JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

TEST -2 EXAMINATION - 2024

B.Tech-IV Semester (BI)

COURSE CODE (CREDITS): 18B11CI415 (3) MAX. MARKS: 25

COURSE NAME: OBJECT ORIENTED PROGRAMMING

COURSE INSTRUCTORS: Dr Emjee Puthooran MAX. TIME: 1 Hour 30 Minutes

---

Note: ***(a)****All questions are compulsory.*

(b)Marks are indicated against each question in square brackets.

(c) The candidate is allowed to make Suitable numeric assumptions wherever required for solving problems

---

Q1. What is the significance of a friend function in C++? Provide an example. [CO2, 2M]

Q2. Elaborate on the use of static member functions in C++ classes. How are they different from regular member functions? Provide an example illustrating their usage. [CO2, 2M]

Q3. Discuss the role of constructors and destructors in object-oriented programming with suitable examples. [CO2, 3M]

Q4. Differentiate between early binding and late binding in C++. Discuss how virtual functions contribute to late binding and how it differs from early binding in terms of execution. [CO3, 3M]

Q5. Define a base class Employee with protected members name and salary. Derive two classes, Manager and Staff, from Employee. Implement a virtual function displayDetails() in the base class and override it in both derived classes to display the details of a manager and a staff. [CO3, 5M]

Q6. Define a class Directory with members: name and phone number. Use the class object to store each set of data into a text file "directory.txt". The names contain only one word and the names and telephone numbers are separated by white spaces. Write a program to read the file and output the list in two columns. [CO4, 5M]

[P. T. O]

Q7. Find the output of the following C++ code snippet. Assume that the code snippet appear inside the `main()` function wherever not given and the header file '`iostream`' is included in the program. [CO3, 5M]

(a)
```
int x = 10;
int y = ++x + 2;
cout << x << " " << y;
```

(b)
```
int num;
num == 5;
cout << num;
```

(c)
```
class X {
public:
  X(){ cout<< "X"; }
 ~X(){ cout<< "~X"; } };
class Y : public X {
public:
  Y(){ cout<< "Y"; }
 ~Y(){cout<< "~Y"; } };
int main() {
    Y y;
    return 0; }
```

(d)
```
class base {
public:
  void hello(){cout<<"base"; } };
class derived: private base {};
int main() {
  derived d;
  d.hello();
  return 0; }
```

(e)
```
class Test {
private:
  int x, y;
public:
  Test():y(10),x(y+10) {}
  void print() {cout << x << ", "<<y; } };
int main() {
  Test *a = new Test();
  a->print();
  delete a;
  return 0; }
```