;
```

16/11/2021 21:26

```
fout.open ("d:/jay.txt", ios :: app);
cout <<"Enter a content : ";

getline (cin, content);
 fout << content <<"  ";

  fout.~~ep~~close ();
  fin. open ("d:/jay.txt", ios::in);

  while (getline(fin, str))
{

     cout<< str;
 }
 fin. close ();
  return 0;

}
```

Output

```
Enter a content : Jay     (suppose)
Jay
```

③ Write a program to create a file to store the data of mobile shop like mobile model, price, availability status. Implement the concept of random file access techniques.

Ans:-
```
#include <iostream>
#include <fstream>
using namespace std;

Class Mobile
{
  private:
    String mobileModel;
    float mobilePrice;
    String Status;

  public:
    void addmobileDetails()
    {
      ofstream fout;
      cout<<"Enter Mobile Model Name:";
      cin.ignore();
      getline(cin, mobileModel);

      cout<<"Enter Mobile Price:";
      cin >> mobilePrice;
      cout<<"Enter Availability Status:";
      cin>>Status;
      fout.open("d:/ mobileDetail.txt", ios::app);
      fout<<"Mobile Model Name is:"
          << mobileModel <<endl;

      fout<<"Mobile Price is:"
          << mobilePrice <<endl;
      fout<<"Availability Status is:"<<Status
                                       <<endl;
      fout.close();
```

```
cout << "Mobile details is added successfully"
     << endl;

}

void   getmobileDetails ()
{
       ifstream fin;
       String  str;
       fin. open("d:/mobileDetail.txt");
       cout<<" your Mobile  Detail is : "<<endl;

    while  (getline(fin,str))
    {
       ~~i~~  cout<<.str<<endl;

    }
    fin.close ();
}

int   main()
{
    Mobile  m;

     int choice;
  do
{ ~~{~~
   cout<<"Enter your choice :\n1. Add Mobile
                                        Details"
       <<"\n2. Get Mobile. Details."
<<"\n3. Exit." << endl;

cin>>choice;

switch (choice)
{
  case 1: m. addmobileDetails();
          break;

  case 2: m. getmobileDetails ();
          break;

  case 3 :
          exit (0);

  default:
          cout<<"Invalid choice" endl;
          break;

};

}while(choice != 0);

return 0;

}
```

Output

```
1. Add Mobile Details:
2. Get Mobile Details.
3. Exit.
```