BACKEND ASSIGNMENT / VIBHAVRI A. SHRIVAS / OOPS CONCEPT

DATE : 1/12 /2022

Topic – OOPS Fundamentals

B1 Ans.	What Is class in Object Oriented Programming Language ? In object – oriented programming , a class is a template definition of the methods and variables in a particular kind of object. thus , an object is specific instance of a class it contains real values instead of variable.
B2 Ans.	What is an Object in Object Oriented Programming Language ? In object-oriented programming , object are the things you think about first in designing a program and they are also the units of code that are eventually derived from the process.
B3 Ans.	What Is Difference Between Class And Interface ? Write an interface is similar to writing a class. But a class describes the attributes and behaviors of an objects. And an interface contains behaviors that a class implements following are the important difference between class and interface.
B4 Ans.	What Is Method Overloading in Object Oriented Programming Language? Method overloading is a from of polymorphism in OOP. Polymorphism allows objects or methods to act in different ways, according to the means in which they are used. One such manner in which the method behave according to their arguments types and number of arguments is method overloading.
B5 Ans.	What Is Data hiding in Object Oriented Programming Language ? Data hiding is an object – oriented programming (OOP) technique specifically used to hide internal object details (i.e., data members). Data hiding guarantees exclusive data access to class member only and protects and maintains objects integrity by preventing intended or unintended change and intrusions.

B6	What are the differences between abstract classes and interfaces?
Ans.	The abstract class and interface both are use to have abstraction. An abstract class contain an abstract keyword on the declaration whereas an interface is a sketch that is used to implement a class.
B7	What are the Virtual Function in Object Oriented Programming ?
Ans.	A virtual function is a member function that you expect to be redefined in derived classes. When you refer to derive class object using a pointer or a reference to the base class , you can call a a virtual function for that object and executed the derive classes version of the function.
B8	What is Constructor in Object Oriented Programming ?
Ans.	In class – based , object – oriented programming , a constructor (abbreviation : ctor) is a special type of subroutine called to create an object. It prepares the new object for use , often accepting arguments that the constructor use to set required member variable.
B9	What is Abstract class in Object Oriented Programming?
Ans.	An abstract class is a class that contain at least one abstract method. An abstract method is a method that is declared , but not implemented in the code.
B10	What is Final Keyword in Object Oriented Programming?
Ans.	The final keyword is a non-access modifier used for classes, attributes and Method, which makes them non-changeable(impossible to inherit or override). The final keywords is useful when you want a variable to always Store the same value , like PI (3.14)the final keyword is called a “modifier”.
B11	What is Pure Virtual function in Object Oriented Programming?
Ans.	A pure virtual function or pure virtual method is a virtual function that Is required to be implemented by a derived class is not abstract. Classes Containing pure virtual method are termed “abstract” and they cannot be Instantiated directly.
B12	What are Sealed Modifier in Object Oriented Programming?
Ans.	When applied to a class, the sealed modifier prevents other classes from inheriting from it. In the following example , class B inherits from class A But no class can inherit from class B. you can also us the sealed modifier

	on a method or property that override a virtual method or property that override a virtual method or property in a base class.
B13 Ans.	What is Dynamic or run time Polymorphism in oops ? Run-time polymorphism: whenever an objects is bound with the functionally at run time ,this is known as runtime polymorphism can be achieved by method overriding. Java virtual machine determines the method to call at the runtime , not at the compile time.
B14 Ans.	What is Access Modifier in Object Oriented Programming ? Access modifiers (or access specifies) are keywords in object-oriented Language that set that accessibility of classes , and other members. Access Modifiers are a specific part of programming language syntax used of facilitate the encapsulation of components.
B15 Ans.	What is Friend Function in Object Oriented Programming? In object-oriented programming , a friend function , that is a “friend” of a given class , is a function that is given the same access as method to protect And protected data .a friend function is declared by the class that is granting access , so friend functions are part of the class interface , like methods.
B16 Ans.	What is Overriding in Object Oriented Programming? Method overriding , in object- oriented programming , is a language feature That allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its super classes or parent classes.
B17 Ans.	What is the role of mutable storage class specified ? The mutable storage class specifier is used only on a class data member to make it modifiable even though the member is part of an object declared As const. you cannot use the mutable specifier with names declared as static Or const , or reference members.

B18	Distinguish between shallow copy and deep copy ?
Ans.	In shallow copy, a copy of the original object is stored and only the reference address is finally copied. In deep copy , the copy of the original and the repetitive copied both are stored.
B19	what is a Reference variable in Object oriented Programming Language ?
Ans.	a reference variable is a variable that point to an object of a given class, letting you access the value of an object. An object is a compound data structure that holds values that you can manipulate. A reference variable does not store its own values.
B20	What is a Copy Constructor ?
Ans.	A copy constructor is a member function that initializes an object using another object of the same class. In simple terms , a constructor which creates an object by initializing it with an object of the same class, which has been created previously is know as a copy constructor.
B21	What is this Pointer in Object oriented Programming Language ?
Ans.	The this pointer is a pointer accessible only within the nonstatic member Function of a class, struct, or union type . it points to the object for which the member function is called. Static member function don't have a this pointer.
I1	Define Constructor and Destructor?
Ans.	Constructor is used to initialize an object of the class and assign value to data member corresponding to the class. While destructor is used to deal locate the memory of an object of a class. 5 there can be multiple constructor for the same class in a class , there is always a single destructor.
I2	How to Load Classes in Object Oriented Programming ?
Ans.	That depends on the exact "object – oriented" feature –set you want to have if you need stuff like overloading and/or virtual methods, you probably need to include function pointers in structure.
I3	How to Call Parent Constructor ?
Ans.	In order to run a parent constructor , a call parent : : _ constructor() within

	the child constructor is required. If the child does not define a constructor then it may be inherited from the parent class just like a normal class method (if it was declared as private).$obj=new othersubclass();.
I4 Ans.	Are Parent Constructor Called Implicitly When Create An Object Of Class? Parent constructor are not called implicitly if the child class define a constructor. In order to run a parent constructor , a call to parent ::_constructor ()within the child constructor is required.
I5 Ans.	What Happen, If Constructor Is Defined As Private Or Protected? If a constructor is declared as private , then its objects are only accessible from within the class declared class. You accept its objects from outside the constructor class.
I6 Ans.	Define New-style Constructor & Old-style Constructor.Check which One Will Be Called? What happen , if new – style constructor & old style constructor are defined. Which one will be called. But if new – style constructor will called but if new-style constructor is missing , old style constructor will called.

I7
 	
Ans.
	Create Abstract class and method?
	
	Abstract Class :-
	
	create an abstract class
	abstract class Language {
	fields and methods
	}
	...
	
	try to create an object Language
	throws an error
	Language obj = new Language();
	
	Abstract method :-
	
	abstract method
	public abstract void display1();
	
	non-abstract method
	public void display2() {
	Console.WriteLine("Non abstract method");
	}
	}

I8 Ans.	Define 3 types of visibility of data member & member function. Public : accessed by only member function of the class. Protected : similar to private , but accessed by all the member function of immediate derived class. Private : accessed by only member function of the class.

I9
 	
Ans.
	Create a method which will never inherited ?
	
	I have a base class of a geometric object which I use on its own but I also want to inherit the class into another one thats sort of an advanced version of the object since they share a lot of logic. The base object has several static creation methods (cannot use new due to argument conflicts), I don't want to inherit those. Can I specify somehow that those are not to be inherited.
	
	Example :
	
	struct Banana {
	float length;
	
	Banana() {}
	
	Banana(float length) {
	this->length = length;
	}
	
	static Banana CreateByHalfLength(float halfLength) {
	return Banana(halfLength * 2);
	}
	};
	struct AdvancedBanana : Banana {
	float bendAmt;
	
	AdvancedBanana(float length, float bendAmt) {
	this->length = length; this->bendAmt = bendAmt;
	}
	};

I10	What is the difference between Abstract class and Interface?
Ans.	The key technical difference between an abstract class and an interface are : abstract classes can have constant , members , method stubs (method without a body) and defined methods, whereas interface can only have constant and method stubs.
I11	Create parent class for car and child class for car_model and use car functionality in car_model class ?.

Ans.	public partial class Form1 : Form
	{
	// Declaring the class
	private Car myCar;
	
	public Form1()
	{
	InitializeComponent();
	}
	
	// GetCarData method that accepts Car object as an argument
	private void GetCarData()
	{
	// Creating the car object
	Car myCar = new Car();
	
	try
	{
	// Get the car year and model
	myCar.year = int.Parse(yearTextBox.Text);
	
	myCar.make = makeTextBox.Text;
	
	myCar.speed = 0;
	
	}
	catch (Exception ex)
	{
	// Display error message
	MessageBox.Show(ex.Message);
	}
	}

```
    // Display speed data when either acceleration or break button is clicked
    private void accelerateButton_Click(object sender, EventArgs e)
    {
        GetCarData();
        myCar.accelerate(5);
        MessageBox.Show("Your car is traveling at " + myCar.speed + "
mph.");
    }

    private void breakButton_Click(object sender, EventArgs e)
    {
        GetCarData();
        myCar.brake(5);
        MessageBox.Show("Your car is traveling at " + myCar.speed + "
mph.");
    }
}
```

I12 Ans.	Override the parent's properties and methods in the child class ? In the same that the child can have its own properties and methods, it can override the properties and method of the parent class. When we override the class's properties and method , we rewrite a method or property that exist in the parent again in the child , but assign to it a different value or code.
I13 Ans.	Prevent the child class from overriding the parent's methods ? How do prevent the child class from overriding the parent's method ? in order to prevent the method in the child from overriding the parent's method , we can prefix the method in the parent with the final keyword.
I15 Ans.	Declare classes and methods as abstract ? Abstract classes are similar to interface . you cannot instantiate them, and they may contain a mix of method declared declare with or without an implementation. However , with abstract classes, you can declare fields that are not static and final ,and define public , protected , and private concrete method.

I16	Can we have non abstract methods inside an abstract class? Explain With Example ?

Ans.	An abstract class means that hiding the implementation and showing the function definition to the user. An abstract class having both abstract method and non-abstract method. For an abstract class, we are not able to create an object directly.
	
	Example : -
	
	abstract class AbstractDemo { // Abstract class
	private int i = 0;
	public void display() { // non-abstract method
	System.out.print("Welcome to Tutorials Point");
	}
	}
	public class InheritedClassDemo extends AbstractDemo {
	public static void main(String args[]) {
	AbstractDemo demo = new InheritedClassDemo();
	demo.display();
	}
	}
	
	Out put : " welcome to tutorials point"

I17	How to create child classes from an abstract class?
Ans.	Since we cannot create object from abstract classes, we need to create child classes that inherit the abstract class code. Child classes of abstract classes are formed with the help of the extends keyword, like any other child class.
I18	What are Magic Methods/Functions? List them in OOPS ? 1.Magic methods are everything for object-oriented Python. 2.Python magic methods can be defined as the special methods which can add "magic" to a class. 3.These magic words start and end with double underscores, for example, _init_ or _add_. The following method names are considered magical: construct(), destruct(), call(), callStatic(), get(), set(), __isset(), __unset(), __sleep(), __wakeup(), __serialize(), __unserialize(),

	__toString(), __invoke(), __set_state(), __clone(), and __debugInfo().

I19
 	
Ans.
	How can we used Virtual Function write an examples in oops ?
	
	A virtual function is a member function which is declared within a base class and is re-defined(Overriden) by a derived class. When you refer to a derived class object using a pointer or a reference to the base class, you can call a virtual function for that object and execute the derived class's version of the function.
	
	Example : -
	
	`#include<iostream>`
	`using namespace std;`
	`//Virtual Function in C++ Programming`
	
	`/*`
	`virtual Function is function is declared in base class and redefined in derived class.`
	`Ability for the runtime polymorphism`
	`*/`
	`class Base`
	`{`
	`public:`
	`virtual void fun()`
	`{`
	`cout<<"Function of Base Class"<<endl;`
	`}`
	`};`
	
	`class Derived:public Base`
	`{`
	`public:`
	`void fun()`
	`{`
	`cout<<"Function of Derived Class"<<endl;`
	`}`
	`};`
	`int main()`
	`{`
	`/*Derived o;`
	`o.fun();*/`
	`Derived o;`
	`Base *p=&o;`
	`p->fun();`
	`return 0;`

I20 How can we Used various type of Constructor write an Examples in oops ?

Ans.

Default Constructor

Whenever we create an object, a constructor is invoked. In case we did not create any constructor, the compiler will automatically write one for us. This constructor known as default constructor. It does not have any parameters and any statement in its body. The main aim of default constructor is to enable you to create an object of a class type. When the compiler create a default constructor, It does nothing. The objects created the default constructor will have fields with their default values.

Example: -

```
class Display {

  int a;
  boolean b;

  public static void main(String[] args) {

    // A default constructor is called
    Display obj = new Display();

    System.out.println("Default Value:");
    System.out.println("a = " + obj.a);
    System.out.println("b = " + obj.b);
  }
}
```

Parameterized Constructor

Unlike default constructor, parameterized constructor have one or more parameters. This type of constructor, it is possible to initialize objects with different set of values at the time of creation. These different set of value initialized to objects must pass as an arguments when the constructor is invoked. The list of parameter can be specified in the parentheses in the same way as parameters are specified in the methods.

Example :-

```
class Display {

 String languages;

 // constructor accepting single value
 Display(String lang) {
  languages = lang;
  System.out.println(languages + " Programming Language");
 }

 public static void main(String[] args) {

  // call constructor by passing a single value
  Display obj1 = new Display("Java");
  Display obj2 = new Display("Python");
  Display obj3 = new Display("C");
 }
}
```

I21

Ans.

Which Constructor have no Parameter write an Examples in oops ?

Default constructor

It does not have any parameters and any statement in its body. The main aim of default contractor is to enable you to create an object of a class type.

```
class Display {
 int a;
 boolean b;

 public static void main(String[] args) {

  // A default constructor is called
  Display obj = new Display();

  System.out.println("Default Value:");
  System.out.println("a = " + obj.a);
  System.out.println("b = " + obj.b);
 }
}
```

I22 How to Secured Internal Data using Encapsulation Write a Example in oops ?

Ans. Encapsulation in OOPs is one of the core properties that makes object-oriented programming an efficient programming paradigm. Encapsulation – together with inheritance, abstraction, and polymorphism – is referred to as the four pillars of object-oriented programming.

Example : -

```
#include <iostream>
using namespace std;

class Adder {
  public:
    // constructor
    Adder(int i = 0) {
      total = i;
    }

    // interface to outside world
    void addNum(int number) {
      total += number;
    }

    // interface to outside world
    int getTotal() {
      return total;
    };

  private:
    // hidden data from outside world
    int total;
};

int main() {
  Adder a;

  a.addNum(10);
  a.addNum(20);
  a.addNum(30);

  cout << "Total " << a.getTotal() <<endl;
  return 0;
}
```

A1 Write a Programme to create a Class in OOPS ?

Ans.

```
#include <iostream>
#include <vector>

using namespace std;

class studyTonight {

   //variable declaration- can be used only within the class as it's declared private
   private:
      int value;

   //Public methods can be called from anywhere- inside as well as outside the class
   public:
      void input() {
         cout << "Entering the input() function\n";
         cout << "Enter an integer you want to display: ";
         cin >> value;
         cout << "Exiting the input() function\n\n";
      }

   void display() {

      cout << "\nEntering the display() function\n";
      cout << "The value entered is: ";
      cout << value;
      cout << "\nExiting the display() function\n\n";
   }
};

int main() {
   cout << "\n\nWelcome to Studytonight :-)\n\n\n";
   cout << " =====  Program to demonstrate the concept of Class, in CPP  ===== \n\n";

   //Declaring class object to access class members from outside the class
   studyTonight object;

   cout << "\n\nCalling the input() function from the main() method\n\n\n";
   object.input();

   cout << "\nCalling the display() function from the main() method\n\n\n";

   object.display();

   cout << "\n\nExiting the main() method\n\n\n";

   //object.value- This will produce an error because variable is declared to be private and
hence cannot be accessed from outside the class
```

```
    return 0;
}
```

A2 Write a Programme to Create a Object in OOPS ?

Ans.

```
// Create a Car class with some attributes
class Car {
  public:
    string brand;
    string model;
    int year;
};

int main() {
  // Create an object of Car
  Car carObj1;
  carObj1.brand = "BMW";
  carObj1.model = "X5";
  carObj1.year = 1999;

  // Create another object of Car
  Car carObj2;
  carObj2.brand = "Ford";
  carObj2.model = "Mustang";
  carObj2.year = 1969;

  // Print attribute values
  cout << carObj1.brand << " " << carObj1.model << " " << carObj1.year << "\n";
  cout << carObj2.brand << " " << carObj2.model << " " << carObj2.year << "\n";
  return 0;
}
```

A3 Write a Programme to Create an Abstract Class In OOPS ?

Ans.

```
// Abstract class
abstract class Animal
{
  // Abstract method (does not have a body)
  public abstract void animal Sound();
  // Regular method
  public void sleep()
  {
    Console.WriteLine("Zzz");
  }
}

// Derived class (inherit from Animal)
class Pig : Animal
{
  public override void animalSound()
  {
    // The body of animalSound() is provided here
```

```
    Console.WriteLine("The pig says: wee wee");
  }
}

class Program
{
  static void Main(string[] args)
  {
    Pig myPig = new Pig(); // Create a Pig object
    myPig.animalSound();  // Call the abstract method
    myPig.sleep();  // Call the regular method
  }
}
```

A4 Write a Programme to Create a Encapsulation in OOPS ?

Ans.

```
#include <iostream>
 using namespace std;

 class Employee {
   private:
     // Private attribute
     int salary;

   public:
     // Setter
     void setSalary(int s) {
       salary = s;
     }
     // Getter
     int getSalary() {
       return salary;
     }
 };

 int main() {
   Employee myObj;
   myObj.setSalary(50000);
   cout << myObj.getSalary();
   return 0;
 }
```

A5 How to Implement Access Modifier Write a Proggramme in OOPS ?

Ans.

```
class Car
{
  private string model = "Mustang";
}

class Program
```

```
{
 static void Main(string[] args)
 {
  Car myObj = new Car();
  Console.WriteLine(myObj.model);
 }
}
```

A6 Write a Proggramme of Copy Constructor in OOPS ?

Ans.

```
class Car {       // The class
 public:          // Access specifier
  string brand;  // Attribute
  string model;  // Attribute
  int year;      // Attribute
  Car(string x, string y, int z) { // Constructor with parameters
    brand = x;
    model = y;
    year = z;
  }
};

int main() {
 // Create Car objects and call the constructor with different values
 Car carObj1("BMW", "X5", 1999);
 Car carObj2("Ford", "Mustang", 1969);

 // Print values
 cout << carObj1.brand << " " << carObj1.model << " " << carObj1.year << "\n";
 cout << carObj2.brand << " " << carObj2.model << " " << carObj2.year << "\n";
 return 0;
}
```

A7 Declare and implement an interface and implement more than one interface in the same class.

Ans. To declare a class that implements an interface, you include an **implements clause in the class declaration**. Your class can implement more than one interface, so the implements keyword is followed by a comma-separated list of the interfaces implemented by the class. By convention, the implements clause follows the extends clause, if there is one.

A8 How to implement the polymorphism principle in OOPS ?

Ans. 1. Compile Time Polymorphism In OOPS, we can achieve compile-time polymorphism with the help of method overloading; in this, java identified

	which method to call by checking their method signature. In method overloading, we should have the same method name but with a different number of arguments and their return type any of them to implement this. ... 2. Run Time Polymorphism
A9 Ans.	Explain Scope resolution operator with example in OOPS ?. The scope resolution operator (::) is used for several reasons. For example: **If the global variable name is same as local variable name, the scope resolution operator will be used to call the global variable**. It is also used to define a function outside the class and used to access the static variables of class.
A10 Ans.	How to Access child class property to Parent class write an Programme in OOPS ? he scope resolution operator (::) is used for several reasons. For example: **If the global variable name is same as local variable name, the scope resolution operator will be used to call the global variable**. It is also used to define a function outside the class and used to access the static variables of class.

A11
 	
Ans.
	Write a Programme of how to define Interface In OOPS ?
	
	package
	{
	public interface Vehicle
	{
	// NO data VARIABLES are allowed in an interface
	// only function PROTOTYPES
	
	/**
	* Comments...
	* Anything that wants to be a "Vehicle" must, implement this fun ction
	*/
	function start_engine() : void;
	
	}
	}

A12 Write a Programme of how to define a Constructor in OOPS ?

Ans.

```
class Example {
 public:
  Example();
  Example(int a, int b);  // Parameterized constructor.

 private:
  int x_;
  int y_;
};

Example::Example() = default;

Example::Example(int x, int y) : x_(x), y_(y) {}
Example e = Example(0, 50);  // Explicit call.
Example e2(0, 50);  // Implicit call.
```

A13 How to define a default constructor write a Programme in OOPS ?

Ans. As a class-based object-oriented programming term, a **constructor** is a unique method used to initialize a newly created object (class). There are a few rules you must follow when creating **constructors**. These rules include: The name of the **constructor** must be the same as the class name. The **constructor** must have no return type.

Example :-

```
// C# Program to illustrate the use
// of Default Constructor
using System;

namespace GFG {

class multiplication
{
   int a, b;

   // default Constructor
   public multiplication()
   {
      a = 10;
      b = 5;
   }

// Main Method
```

```
public static void Main() {

    // an object is created,
    // constructor is called
    multiplication obj = new multiplication();

    Console.WriteLine(obj.a);
    Console.WriteLine(obj.b);

    Console.WriteLine("The result of multiplication is: "
                                +(obj.a * obj.b));
}

}
}
```

Thank you