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

**ST-3 (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. When a class is derived from a virtual base class, how many copies of the virtual base class's members will exist?
   1. **Only one copy, shared among all derived classes.**
   2. One copy for each derived class.
   3. Two copies, one for each base class and one for derived class.
   4. The number of copies depends on the inheritance type (public, private, protected).
2. Which keyword is used to declare a virtual function in C++?
   1. **virtual**
   2. override
   3. vfunc
   4. base
3. Early binding is performed at:
   1. **Compile-time**
   2. Link-time
   3. Runtime
   4. Execution time
4. Late binding is performed at:
   1. Compile-time
   2. Link-time
   3. **Runtime**
   4. Execution time
5. What does the std::exception class in C++ provide?
   1. **It is a base class for all C++ exceptions.**
   2. It is used to define custom exception classes.
   3. It is a built-in exception that cannot be derived from.
   4. It is used to terminate the program.
6. How do you handle an exception without catching it?
   1. by using the catch block with no arguments
   2. **by using the catch block with an ellipsis (...)**
   3. by using the catch block with a numeric value
   4. by using the catch block with a character value
7. Virtual functions are used to achieve which type of polymorphism?
   1. Compile-time polymorphism
   2. Static polymorphism
   3. **Run-time polymorphism**
   4. Overload polymorphism
8. In C++, what is the correct way to define a virtual destructor in a class?
   1. **virtual ~MyClass() {}**
   2. virtual MyClass::~MyClass() {}
   3. ~virtual MyClass() {}
   4. ~MyClass() virtual {}
9. What symbol is used to specify a template parameter in C++?
   1. \*
   2. $
   3. #
   4. **typename or class**
10. When are function templates instantiated in C++?
    1. **At compile time**
    2. At runtime
    3. When the program starts
    4. When the template is defined

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

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

1. What is the output of the following code?

#include <iostream>

class Base {

public:

virtual void print() const {

std::cout << "Base ";

}

};

class Derived : public Base {

public:

void print() const override {

std::cout << "Derived ";

}

};

void display(const Base& obj) {

obj.print();

}

int main() {

Base b;

Derived d;

display(b);

display(d);

return 0;

}

* 1. Base Base
  2. Derived Base
  3. **Base Derived**
  4. Derived Derived

1. What is the output of the following code?

#include <iostream>

class A {

public:

virtual void foo() {

std::cout << "A::foo()" << std::endl;

}

};

class B : public A {

public:

void foo() override {

std::cout << "B::foo()" << std::endl;

}

};

int main() {

A\* a\_ptr = new B;

B\* b\_ptr = dynamic\_cast<B\*>(a\_ptr);

if (b\_ptr) {

b\_ptr->foo();

} else {

std::cout << "Dynamic cast failed" << std::endl;

}

return 0;

}

* 1. A::foo()
  2. **B::foo()**
  3. Compiler error
  4. Dynamic cast failed

1. What is the output of the following code?

#include <iostream>

class Animal {

public:

virtual void speak() const { std::cout << "Animal sound\n"; }

};

class Dog : public Animal {

public:

void speak() const override { std::cout << "Bark!\n"; }

};

int main() {

Dog\* d = new Dog;

Animal\* a = d;

a->speak();

return 0;

}

* 1. Animal sound
  2. **Bark!**
  3. Compile error
  4. Runtime error

1. What is the output of this program?

#include <iostream>

using namespace std;

template <typename T>

T maximum(T x, T y)

{

return (x > y)? x : y;

}

int main()

{

cout << maximum(3, 7) << std::endl;

cout << maximum(3.0, 7.0) << std::endl;

cout << maximum(3, 7.0) << std::endl;

return 0;

}

1. **Compiler Error in last cout statement as call to maximum is ambiguous**
2. Compiler Error in all cout statements as data type is not specified
3. 7 7.0 7.0
4. None of the above
5. What is the output of the following code?

#include <iostream>

int main() {

try {

throw 10;

}

catch (int x) {

std::cout << "Caught an integer: " << x;

}

catch (...) {

std::cout << "Caught something else.";

}

return 0;

}

* 1. **Caught an integer: 10**
  2. Caught something else.
  3. The program will terminate.
  4. Compiler error

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

Q16) Imagine you are organizing a grand Multimedia Festival, where various artists and creators will be showcasing their creative works, including music and videos. To manage the multimedia playback during the festival, you decide to create a C++ program that efficiently handles the media players for different types of content such as audio or video.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | AudioPlayer("Song1.mp3") | VideoPlayer("Movie1.mp4") | AudioPlayer("zingat.mp3") |
| **Output** | Playing audio: Song1.mp3 | Playing video: Movie1.mp4 | Playing audio: zingat.mp3 |

Solution :

**#include <iostream>**

**#include <string>**

**class MediaPlayer {**

**protected:**

**std::string media;**

**public:**

**MediaPlayer(const std::string& m) : media(m) {}**

**virtual void play() {**

**std::cout << "Playing generic media." << std::endl;**

**}**

**};**

**class AudioPlayer : public MediaPlayer {**

**public:**

**AudioPlayer(const std::string& m) : MediaPlayer(m) {}**

**void play() override {**

**std::cout << "Playing audio: " << media << std::endl;**

**}**

**};**

**class VideoPlayer : public MediaPlayer {**

**public:**

**VideoPlayer(const std::string& m) : MediaPlayer(m) {}**

**void play() override {**

**std::cout << "Playing video: " << media << std::endl;**

**}**

**};**

**int main() {**

**MediaPlayer\* media1 = new AudioPlayer("Song1.mp3");**

**MediaPlayer\* media2 = new VideoPlayer("Movie1.mp4");**

**media1->play();**

**media2->play();**

**delete media1;**

**delete media2;**

**return 0;**

**}**

Q17) Write a C++ program that takes two floating-point numbers as input and attempts to calculate their difference.

Handle the underflow\_error exception if the result is too close to zero.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | Enter two numbers: 9 9 | Enter two numbers: 12 5 | Enter two numbers: 12 19 |
| **Output** | Exception: Underflow error! | Difference: 7 | Difference: -7 |

Solution :

**#include <iostream>**

**#include <cmath>**

**#include <limits>**

**using namespace std;**

**int main() {**

**double num1, num2, difference;**

**cout << "Enter two numbers: ";**

**cin >> num1 >> num2;**

**try {**

**difference = num1 - num2;**

**if (fabs(difference) < numeric\_limits<double>::epsilon()) {**

**throw underflow\_error("Underflow error!");**

**}**

**cout << "Difference: " << difference << endl;**

**} catch (const underflow\_error& e) {**

**cout << "Exception: " << e.what() << endl;**

**}**

**return 0;**

**}**

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

Q18) You are a talented software engineer working on an exciting project to develop an advanced scientific calculator targeted at students and academic professionals. As part of this ambitious undertaking, you have been assigned a crucial task: to create a flexible program that can efficiently calculate the power of a number.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Test Case 1** | **Test Case 2** | **Test Case 3** |
| **Input** | base=2, exponent=5 | base=3.14, exponent=2 | base=2, exponent=3 |
| **Output** | Power: 32 | Power: 9.8596 | Power: 8 |

Solution :

**#include <iostream>**

**using namespace std;**

**// Function template 'calculatePower' calculates the power of a number.**

**// The template takes two parameters: 'base', representing the base number of type 'T', and 'exponent', an integer representing the power.**

**// The function returns the result of 'base' raised to the power of 'exponent'.**

**template <typename T>**

**T calculatePower(T base, int exponent) {**

**T result = 1;**

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

**result \*= base;**

**}**

**return result;**

**}**

**int main() {**

**// Calculate the power of an integer (2) raised to the exponent (5).**

**cout << "Power: " << calculatePower<int>(2, 5) << endl; // Output: Power: 32**

**// Calculate the power of a double (3.14) raised to the exponent (2).**

**cout << "Power: " << calculatePower<double>(3.14, 2) << endl; // Output: Power: 9.8596**

**return 0;**

**}**