Code : 051401

B.Tech 4th Semester Exam., 2018

OBJECT-ORIENTED PROGRAMMING WITH C++

Time : 3 hours *Full Marks* : 70

Instructions :

(i) All questions carry equal marks.

*(ii) There are **NINE** questions in this paper.*

*(iii) Attempt **FIVE** questions in all.*

*(iv) Question No. **1** is compulsory.*

1. Choose the correct answer (any *seven*) :

(a) Which of the following is true about this pointer?

(i) It is passed as a hidden argument to all function calls

(ii) It is passed as a hidden argument to all non-static function calls

(iii) It is passed as a hidden argument to all static functions

(iv) None of the above

(b) What is the use of this pointer?

(i) When local variable's name is same as member's name, we can access member using this pointer

(ii) To return reference to the calling object

(iii) Can be used for chained function calls on an object

(iv) All of the above

(c) Can we access this pointer from inside a static member function of a class?

(i) Yes

(ii) No

(d) Which of the following is true?

(i) Static methods cannot be overloaded

(ii) Static data members can only be accessed by static methods

(iii) Non-static data members can be accessed by static methods

(iv) Static methods can only access static members (data and methods)

(e) Predict the output of the following C++ program :

```
#include <iostream>
class Test {
public :
  void fun ( );
};
```

```
static void Test : : fun ( ) {
   std : : cout<<" fun ( ) is static\n";
}
int main ( ) {
  Test : : fun ( );
  return 0;
}
```

(i) fun ( ) is static

(ii) Empty screen

(iii) Compiler error

(iv) None of the above

(f) Predict the output of the following C++ program :

```
#include <iostream>
using namespace std;
class Base1 {
public :
 ~Base1 ( ) {cout << "Base1's
                    destructor" << end1;}
};
class Base2 {
public :
 ~Base2 ( ) {cout << "Base2's
                    destructor" << end1;}
};
class Derived : public Base1,
                          public Base2 {
public :
```



```
~Derived ( ) {cout << "Derived's
                    destructor" << end1;}
};
Int main ( ) {
  Derived d;
  return 0;
}
```

(i) Base1's destructor
Base2's destructor
Derived's destructor

(ii) Derived's destructor
Base2's destructor
Base1's destructor

(iii) Derived's destructor

(iv) Compiler Dependent

(g) Predict the output of the following C++ program :

```
#include <iostream>
using namespace std;
class base {
  int arr [10];
};
class b1 : public base { };
class b2 : public base { };
class derived : public b1, public b2 { };
int main ( ) {
  cout << sizeof (derived);
```

```
    return 0;
  }
```

(ii) 40

(ii) 80

(iii) 0

(iv) 4

(h) Predict the output of the following C++ program :

```
#include <iostream>
using namespace std;
class Base {};
class Derived : public Base {};
int main () {
  Base *bp = new Derived;
  Derived *dp = new Base;
}
```

(i) No Compiler Error

(ii) Compiler Error in line "Base *bp = new Derived;"

(iii) Compiler Error in line "Derived *dp = new Base;"

(iv) Runtime Error

(i) Predict the output of the following C++ program :

```
#include <iostream>
using namespace std;
int main () {
```



```
  const int x;
   x = 10
   cout << x;
   return 0;
}
```

(i) Compiler Error

(ii) 10

(iii) 0

(iv) Runtime Error

(j) Which of the following are true about templates?

1. Template is a feature of C++ that allows us to write one code for different data types.
2. We can write one function that can be used for all data types including user-defined types. Like sort( ), max( ), min( ),..etc.
3. We can write one class or struct that can be used for all data types including user-defined types. Like Linked List, Stack, Queue etc.
4. Template is an example of compile time polymorphism.

(i) 1 and 2

(ii) 1, 2 and 3

(iii) 1, 2 and 4

(iv) 1, 2, 3 and 4

2. What is pure virtual function? How is it different from a virtual function? Write a C++ program that prints "I'm Virtual" from inside a member function of a subclass, overriding a pure virtual function.

3. List out the decision constructs available in C++. Is it possible to skip the last 'else' clause in an if-else or in an else-if ladder construct? Is it possible to skip the 'default :' clause in switch construct?

4. Write a C++ class T which contains a const integer field n. T should also have constructor (s) which initialise n to an integer argument passed as a parameter or to zero if no argument is given; T should also have a destructor. The constructor (s) and destructor should print the value of the n field of the object being constructed or destructed. Indicate why, or why not, any of your fields or methods are qualified with virtual. Explain how objects of class T are allocated and deallocated, for each of the three areas : heap, stack and static store, noting one case where appropriate use of virtual is essential.

5. With an example, explain the order of invocation of constructors and destructors in multiple inheritance.

6. Differentiate between function overloading and function templates. Can we overload a function template? Illustrate with an example.

7. Differentiate between class and structure. With an example, explain the syntax for defining a class.

8. Can you create an instance of an abstract class? Write a C++ program to demonstrate this scenario.

9. Write a C++ program to find a substring inside a string. Write a user-defined exception class to throw an exception when a string has 'queen' as its substring.

★ ★ ★