CS304-OPP
FINAL TERM SOLVED SUBJECTIVE
Prepared by JUNAID MALIK

Question No. 1:

Write the names of two types of Templates. MARKS: 2

Answer:- (Page 256)

a. Function Templates (in case we want to write general function like printArray) b. Class Templates (in case we want to write general class like Array class)

Question No. 2:

Describe the way to declare a template function as a friend of any class. 2marks

Answer:- (Page 294)

```
template< class T >
class B {
int data;
friend void doSomething( T ); // granting friendship to template
doSomething in // class B
friend A< T >; // granting friendship to class A in class B
};
```

Question No. 3:

Can a constructor throw an exception? How to handle the errors, when the constructor fails?

Answer:-

Yes, Constructor can throw an exception But In constructor if an exception is thrown than all objects created so far are destroyed. Exception due to constructor of any contained object or the constructor of a parent class can be caught in the member initialization list.

Question No. 4:

Why these operators can not be overloaded:

.,.*,?:,::

Answer:- (Page 141)

Reason: They take actual current object name, rather than value in their argument

Question No. 5:

What is graceful termination method of Error handling; give its example using C++.

Answer:- (Page 330)

Program can be designed in such a way that instead of abnormal termination, that causes the wastage of resources, program performs clean up tasks, mean we add check for expected errors (using if conditions),

Example – Graceful Termination

```
int Quotient (int a, int b ) {
if(b == 0){
cout << "Denominator can't " << " be zero" << endl;
// Do local clean up
exit(1);
}
return a / b;
}
```

Question No. 6:

How can you describe Virtual destructor? Why Virtual destructor is used?

Answer:- (Page 234)

When delete operator is applied to a base class pointer, base class destructor is called regardless of the object type. Only base class part of object will be deleted other parts will remain as it is this result in memory leak (wastage of memory), Virtual Destructors make it possible that complete object is being deleted so there is no memory leak (waste of memory).

Question No. 7:

Motor car can be called a good candidate for being an object so, this ibject contains Encapsulation characteristic. You are required to apply the concept of Encapsulation on "Motor Car".

Answer:-

Yes Motor Care is a good candidate for apply the concept of encapsulation
In Motor Car we have the following Objects which fully describes the phenomena of encapsulation.

1. Steer Wheels
2. Accelerate
3. Change Gear
4. Apply Brakes
5. Turn Lights On/Off

In encapsulation we just provide the user an interface and hide the Full implementation from the user because user don't want to know the implementation he Just want his work should be done perfectly

For example: Breaks of car, we gave just a paddle to the User and encapsulate how it works inside the engine. Same for accelerator, gear, steer wheels and turn on and off of Light.

Question No. 8:

Suppose base class and derived class has a member function with same signature. Now call that member function through the pointers to a base class object. Now, what determines that the base class member function or derived class function will be called?

Question No. 9:

What do you know about exception in initialization? Explain with example.

Answer:- (Page 345)

Exception in Initialization List

Exception due to constructor of any contained object or the constructor of a parent class can be caught in the member initialization list.

Example

```
Student::Student (String aName) : name(aName)
/*The constructor of String can throw a exception*/
{
...
}
```

Exception in Initialization List

The programmer may want to catch the exception and perform some action to rectify the problem

Example

```
Student::Student (String aName)
try : name(aName) {
...
}
catch (…) {
}
```

Question No. 10:

How can we set the default values for non type parameters?

Answer:- (Page 284)

We can set default value for this non type parameters, as we do for parameters passed in ordinary functions,

```
template< class T, int SIZE = 10 >
class Array
{ private:
T ptr[SIZE];
public:
void doSomething();
…
}
```

Question No. 11:

Explain the statement vector <int> ivec (4,3)

Answer:-

This would create a vector of four elements, each one initialized to 3. ivec looks like this:

3 3 3 3

Question No. 12:

Tell the procedure of making a class abstract in C++ by giving an example.

Answer:- (Page 230)

In C++, we can make a class abstract by making its function(s) pure virtual. Conversely, a class with no pure virtual function is a concrete class (which object can be instantiated)

A class having pure virtual function(s) becomes abstract

```
class Shape {
…
public:
virtual void draw() = 0;
}
```

Question No. 13:

Write down a list of four intangible objects?

Answer:-

1. Time
2. Light
3. inteligence
4. Temperature
5.software

Question No. 14:

Give the pseudo code of non case sensitive comparison function of string class.

Answer:- (Page 263)

```
int caseSencompare( char* str1, char* str2 )
{
for (int i = 0; i < strlen( str1 ) && i < strlen( str2 ); ++i)
if ( str1[i] != str2[i] )
return str1[i] - str2[i];
return strlen(str1) - strlen(str2);
}
```

Question No. 15:

Describe three properties necessary a container to implement generic algorithms.

Answer:- (Page 301)

We claimed that this algorithm is generic, because it works for any aggregate object (container) that defines following three operations

a. Increment operator (++) b. Dereferencing operator (*) c. Inequality operator (!=)

Question No. 16:

Mobile is a good entity for being an object. Write characteristics, behavior and unique ID.

Answer:-

Characteristics: Buttons, Battery, Screen
Behavior: Calling, Sending Message, Internet, Multimedia
Unique ID: Nokia

Question No. 17:

What are the container requirements? Write details.

Answer:

As each container need to perform certain operations on the elements added to it like their copy while creating another instance of container or their comparison while performaing certain operation on them like their sorting , so the elements that we are going to add in containers should provide this kind of basic functionality.
Examples of these functionalities are given below,
When an element is inserted into a container, a copy of that element is made using,
, ❖ Copy Constructor ❖ Assignment Operator
So the elements need to be added to any container should provide copy and assignment functionality.

- Associative containers and many algorithms compare elements so the elements that are added to associative containers should have this functionality,

o Operator ==
o Operator <
c++ does not provide functionality of comparison operator (==) or less than operator (<) by itself so we have to provide this functionality by ourselves in element class if we want to use it in associative containers.

Question No. 18:
Resolve problems in overloading assignment operator in string class.
Answer:-
The problem in statement
str1 = str2 = str3 is,
str1=str2=str3 is resolved as:
str1.operator=(str2.operator=(str3))
Assignment operator is beiing called two times one for part str2 = str3 and then for str1 = (str2 = str3) as assignment operator is right associate so first str2=str3 will be executed, and str2 will become equal to str3, then first overloaded assignment operator execution result will be assigned to s1,
str1.operator=(str2.operator=(str3))

Question No. 19:
What is random-iterator? What relation b/w random iterator and vector 5 marks
Answer:-
Random Access Iterators:-
They have all the capabilities of bidirectional Iterators plus they can directly access any element of a container.

You can think of vectors as smart arrays. They manage storage allocation for you, expanding and contracting the size of the vector as you insert or erase data. You can use vectors much like arrays, accessing elements with the [] operator. Such random access is very fast with vectors. as we can access any element of vector using its index so we can use random access iterator.

It's also fast to add (or push) a new data item onto the end (the back) of the vector. When this happens, the vector's size is automatically increased to hold the new item.

Question No. 20:
Class complex {double real,ing public ..... }; overload the *= operator for the complex class by writing C++ code 3 marks

Answer:- (Page 153)

```
class Complex{
double real, img;
public:
Complex & operator+=(const Complex & rhs);
Complex & operator+=(count double & rhs);
...
};
```

Question No. 21:

suppose there is a class Doctor with data member. Name Department and Doctor. you are required to write default copy constructor as well as deep copy constructor in order to initialize object and explain the difference with the help of main() function. Only write up to constructor code and main() function .(5 marks)

Question No. 22: (Marks: 2)

Write three important features of virtual functions.

Answer:-

These are one of the core components of C++ and enable different specializations of a base class. An important feature of the virtual function mechanism is that it is truly dynamic with respect to the resolution of the correct method to be called.

Question No: 23 (Marks: 5)

There are some errors in the code given below, you have to Indicate the line no. with error/s
Give the reason for error/s
Correct the error/s.

```
1.  #include <iostream>
2.  #include <stdlib.h>

3.  using namespace std;
4.  template <typename T>
5.  class MyClass{
6.  public:
7.  MyClass(){
8.  cout<<"This is class1"<<endl;
9.  }
10. };
11. template <typename T>
12. class MyClass<int*>{
12. public:
13. MyClass(){
14. cout<<"This is class2"<<endl;
15. }
16. };
19.
20. int main(int argc, char *argv[])
21. {
22. MyClass<int> c1;
```

```
23. MyClass<int*> c2;
24. system("PAUSE");
25. return 0;
26. }
```

Answer: Correct Code

```
#include <iostream>
#include <stdlib.h>

using namespace std;
template <typename T>
class MyClass{
public:
MyClass(){
cout<<"This is class1"<<endl;
}
};

template <typename T>
class MyClasss{ // class MyClass<int*>{ templates we do not specify type in definition.
public:
MyClasss(){
cout<<"This is class2"<<endl;
}
};

int main(int argc, char *argv[])
{
MyClass<int> c1;
MyClasss<int> c2;*
system("PAUSE");
return 0;
}
```

Question No: 24 (Marks: 5)

What is random iterator? What is its relation with vectors?

Answer:-

Random Access Iterators:-

They have all the capabilities of bidirectional Iterators plus they can directly access any element of a container.

You can think of vectors as smart arrays. They manage storage allocation for you, expanding and contracting the size of the vector as you insert or erase data. You can use vectors much like arrays, accessing elements with the [] operator. Such random access is very fast with vectors. as we can access any element of vector using its index so we can use random access iterator

It's also fast to add (or push) a new data item onto the end (the back) of the vector. When this happens, the vector's size is automatically increased to hold the new item.

Question No. 25:

Create a vector array of length 8,and also initialize the elements of this array with values 0 1 2 3 4 5 6 7 (5 mark)

Question No. 26:

Suppose base class and derive class have a member function with same signature. Now call that member function through pointer to a base class object. Now what determines that the base class member function or derived class will be called? (mark 5)

Question No. 27:

Describe the salint feature of abstract class (mrk3)

Answer:- (Page 230)

Abstract class's objects cannot be instantiated they are used for inheriting interface and/or implementation, so that derived classes can give implementation of these concepts.
In C++, we can make a class abstract by making its function(s) pure virtual. Conversely, a class with no pure virtual function is a concrete class

Question No. 28:

Give the C++ code of template function to print the values of any type of array I int.this function will take 2 parameters one will be pointer and other will be will be size of array (mrk3)

Answer:- (Page 257)

```
template< typename T >
void printArray( T* array, int size )
{
for ( int i = 0; i < size; i++ )
cout << array[ i ] << “, ”; // here data type of array is T
}
```

Question No. 29:

Give the name of three operation that a corsor or itrator generally provide (mrk3)

Answer:- (Page 305, 309)

• T* first()
• T* beyond()
• T* next(T*) **What do you know about function Template?**

(Answer: Page 262)

Function templates are used when we want to have exactly identical operations on different data types in case of function templates we cannot change implementation from data type to data type however we can specialize implementation for a particular data type.

Question No. 30:

Give the basic difference between iterator and cursors?

Answer:- (Page 308)

cursors were external pointer that we accessing internal data of any container like vector, it is against the principle of data hiding as we can access any container data using cursors so it is not good programming practice to given access in container for the use of cursors (first, next, beyond methods) we have alternate to cursors in the form of Iterators which are that traverse a container without exposing its internal representation.

Question No. 31:

The least one advantage and Disadvantage of Template

Answer:- (Page 300)

Advantages:

Templates provide
• Reusability
• Writability

Disadvantages:
• Can consume memory if used without care.

Question No. 32:
define composition and give its example with coding
Answer:- (Page 53)
An object may be composed of other smaller objects, the relationship between the "part" objects and the "whole" object is known as Composition.
Example
Ali is made up of different body parts; They can't exist independent of Ali.

Question No. 33:
what are container classes? How many types of container classes are there?
Answer:- (Page 312)
Container is an object that contains a collection of data elements like we have studied before now we will study them in detail.
STL provides three kinds of containers,
1. Sequence Containers
2. Associative Containers
3. Container Adapters

Question No. 34:
what is virtual inheritance?
Answer:- (Page 253)
In virtual inheritance there is exactly one copy of the anonymous base class object

Question No. 35:
can a constructor throw exception? If it fails, how should this error be handled?
Answer:-
One-stage constructors should throw if they fail to fully initialize the object. If the object cannot be initialized, it must not be allowed to exist, so the constructor must throw.

Question No. 36:
define inheritance and give its example.
Answer:- (Page 29)
- Derived class inherits all the characteristics of the base class
- Besides inherited characteristics, derived class may have its own unique characteristics
- Major benefit of inheritance is reuse

Question No. 37:
what is constructor?
Answer:- (Page 74)
Constructor is used to **initialize** the objects of a class. Constructor is used to ensure that object is in well defined state at the time of creation.

Question No. 38:

define static and dynamic binding.

Answer:- (Page 227)

Static binding means that target function for a call is selected at compile time
Dynamic binding means that target function for a call is selected at run time

Question No: 39 (Marks: 3)

If we declare a function as friend of a template class Will it be a friend for a particular data type or for all data types of that class?

Question No. 40:

What is the difference (if any) between the two types of function declarations?
template function_declaration;
template function_declaration;

Answer:-

The format for declaring function templates with type parameters is:

template <class identifier> function_declaration;
template <typename identifier> function_declaration;

The only difference between both prototypes is the use of either the keyword class or the keyword typename. Its use is indistinct, since both expressions have exactly the same meaning and behave exactly the same way.

Question No. 41:

State any two reasons why the virtual methods can not be static?

Answer:-

The virtual method implies membership, so a virtual function cannot be a nonmember function. Nor can a virtual function be a static member, since a virtual function call relies on a specific object for determining which function to invoke. A virtual function declared in one class can be declared a friend in another class.

Question No. 42:

Give three advantages that Iterators provide over Cursors.

Answer: - (Page 311)

a. With Iterators more than one traversal can be pending on a single container
b. Iterators allow to change the traversal strategy without changing the aggregate object c.
They contribute towards data abstraction by emulating pointers

Question No. 43:

Consider the code given below explain what kind of association exists between class A and class B. Justify your answer as well.

```
class A{

private:
int a,b,c;

public:
```

```
....
};

class B{

private:
int d,e,f;
A obj1;

public:
....
};
```

Question No. 44:

Is it possible to have Virtual Constructor? Justify your answer.

Answer: -

There is nothing like Virtual Constructor. The Constructor cant be virtual as the constructor is a code which is responsible for creating a instance of a class and it cant be delegated to any other object by virtual keyword means.

Question No. 45:

Explain the difference between static member variables with Non-static member variables of a class with the help of example.

Question No: 46 (Marks: 2)

What is the purpose of template parameter?

Answer:- (page 263)

We can change behavior of a template using template parameter.

Question No: 47 (Marks: 3)

Describe in simple words how we can use template specialization to enforce case sensitive specialization in String class.

Question No: 48 (Marks: 3)

Can we use compiler generated default assignment operator in case our class is using dynamic memory? Justify your answer.

Answer:- (Page 197)

In case our class involves dynamic memory allocation we had to write assignment operator code by our self as we had to write the user defined code for copy constructor.

Question No: 49 (Marks: 3)

Give the names of three ways to handle errors in a program.

Answer:- (page 329)

a. Abnormal termination
b. Graceful termination
c. Return the illegal value

Question No. 50:

Give the name of two cases when you MUST use initialization list as opposed to assignment in constructors. 5

Answer:-

Both non-static const data members and reference data members cannot be assigned values; instead, you should use initialization list to initialize them.

Question No. 51:

In which situation do we need to implement Virtual inheritance? explain with an example (5 marks)

Answer:-

In multiple inheritance while solving diamond problem virtual inheritance need to implement. The solution of avoid this problem is virtual inheritance so that in multiple inheritance only one copy of base class is generated as shown below instead of two separate copies.

In virtual inheritance there is exactly one copy of the anonymous base class object

Example:

```
class Vehicle{
protected:
int weight;
};
class LandVehicle :
public virtual Vehicle{
};
class WaterVehicle :
public virtual Vehicle{
};
Example
class AmphibiousVehicle:
public LandVehicle,
public WaterVehicle{
public:
AmphibiousVehicle(){
weight = 10;
}
};
```


Question No: 52 (Marks: 1)

What is meant by direct base class?

Answer:- (page 208)

A direct base class is explicitly listed in a derived class's header with a colon (:)

class Child1:public Parent1 { // Here Parent1 is Direct Base Class of Child1

…

};

Question No. 53:

Give C++ code of template function to print the values of any type of array.. 3Marks

[Hint: this function will take teo parameters, one will be array pointer and other will be size of the array]

Question No. 54:

if iter is an iterator to a container. Write an expression that will have the value of the object pointed to by iterator, and will then cause iterator to point to the next element. (3marks)

Answer:-

*iter++

Question No. 55:

Describe three problems with multiple inheritance (3 marks)

Answer:- (Page 248)

- If more than one base class have a function with same signature then the child will have two copies of that function
- Calling such function will result in ambiguity

Question No. 56:

Name two types of template (2 Marks)

Answer:- (Page 256)

a. Function Templates (in case we want to write general function like printArray)
b. Class Templates (in case we want to write general class like Array class)

Question No. 57:

Sort data in the order in which compiler searches a function. Complete specialization, generic template, Partial specialization, Ordinary function. (2Marks)

Answer:- (Page 286)

a. First of all compiler looks for complete specialization
b. If it can not find any required complete specialization then it searches for some partial specialization
c. In the end it searches for some general template

Question No. 58:

Give the names of two types of containers basically known as first class containers.(2 Marks)

Answer:- (Page 317)

Sequence and associative containers are collectively referred to as the first-class containers

Question No: 59 (Marks: 2)

Give two uses of a destructor.

Answer:- (page 92)

1. Destructor is used to free memory that is allocated through dynamic allocation. We have to free memory allocated using new operator by over self in destructor otherwise it remain occupied even after our program ends.
2. Destructor is used to perform house keeping operations.

Question No: 60 (Marks: 5)

Consider the following class,

```
class Base
{
        char * p;
public:
        Base() { p = new char[10]; }

        ~Base() { delete [] p; }
```

```
};
class Derived : public Base
{
        char * q;
public:
        Derived() { q = new char[20]; }

        ~Derived() { delete [] q; }
};
void foo()
{
        Base* p = new Derived();

        delete p;
}
```

With this program, every time function foo is called, some memory will leak.
Explain why memory will leak. Also, explain how to fix this problem.

Question No: 61 (Marks: 1)

Write the syntax of declaring a pure virtual function in a class?

Answer:- (page 230)

```
class Shape {
…
public:
virtual void draw() = 0;
}
```

Question No: 62 (Marks: 2)

Explain two benefits of setter functions.

Answer:- (Page 67)

1- It minimize the changes to move the objects in inconsistent states

2- You can write checks in your setter functions to check the validity of data entered by the user, for example age functions to check to calculate the age from date entered.

Question No: 63 (Marks: 3)

Consider the code below,

```
template< typename T >
Class T1{     Public:
   T I;
   Protected:
   T j;
   Private: T
   k;
   Friend void Test();
    }
```

This code has a template class T1 with three members i,j and k and a friend function Test(), you have to describe which member/s of T1 will be available in function Test().

Answer:-
All of them (i, j, k) will be available in function Test().

Question No: 64 (Marks: 3)
What do you mean by Stack unwinding?
Answer:- (Page 336)
The flow control (the order in which code statements and function calls are made) as a result of throw statement is referred as "stack unwinding"

Question No: 65 (Marks: 3)
Give the c++ code of case sensitive comparison function of string class.
Answer:- (Page 265)

```
class CaseSenCmp {
public:
static int isEqual( char x, char y ) {
return x == y;
}
};
class NonCaseSenCmp {
public:
static int isEqual( char x, char y ) {
return toupper(x) == toupper(y);
}
};
template< typename C >
int compare( char* str1, char* str2 )
{
for (int i = 0; i < strlen( str1 ) && i < strlen( str2 ); i++)
if ( !C::isEqual (str1[i], str2[i]) )
return str1[i] - str2[i];
return strlen(str1) - strlen(str2);
};
int main() {
int i, j;
char *x = "hello", *y = "HELLO";
i = compare< CaseSenCmp >(x, y);
j = compare< NonCaseSenCmp >(x, y);
cout << "Case Sensitive: " << i;
cout << "\nNon-Case Sensitive: "
<< j << endl; return
0;
}
```

Question No: 66 (Marks: 5)
Suppose the base class and the derived class each have a member function with the same signature. When you have a pointer to a base class object and call a function member through the pointer, discuss

what determines which function is actually called, the base class member function or the derived-class function.

Question No: 67 (Marks: 5)

What would be the output of this code?

```
class mother {
  public:
    mother ()
    { cout << "mother: no parameters\n"; } mother (int a)

{ cout << "mother: int parameter\n"; }
};
class daughter : public mother {
  public:

    daughter (int a)
      { cout << "daughter: int parameter\n\n"; }
};
class son : public mother {
  public:
    son (int a) : mother (a)
      { cout << "son: int parameter\n\n"; }
};
int main () {
  daughter rabia (0);
  son salman(0);

  return 0;
}
```

Answer:-

mother: no parameters

daughter: int parameter

mother: int parameter

son: int parameter

Question No: 68 (Marks: 5)

The code given below has one template function as a friend of a template class,

1. **You have to identify any error/s in this code and describe the reason for error/s.**
2. **Give the correct code after removing the error/s.**

```
template<typename U>void Test(U);
template< class T > class B {
        int data;
        public:
```

```
        friend void Test<>( T );
};
template<typename U>
void Test(U u){
   B < int> b1;
   b1.data = 7;


   }

   int main(int argc, char *argv[])
   {
        char i;
        Test(i);
        system("PAUSE");
        return 0;
   }
   Answer:-
   #include <cstdlib>
   template<typename U> void Test(U);
   template< class T > class B {
        int data;
        public:
        template <typename U >      friend void Test(U);  // this statement is missing
   };
   template<typename U>
   void Test(U u){
        B < int> b1;
        b1.data = 7;
   }
   int main(int argc, char *argv[])
   {
        char i;
        Test(i);
        system("PAUSE");
        return 0;
   }
   }

        b1.data = 7;
   }
   int main(int argc, char *argv[])
   {
        char i;
        Test(i);
        system("PAUSE");
        return 0;
   }
```

Question No: 69 (Marks: 3)

Tell the logical error/s in the code given below with reference to resource management; also describe how we can correct that error/s.

```
class Test{

public:
int function1(){
                try{
                    FILE *fileptr = fopen("filename.txt","w");
                     throw exception();
                     fclose(fileptr);
                     return 0;
                     }
                     catch(Exception e){
                     ...
                     }
}
};
```

Answer:- (Page 343)

In the above code, In case of exception the call to fclose will be ignored and file will remain opened. We can remove this issue in following ways

```
int function1(){
try{
FILE *fileptr = fopen("filename.txt","w");
fwrite("Hello World",1,11,fileptr);
...
throw exception();
fclose(fileptr);
} catch(...) {
fclose(fileptr); // adding fclose in catch handler as well
throw;
}
return 0;
}
```

Question No: 70 (Marks: 3)

What will be the output after executing the following code?

```
class c1{
public:
virtual void function(){
cout<<"I am in c1"<<endl;
}
```

```
};
class c2: public c1{
public:
void function(){
cout<<"I am in c2"<<endl;

}

};
class c3: public c1 {
public:
void function(){
cout<<"I am in c3"<<endl;
}

};

int main(){

c1 * test1 = new c2();
c1 * test2 = new c3();
test1->function();
test2->function();
system("PAUSE");
return 0;
}
```

Answer:-

am in c2
I am in c3

Question No: 71 (Marks: 2)

Q. Enlist the kinds of association w.r.t Cardinality (3)

Answer:- (page 51)

With respect to cardinality association has the following types,

a. Binary Association
b. Ternary Association
c. N-ary Association

Question No: 72 (Marks: 2)

State any conflict that may rise due to multiple inheritance?

Answer :-(page 248)

If more than one base class has a function with same signature then the child will have two copies of that function.Calling such function will result in ambiguity.

Question No: 73 (Marks: 3)

Give the differences between virtual inheritance and multiple inheritances.

Answer:- (Page 248, 253)

In Multiple Inheritance, If more than one base class have a function with same signature then the child will have two copies of that function; Calling such function will result in ambiguity But In virtual inheritance there is exactly one copy of the anonymous base class object.

Question No. 74:

Fill in the blanks below with public, protected or private keyword.

a. Public members of base class are _____ members of derived class
b. Protected members of base class are _____members of derived class.

Answer:-
a. Public members of base class are _____ **public** _____ members of derived class
b. Protected members of base class are _____ **protected** _____members of derived class.

Question No: 75 (Marks: 5)
See the 5 code snippets below and tell whether these are correct or incorrect also justify your answers in the table given at the end.

Snippet No.1

```
template< class T>
class A {
} ;
template< class T >
class B : public A< T* >
{ … }
```

Snippet No.2

```
template< >
{ … }
```

Snippet No.3

```
class B : public A< T* >
        { … }
```

Snippet No.4

```
        template< >
        class B< char* > : public A
        {…};
```

Snippet No.5

```
template< class T >
class B : public A< T* >
{ … }
```

Table:

Snippet	Is it correct or not (Correct/ Incorrect)	Justification of your answer No.
1		
2		
3		
4		
5		

Question No: 76 (Marks: 5)

What is the output produced by the following program?

```
#include<iostream.h>

void sample_function(double test) throw (int);

int main()
{
        try
        {
                cout <<"Trying.\n";
                sample_function(98.6);
                cout << "Trying after call.\n";
 }

        catch(int)
        {
                cout << "Catching.\n";
        }

        cout << "End program.\n";
        return 0;
}
void sample_function(double test) throw (int)
{
        cout << "Starting sample_function.\n";
        if(test < 100)
          throw 42;
}
```

Answer:-

Trying.
Starting sample_function.
Catching.
End program.


AL-JUNAID TECH INSTITUTE
0304-1659294
www.vulmshelp.com