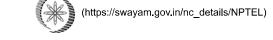
Χ



rg0102@srmist.edu.in ~



NPTEL (https://swayam.gov.in/explorer?ncCode=NPTEL) » Fundamentals of Object Oriented Programming (course)

Announcements (announcements) About the Course (preview) Q&A (forum) Progress (student/home) Mentor (student/mentor)

Review Assignment (assignment review) Course Recommendations New! (/course recommendations)

Click to register for **Certification exam** 

(https://examform.nptel.ac.]n/2025\_01/Weeksh4:Assignment 4

The due date for submitting this assignment has passed.

Due on 2025-02-19, 23:59 IST.

If already registered, click to check your payment status

Course outline

**About NPTEL ()** 

Assignment submitted on 2025-02-16, 13:43 IST

1) Which of the following best describes polymorphism in object-oriented programming?

A class having multiple constructors.

The ability of different objects to respond to the same function call in different ways.

A function having the same name as its class.

None of the above.

	How does an NPTEL online course work? ()
١	Week 0 ()
١	Week 1 ()
١	Week 2 ()
١	Week 3 ()
١	Week 4 ()
	Polymorphism (unit? unit=42&lesson=51)
á	Overloading: Operator and Constructor (unit? unit=42&lesson=52)
	this' keyword in C++ (unit?unit=42&lesson=53)
	Method Overloading (unit?unit=42&lesson=54)
	Method Overriding (unit? unit=42&lesson=55)
,	Quiz: Week 4: Assignment 4 (assessment?name=69)
	Solution for Week 4 (unit? unit=42&lesson=97)
١	Week 5 ()

Yes, the answer is correct. Score: 1	
Accepted Answers:  The ability of different objects to respond to the same function call in different ways.	
2) Which of the following is an example of static polymorphism?	1 point
Method overloading	
○ Virtual functions	
O Abstract classes	
Method overriding	
No, the answer is incorrect. Score: 0	
Accepted Answers:	
Method overloading	

```
Week 6 ()
Week 7 ()
Week 8 ()
Week 9 ()
Week 10 ()
Week 11 ()
Week 12 ()
Download Videos ()
Weekly Feedback ()
```

```
3) Consider the following code:
class Complex {
    int real, imag;
public:
    Complex(int r, int i) : real(r), imag(i) {}
    Complex operator+(const Complex& c) {
         return Complex(real + c.real, imag + c.imag);
    void display() {
         std::cout << real << " + " << imag << "i" << std::endl;
};
int main() {
    Complex c1(2, 3), c2(4, 5);
    Complex c3 = c1 + c2;
    c3.display();
    return 0;
}
What is the output of this program?
  6 + 8i
  6 + 15i
  8 + 8i
  Ompilation error
 Yes, the answer is correct.
```

Score: 1 Accepted Answers: 6 + 8i
<ul> <li>4) In a C++ program to overload the * operator for a class Matrix, where:</li> <li>The class stores a 2D matrix as a private member.</li> <li>The * operator multiplies two matrices.</li> <li>The result of multiplication is displayed in the console.</li> </ul>
Which of the following correctly implements the operator overloading?
The operator is defined inside the class.
The operator is defined as a friend function.
Both A and B are valid.
Operator overloading is not possible for matrix multiplication.
Yes, the answer is correct. Score: 1 Accepted Answers: Both A and B are valid.

```
5) Consider the following Java code:
class Calculator {
     int add(int a, int b) {
         return a + b;
     }
     double add(double a, double b) {
         return a + b;
public class Main {
    public static void main(String[] args) {
         Calculator calc = new Calculator();
         System.out.println(calc.add(2, 3));
         System.out.println(calc.add(2.5, 3.5));
     }
What is the output of this program?
  56.0
  5.0 6.0
  Compilation error
  55
 Yes, the answer is correct.
 Score: 1
 Accepted Answers:
 5 6.0
```

```
6) Which of the following demonstrates dynamic polymorphism?
class Base {
public:
     virtual void display() { std::cout << "Base class\n"; }</pre>
};
class Derived : public Base {
public:
     void display() override { std::cout << "Derived class\n"; }</pre>
};
int main() {
     Base* ptr;
     Derived obj;
     ptr = &obj;
     ptr->display();
     return 0;
}
What is the output of this program?
   Base class

    Derived class

   Compilation error
   Undefined behavior
 No, the answer is incorrect.
 Score: 0
 Accepted Answers:
  Derived class
```

7) Which of the following is true about virtual functions in C++?
They allow runtime polymorphism.
They must be redefined in the derived class.
They can be called on an object of the base class.
They cannot be used with pointers.
Yes, the answer is correct. Score: 1
Accepted Answers:
They allow runtime polymorphism.

```
8) Consider the following Java code:
class Animal {
    void sound() {
         System.out.println("Animal makes a sound");
    }
class Dog extends Animal {
    @Override
    void sound() {
         System.out.println("Dog barks");
public class Main {
    public static void main(String[] args) {
         Animal animal = new Dog();
         animal.sound();
What is the output of this program?
   Animal makes a sound
   Dog barks
  Compilation error
  Undefined behavior
 Yes, the answer is correct.
```

Score: 1 Accepted Answers:	
Dog barks	d maind
9) Which of the following is a limitation of static polymorphism?	1 point
It requires pointers.	
It is resolved at compile time and cannot adapt to runtime behavior.	
It can only be implemented in C++.	
It cannot be overloaded.	
Yes, the answer is correct. Score: 1	
Accepted Answers:	
It is resolved at compile time and cannot adapt to runtime behavior.	
<ul> <li>10) In a C++ program to demonstrate both static and dynamic polymorphism usi</li> <li>Method overloading for static polymorphism.</li> </ul>	ng the following: 1 point
Virtual functions for dynamic polymorphism.	
Which of the following correctly calls both overloaded and overridden methods?	
Overloaded methods are called directly, and overridden methods are called us	sing a
base class pointer.	
Overloaded methods are called using a base class pointer, and overridden me	ethods are
called directly.  Both methods are called directly.	
Both methods require pointers.	
Yes, the answer is correct. Score: 1	
Accepted Answers:	
Overloaded methods are called directly, and overridden methods are called using base class pointer.	ıy a
- r	