SECTION ONE (COMPULSORY)

Question #1 [30 Marks]

a) Describe the main characteristics of object-oriented programming. {5 Marks}

Encapsulation – the ability to define a new type and a set of operations on that type, withoutrevealing the representation of the type.

• Inheritance – the ability to create new types that inherit properties from existing types.

• Polymorphism – the ability to use one name for two or more related but technically differentpurposes; "one interface, multiple methods.

What does it mean by information hiding? What are the advantages of it? {5 Marks}

In computer science, information hiding is the principle of segregation of the design decisions in a computer program that are most likely to change, thus protecting other parts of the program from extensive modification if the design decision is changed.

b) i) Define inheritance {2 Marks}

Ability to derive new objects from old-permits objects of a more specific class to inherit the properties (data) and behaviors (functions) of a more general class -ability to define a hierarchical relationship between objects

ii) Explain the importance of inheritance in generic programming {3 Marks}

c) i) Define an abstract class. {2 Marks}

An abstract class cannot be instantiated. An abstract class is a class that is declared `abstract`—it may or may not include abstract methods. Abstract classes cannot be instantiated, but they can be subclassed.

ii) Explain the importance of abstract class. {3 Marks}

The purpose of an abstract class is to provide a common definition of a base class that multiple derived classes can share.

e) i) With an example, write a syntax of one class inheriting another class{3 Marks}

```
classDerivedClassName(BaseClassName):
<statement-1>
    .
    .
    .
<statement-N>
```

ii) Whatmakesavirtualfunctionspecialandinterestingisthatitsupportssocalledrun-time polymorphism.That is, when a base class pointer (or a base class reference) points to a derived object that contains avirtual function and that virtual function is called by using that pointer, C++ determines which versionof that function to call based upon the type of the object that is pointed to. Using the sample code below explain how this works. {7 Marks}

```
.#include <iostream>

using namespace std;

class base {

public:

virtual void print() const { cout<< "base" <<endl; }

};

class sub : public base {

public:

void print() const { cout<< "sub" <<endl; }

};

classsubsub : public sub {

public:

void print() const { cout<< "subsub" <<endl; }

};

int main() {

base b; sub s; subsubss;

b.print(); s.print(); ss.print();

base *p[3];

p[0] = &ss; p[1] = &s; p[2] = &b;
```

```
for (int i = 0; i < 3; i++) p[i]->print();

return 0;

#include <iostream>

using namespace std;

class base {

public:

virtual void print() const { cout<< "base" <<endl; } // virtual !

};

class sub : public base {

public:

void print() const { cout<< "sub" <<endl; }

};

classsubsub : public sub {

public:

void print() const { cout<< "subsub" <<endl; }

};

int main() {

base b; sub s; subsubss;

b.print(); s.print(); ss.print(); // "base", "sub", "subsub"

base *p[3]; // pointers to base class

p[0] = &ss; p[1] = &s; p[2] = &b;

for (int i = 0; i < 3; i++) p[i]->print(); // "subsub", "sub", "base"

return 0;
```

}

SECTION TWO (ANSWER ANY TWO QUESTIONS)

Question #2 [20 Marks]

a) i) Define polymorphism as used in generic programming {2 Marks}

the ability to use one name for two or more related but technically different purposes; "one interface, multiple methods."how objects respond to certain kinds of messages

-ability for different objects to interpret functions differently

ii) Explain the importance of polymorphism as used in generic programming {3 Marks}

Polymorphism *is a very important concept in* ***object oriented programming*** *which enables to change the behavior of the applications in the run time based on the object on which the invocation happens. This enables to easily change the system without making much changes to the application.*

b) Define a Point structure with x and y coordinates. Write a function pDistance(Point, Point)that computes the distance between two points. Write a function pathLength(Point[], int) that takes anarray of Point (representing a path) and its size as input parameters and compute the path length by using the pDistance function. Write a main program to test your structure and functions. {15 marks}

Question #3[20 Marks]

a) Define the term encapsulation {2 marks}

the process of compartmentalizing the elements of an abstraction that constitute its structure and behavior; encapsulation serves to separate the contractual interface of an abstraction and its implementation.

b) Explain the importance of encapsulation as used in generic programming. {3 Marks}

The purpose is to achieve potential for change: the internal mechanisms of the component can be improved without impact on other components, or the component can be replaced with a different one that supports the same public interface. Encapsulation also protects the integrity of the component, by preventing users from setting the internal data of the component into an invalid or inconsistent state.

c) Exception handling can be a useful tool when developing and debugging a program. The three commands used for it are try, throw and catch. Write a program to clearly demonstrate use of these commands. {15 Marks}

```
#include <iostream>
#include <string>
using namespace std;
int main () {
try {
double *m;
m = new double[10];
if (m == NULL) throw string("Allocation failure");
for (int i = 0; i < 100; i++) {
if (i > 9) throw i;
m[i] = i * i;
}
// etc
delete [] m;
}
catch (int n) {
cout<< "Exception: index " << n << " is out of range." <<endl;
return 1;
}
catch (string &s) {
cout<< "Exception: " << s <<endl;
return 2;
}
return 0;
}
```

Question #4[20 Marks]

a) Define the term copy constructor. {2 Marks}

The copy constructor is a constructor which creates an object by initializing it with an object of the same class, which has been created previously.

b) Explain the importance of constructors. {3 Marks}

The copy constructor is used to:

- ***Initialize one object from another of the same type.***
- ***Copy an object to pass it as an argument to a function.***
- ***Copy an object to return it from a function.***

c) What would be the output of the following code: (5 Marks)

```
#include <iostream>
using namespace std;
voiddo_calc(int n1, int& n2);
int main(void) {
int num1 = 1;
int num2 = 2;
do_calc(num2,num1);
cout<< num2 << " " << num1 <<endl;
}
voiddo_calc(int n1, int& n2) {
n1 = n1 + 5;
n2 = n2 * 5;
cout<< n2 << " " << n1 <<endl;
}
```

d) With an example of a program demonstrate function overloading. {10 Marks}

Question #5[20 Marks]

a) With the help of a code show the difference between a class and an object {5 marks}

A class is a datatype, an object is an instance of a datatype.

b) Explain the difference between private, public and protected as used in generic programming {6 Marks}

They are access modifiers and help us implement Encapsulation (or information hiding). They tell the compiler which other classes should have access to the field or method being defined.

`Private` - Only the current class will have access to the field or method. `Protected` - Only the current class and subclasses (and sometimes also same-package classes) of this class will have access to the field or method.`Public` - Any class can refer to the field or call the method.

c) Virtual functions implement the "one interface, multiple methods" philosophy. With the help of a code demonstrate the use of virtual functions. {9 Marks}

```
#include <iostream>
using namespace std;
class base {
public:
virtual void print() const { cout<< "base" <<endl; } // virtual !
};
class sub : public base {
public:
void print() const { cout<< "sub" <<endl; }
};
classsubsub : public sub {
public:
void print() const { cout<< "subsub" <<endl; }
};
int main() {
base b; sub s; subsubss;
b.print(); s.print(); ss.print(); // "base", "sub", "subsub"
base *p[3]; // pointers to base class
p[0] = &ss; p[1] = &s; p[2] = &b;
for (int i = 0; i < 3; i++) p[i]->print(); // "subsub", "sub", "base"
return 0;
}
```