**Roll No…………….. Total No. of Pages:……**

**ST-2 (SET-II)**

**4th SEMESTER 2022-23**

**22CS006- Object Oriented Programming**

**Time allowed: 90 Minutes Max. Marks: 40**

**General Instructions:**

* **Follow the instructions given in each section.**
* **Make sure that you attempt the questions in order.**

**SECTION-A (10\*1 mark=10 marks)**

***(All questions are compulsory)***

1. Which keyword is used to create an instance of a class on the heap?
   1. **new**
   2. malloc
   3. create
   4. heap
2. What happens if a derived class has its own constructor, but the base class doesn't have a default constructor?
   1. It will result in a compile-time error.
   2. The program will crash at runtime.
   3. **The derived class constructor will implicitly call the base class's default constructor.**
   4. The derived class cannot have its own constructor.
3. When does dynamic binding (runtime polymorphism) occur in C++?
   1. During compile-time
   2. During linking
   3. **During runtime**
   4. During preprocessing
4. Which of the following operator cannot be overloaded?
5. +
6. **?:**
7. –
8. %
9. What is operator overloading in C++?
   1. Overriding the operator meaning by the user defined meaning for user defined data type
   2. Redefining the way operator works for user defined types
   3. Ability to provide the operators with some special meaning for user defined data type
   4. **All of the mentioned**
10. What is the function of the copy constructor in C++?
    1. It is used to create a copy of a derived class.
    2. It is used to create a copy of a base class.
    3. **It is used to create a copy of an object.**
    4. It is used to create a copy of a constructor.
11. Can a destructor be explicitly called for an object in C++?
    1. Yes
    2. **No**
    3. Only for dynamically allocated objects
    4. Only for statically allocated objects
12. Which operator is used to access the address of a class object?
    1. \*
    2. **&**
    3. -
    4. |
13. What is the purpose of a constructor with default arguments?
    1. It is used to provide multiple constructors for a class.
    2. It is used to deallocate memory.
    3. **It is used to create objects without specifying all constructor parameters.**
    4. It is used to initialize static class members.
14. What is the order of execution of constructors and destructors in a derived class?
    1. Destructor, Base Class Constructor, Derived Class Constructor
    2. **Base Class Constructor, Derived Class Constructor, Destructor**
    3. Derived Class Constructor, Base Class Constructor, Destructor
    4. Base Class Constructor, Destructor, Derived Class Constructor

**SECTION-B (5\*2 mark=10 marks)**

***(All questions are compulsory)***

1. What will be the output of the following C++ code?

#include <iostream>

#include <string>

using namespace std;

class complex

{

int i;

int j;

public:

complex(int a, int b)

{

i = a;

j = b;

}

complex operator+(complex c)

{

complex temp;

temp.i = this->i + c.i;

temp.j = this->j + c.j;

return temp;

}

void show(){

cout<<"Complex Number: "<<i<<" + i"<<j<<endl;

}

};

int main(int argc, char const \*argv[])

{

complex c1(1,2);

complex c2(3,4);

complex c3 = c1 + c2;

c3.show();

return 0;

}

* 1. 4 + i6
  2. 2 + i2
  3. **Error**
  4. Segmentation fault

1. Which of the followings are true about constructors?
   * 1. A class can have more than one constructor.
     2. They can be inherited.
     3. Their address can be referred.
     4. Constructors cannot be declared in protected section of the class.
     5. Constructors cannot return values.
     6. Only 1,2,4
     7. 1,2,4,5
     8. 1,3,5
     9. **1,4,5**
2. In case of operator overloading, operator function must be \_\_\_\_\_\_ .
3. Static member functions
4. Non- static member functions
5. Friend Functions
   * + 1. Only 2
       2. Only 1, 3
       3. **Only 2 , 3**
       4. All 1 , 2, 3
6. What is the output of the following code?

#include <iostream>

using namespace std;

class Base {

public:

Base() { cout << "Base constructor" << endl; }

virtual ~Base() { cout << "Base destructor" << endl; }

};

class Derived : public Base {

public:

Derived() { cout << "Derived constructor" << endl; }

~Derived() { cout << "Derived destructor" << endl; }

};

int main() {

Base\* ptr = new Derived();

delete ptr;

return 0;

}

* + - 1. **Base constructor, Derived constructor, Derived destructor, Base destructor**
      2. Derived constructor, Base constructor, Base destructor, Derived destructor
      3. Base constructor, Derived constructor, Base destructor
      4. Derived constructor, Base destructor

1. Which of the following statement is true?
   * + - 1. In Procedural programming languages, all function calls are resolved at compile-time
         2. In Object Oriented programming languages, all function calls are resolved at compile-time
2. **I only**
3. II only
4. Both I and II
5. Neither I nor II

**SECTION-C(Coding Question) (2x5 marks=5 marks)**

Q16) Imagine you are a software developer working on a project for a travel agency.

The agency wants to offer a new feature that allows users to calculate the distance between two cities using their

geographical coordinates.

As the lead programmer, you decide to create a C++ program to accomplish this task.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | p1(2.0, 3.0)  p2(5.0, 7.0) | p1(1.5, 3.2)  p2(5.1, 7.2) | p1(2.3, 1.2)  p2(3.1, 4.2) |
| **Output** | Distance between two points: 5 | Distance between two points: 5.38145 | Distance between two points: 3.10483 |

Solution :

**#include <iostream>**

**#include <cmath>**

**class Point {**

**private:**

**double x;**

**double y;**

**public:**

**Point(double x, double y) : x(x), y(y) {}**

**friend double calculateDistance(const Point& p1, const Point& p2);**

**};**

**double calculateDistance(const Point& p1, const Point& p2) {**

**double dx = p2.x - p1.x;**

**double dy = p2.y - p1.y;**

**return std::sqrt(dx \* dx + dy \* dy);**

**}**

**int main() {**

**const Point p1(2.0, 3.0);**

**const Point p2(5.0, 7.0);**

**double distance = calculateDistance(p1, p2);**

**std::cout << "Distance between two points: " << distance << std::endl;**

**return 0;**

**}**

Q17) You are a software developer working on an educational application.

The application allows teachers to store and manage student scores for various assessments.

To achieve this, you decide to create a class called "DynamicArray" that will dynamically store an array of integers representing the scores of students in a particular assessment.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | Enter size of the array: 6  Enter array elements: 34 67 43 55 60 64 | Enter size of the array: 5  Enter array elements: 98 77 53 55 69 | Enter size of the array: 5  Enter array elements: 68 67 92 65 69 |
| **Output** | Array elements: 34 67 43 55 60 64 | Array elements: 98 77 53 55 69 | Array elements: 68 67 92 65 69 |

Solution :

**#include <iostream>**

**class DynamicArray {**

**private:**

**int\* arr;**

**int size;**

**public:**

**// Constructor with dynamic memory allocation**

**DynamicArray(int arrSize) {**

**size = arrSize;**

**arr = new int[size];**

**std::cout<<"Enter array elements: ";**

**for (int i = 0; i < size; i++) {**

**int k;**

**std::cin>>k;**

**arr[i] = k;**

**}**

**}**

**void displayArray() {**

**std::cout << "Array elements: ";**

**for (int i = 0; i < size; i++) {**

**std::cout << arr[i] << " ";**

**}**

**std::cout << std::endl;**

**}**

**// Destructor to free the dynamically allocated memory**

**~DynamicArray() {**

**delete[] arr;**

**}**

**};**

**int main() {**

**int s;**

**std::cout<<"Enter size of the array: ";**

**std::cin>>s;**

**DynamicArray arrObj(s);**

**arrObj.displayArray();**

**return 0;**

**}**

**SECTION-D (Coding Question)(1x10 mark=10 mark)**

Q18) You are working on a navigation software development project for a cutting-edge autonomous vehicle.

The vehicle's system relies on advanced algorithms to calculate and compare points in 2D space.

As part of the development, you need to create a class that represents a 2D point and overload the > operator to compare two points based on their distance from the origin (0, 0).

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | p1(3, 4)  p2(5, 12) | p1(4, 6)  p2(5, 2) | p1(1, 1)  p2(-3, 4) |
| **Output** | (5, 12) is farther from the origin than (3, 4) | (4, 6) is farther from the origin than (5, 2) | (-3, 4) is farther from the origin than (1, 1) |

Solution :

**#include <iostream>**

**#include <cmath>**

**using namespace std;**

**class Point {**

**private:**

**double x;**

**double y;**

**public:**

**Point(double xx, double yy) : x(xx), y(yy) {}**

**// Overloading the '>' operator**

**bool operator>(const Point& other) {**

**double distance1 = sqrt(x \* x + y \* y);**

**double distance2 = sqrt(other.x \* other.x + other.y \* other.y);**

**return distance1 > distance2;**

**}**

**void display() {**

**cout << "(" << x << ", " << y << ")";**

**}**

**};**

**int main() {**

**Point p1(3, 4);**

**Point p2(5, 12);**

**if (p1 > p2) {**

**p1.display();**

**cout << " is farther from the origin than ";**

**p2.display();**

**} else {**

**p2.display();**

**cout << " is farther from the origin than ";**

**p1.display();**

**}**

**cout << endl;**

**return 0;**

**}**