Test 4 C#

1. Choose paradigms of the object-oriented programming:
* Subtraction
* Polymorphism
* Inheritance
* Encapsulation
1. Choose reference data types in C#:
* short
* int
* string
* class
1. Find declaration of a jagged array:
* string[] cars = {"Volvo", "BMW", "Ford", "Mazda"};
* int[] myNum
* int[,] array = new int[,] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
* int[][] array = new int[3][];
1. You have the following string

string s3 = "Visual C# Express";

System.Console.WriteLine(s3.Substring(7, 2));

What will be the output?

* Visual
* Express
* C#
* Ex
1. What member of the class provides a flexible mechanism to read, write, or compute the value of a private field?
* Destructor
* Constructor
* Method
* Property
1. It means having many forms, usually expressed as “one interface, multiple functions”
* Inheritance
* Abstraction
* Polymorphism
* Constructor
1. This type of class allows us to write class across multiple files
* Static
* Overloaded
* Sealed
* Partial
1. What keyword has to be used so that a function does not return any value?
* Static
* Double
* Void
* PI
1. An element of the class that has the same name as the class and no return value is called
* Object
* Constructor
* Method
* Property
1. Choose an example of a jagged array.
* Int a = new int[2][4]
* string[,] arr2D\_s = new string[4,5]
* Int32[] Intarray
* int[] arr = new int[4] { 10, 20, 30, 40 }
1. Object refers to a particular instance of a class (True / False)
2. A method is a combination of instructions grouped together to achieve some result. It may take arguments and return result (True / False)
3. A constructor is a member that provides a flexible mechanism to read, write, or compute the value of a private field (True / False)
4. A class that is used as the basis for inheritance is called a superclass or base class (True / False)
5. A constructor is a public method with the same name as the class with no return type, which is called once upon object creation (True / False)
6. Encapsulation is the process of showing only essential features of an object to the outside world and hide the other irrelevant information (True / False)
7. Overloading was developed to reduce the number of different method names to be created by the programmer and it also makes life easier for the end object user (True / False)
8. Structures support inheritance (True / False)
9. Array is a data structure that contains several variables of a different type (True / False)
10. ArrayList is a **special array** that provides us with some functionality over and above that of the standard Array (True / False)
11. What is the result of the following code snippet?

|  |
| --- |
| static void Main(string[] args){ int a = 10; int b = 20; Console.WriteLine(Math\_operation(a, b)); Console.ReadLine();}static int Math\_operation(int a, int b){ int result = (a + b) % b; return result;} |

* 15
* 10
* 23
* 11
1. What is the result of the following code snippet?

|  |
| --- |
|  static void Main(string[] args) { float [] arr = new float[] { 10.7f, 20.8f, 45.2f, 60.0f }; float sum = 0; foreach(float i in arr) { if (i < 30) sum += i; else sum += i / 2; } Console.WriteLine("Sum = " + sum); Console.ReadLine(); } |

* 87
* 78.5
* 84.1
* 87.4
1. What will be the output of the following code snippet?

|  |
| --- |
| class Box { public double length; // Length of a box public double breadth; // Breadth of a box public double height; // Height of a box } class Boxtester { static void Main(string[] args) { Box Box1 = new Box(); // Declare Box1 of type Box Box Box2 = new Box(); // Declare Box2 of type Box double volume = 0.0; // Store the volume of a box here // box 1 specification Box1.height = 5.0; Box1.length = 6.0; Box1.breadth = 7.0; // box 2 specification Box2.height = 10.0; Box2.length = 12.0; Box2.breadth = 13.0;  // volume of box 1 volume = Box1.height \* Box1.length \* Box1.breadth; Console.WriteLine("Volume of Box1 : {0}", volume); // volume of box 2 volume = Box2.height \* Box2.length \* Box2.breadth; Console.WriteLine("Volume of Box2 : {0}", volume); Console.ReadKey(); } } |

* 210 1500
* 1500 210
* 210 1560
* 1560 210
1. What will be the output of the following code snippet?

|  |
| --- |
| class Program { public static void Main() { int[] arr = new int[5] { 10, 12, 6, 8, 1 }; int sum = 0; for (int i = 0; i < arr.Length; i++) { if (i % 2 == 0) sum += arr[i]; } Console.WriteLine("Sum = " + sum); } } |

* 22
* 17
* 16
* 21
1. What will be the output of the following code snippet?

|  |
| --- |
|  class Program { public static void Main() { int[] arr = new int[] { 6, 3, 9, 10, 1, 7, 12 }; int sum = 0; for (int i = 0; i < arr.Length; i++) { if (i % 3 == 0) sum += arr[i]; else sum += 1; } Console.WriteLine("Sum = " + sum); } } |

* 32
* 20
* 22
* 31