**C# programming test**

1. What is a reference type in C#?
* Arrays
* Byte
* Int
* Pointer
1. What is a maximum number of available elements in an array?
* 10
* 100
* 1000
* None of them
1. What are the example of loops in C#?
* While
* Do … while
* If
* For
1. What keyword has to be used so that a function does not return any value?
* Static
* Double
* Void
* PI
1. Choose paradigms of the object-oriented programming.
* Subtraction
* Polymorphism
* Inheritance
* Encapsulation
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. What access modifier of the field will you choose if you only want to get access to it within a method inside this class, but not from the object?
* public
* private
* output
* protected
1. What keyword is required to be used to address a class by its name and not its object?
* Override
* Protected
* Static
* External
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 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 result of the following code snippet?

|  |
| --- |
|  public static void Main() { classB b = new classB(); Console.WriteLine(b.Print()); } public class classA { public virtual string Print() { return "classA"; } } public class classB: classA { public override string Print() { return "classB"; } } public class classC: classB { public new string Print() { return "ClassC"; } } |

* ClassA
* ClassB
* ClassC
* Error
1. What will be the result of the following code snippet?

|  |
| --- |
|  class Program { public static void Main() { Arithmetic arith = new Arithmetic(); arith.Sum(arith.a, arith.b); } } class Arithmetic { public int a = 10; public int b = 15; static public int Sum(int a, int b) { return a + b; } } |

* 25
* 15
* Error
* 10
1. What will be the output of the following code snippet?

|  |
| --- |
|  class Program { public static void Main() { Person p = new Person(); p.FirstName = "Andrew"; p.LastName = "JackSon"; p.Country = "Germany"; Console.WriteLine(p.FirstName + " " + p.LastName + " " + p.Country + " " +p.BirthDate); } } class Person { private string firstName; private string lastName; DateTime birthDate; string country; public string FirstName { get { return firstName; } set { firstName = value; } } public string LastName { get { return lastName; } set { lastName = value; } } public DateTime BirthDate { get; set; } public string Country { get; set; } public Person() { firstName = "Unknown"; } public Person(string name, string lastname) { firstName = name; LastName = lastname; } } |

* Andrew JackSon Germany
* Andrew JackSon Germany 01.01.0001 0:00:00
* Andrew JackSon
* Andrew
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
1. These classes are used to restrict the inheritance feature of object-oriented programming
* Protected
* Private
* Sealed
* Abstract
1. The keyword **…** can be used in the case of ambiguous and unrecommended naming
* it
* this
* override
* virtual
1. A variable which is declared inside a method is called a\_\_\_\_\_\_\_\_variable
* Local
* Private
* Static
* Serial
1. Feature of a local variable
* It must be declared within a method
* It represents a class object
* It can be used anywhere in the program
* It must accept a class

#### Two methods with the same name but with different parameters.

* Overloading
* Multiplexing
* Duplexing
* Loading

#### Which of the following class cannot be inherited?

* Abstract
* Sealed
* Both
* None