1. **El CTS (Common Type System) define tipos de valor y referencia. Se tiene un array de enteros.**
   a. Pertenece al grupo de tipos de valor (y sus elementos también).
   b. Pertenece al grupo de tipos de referencia (y sus elementos también).
   c. Por defecto sus elementos son inicializados en null.
   d. Por defecto es inicializado en null.
   e. a y c.
   f. b y d
   g. c, d y e
   h. Ninguna de las anteriores.

2. Ciclo de vida de un objeto:
   a. Comienza al asignar memoria mediante la palabra reservada new.
   b. Comienza al asignar memoria mediante la llamada al constructor.
   c. Su presencia dentro del programa finaliza al eliminar una o todas las referencias que tenemos a él.
   d. No desaparece de memoria hasta que el Garbage Collector decide liberarla.
   e. Todas las anteriores.
   f. Ninguna de las anteriores.
   g. a y c.
   h. b y d.
   i. a y d.

3. **Operadores lógicos. Las evaluaciones se realizan por "cortocircuito", esto implica que:**
   a. No se evaluara la segunda condición.
   b. Siempre se evaluara la primera condición.
   c. Puede evaluarse la segunda condición, si la primera es verdadera.
   d. Puede evaluarse la segunda condición, si la primera es falsa.
   e. b y c.
   f. b y d.
   g. b, c y d.

4. Se tiene la clase sellada Persona, con un constructor de clase, uno de instancia y privado y la última sobrecarga es de instancia y publica:
   a. Al instanciar el primer objeto, el constructor público es el primero por el cual pasara.
   b. El constructor público debe ser llamado por sus clases derivadas.
   c. El constructor privado solo puede ser llamado por el constructor público.
   d. Al instanciar un objeto podrá pasar por el constructor público o estático, lo cual se definirá en tiempo de ejecución.
   e. El constructor de clase no puede recibir parámetros.
   f. Todas las anteriores.
   g. Ninguna de las anteriores.
   h. c, d y e.
   i. c y d.
   j. a, b y c.

5. **Formularios:**
   a. **Heredan, directamente o indirectamente de System.Windows.Forms.**
   b. **Son objetos que exponen propiedades, métodos y eventos.**
   c. **Utilizan el concepto de partial class.**
   d. **Su ciclo de vida consta de 7 etapas: New, Load, Paint, Activated, FormClosing, FormClosed y Dispose.**
   e. **Todas las anteriores.**
   f. Ninguna de las anteriores.
   g. a, b y d.
   h. a, b y c.

6. **Las conversiones implícitas se utilizan cuando:**
   a. Por medio de casteo convierto de un tipo a otro.
   b. La conversión no implícita no implica perdida de información.
   c. La conversión la realizo por intermedio de un método (por ejemplo, Parse).
   d. **La conversión sea automática.**
   e. Ninguna de las anteriores.
   f. Todas las anteriores.

7. **Dada la siguiente porción de código, ¿Qué se mostraría en la consola?**

```
Stack<string> a = new Stack<string>();
a.Push("a");
a.Push("b");
a.Push("c");
while (a.Count > 0)
   Console.Write((a.Pop()).ToString());
```

   a. “c”.
   b. “abc”.
   c. **“cba”.**
   d. “a”
   e. Sintéticamente esta mal.
   f. Arroja una excepción.
   g. b y f.
   h. c y f.

8. **Herencia:**
   a. Cada clase que hereda de otra posee los atributos, métodos y constructores de la clase base, además de los propios.
   b. Existen dos tipos de herencia en .NET: La simple y la compuesta y C# puede utilizar cualquiera.
   c. Si un miembro de la clase base es privado, no se podrá heredar.
   d. **La clase derivada no podrá ser más accesible que su clase base.**
   e. Si la clase base posee un constructor sobrecargado, las clases derivadas están obligadas a invocar a dicho constructor.
   f. a y d.
   g. b y e.

h. a y e.
i. d y e.
j. Ninguna de las anteriores.

9. **Si quiero generar una propiedad que deba ser sobrescrita en las clases derivadas y solo accedidas por las mismas, la debo definir como:**
   a. public abstract.
   b. public virtual.
   c. private abstract.
   d. private virtual.
   e. protected abstract.
   f. protected virtual.

10. Sobrecargas:
   a. La sobrecarga de métodos consiste en cambiar el número de parámetros, tipo de parámetros y el tipo de retorno.
   b. Una sobrecarga valida es cuando se cambian los nombres de los parámetros.
   c. El compilador distingue los métodos que están sobrecargados comparando la lista de parámetros.
   d. Las sobrecargas de métodos deben tener el mismo modificador de visibilidad.
   e. Todas las anteriores.
   f. Los constructores de instancia se deben sobrecargar.
   g. Los constructores de clase se pueden sobrecargar.

11. Dadas las clases A, B y C.
   a. A puede heredar de B, y C, si las clases bases son públicas.
   b. Si A hereda de B, A hereda los constructores de B.
   c. Si C hereda de B y B hereda de A, entonces C hereda solo los miembros de A.
   d. Si B hereda de A y C hereda de A, entonces A hereda los miembros de B.
   e. Si B hereda de A y C hereda de A, entonces A debe ser una clase sellada.
   f. Todas las anteriores.
   g. Ninguna de las anteriores.
   h. a y e.
   i. b y d.
   j. c y d.

12. Si hablo de abstracción, ¿Cuáles de estas características le corresponde¿
   a. Ignorancia selectiva.
   b. El exterior de la clase lo ve como una caja negra.
   c. Va de la generalización a la especialización.
   d. Para reforzarse, utiliza el encapsulamiento.
   e. La invocación es resuelta al momento de la ejecución.
   f. Todas las anteriores.
   g. a, b, c y d.
   h. a y d.

i. a, d y e.
j. Ninguna de las anteriores.

13. Si deseo sobrescribir un método:

a. Debo cambiar el número, el tipo y el orden de los parámetros.
b. El compilador de C# distingue métodos comparando las listas de parámetros.
c. Debo conservar el mismo nombre y valor retorno.
d. Todas las anteriores.
e. Ninguna de las anteriores.

14. ¿Cómo puedo utilizar clases de otro proyecto en una misma solución?

a. Agregando la palabra using y el nombre del proyecto.
b. Agregando la referencia al proyecto, la palabra using y el nombre del proyecto.
c. Agregando la referencia al proyecto, la palabra using y el nombre del nameSpace.
d. Agregando la referencia a la dll.

15. ¿Qué opción es correcta para poder tener la siguiente línea de código Objecto o += obj; (siendo obj de tipo Objeto)?

a. public static Objeto operator +(Objeto c, Objeto m)
b. public Objeto operator +(Objeto c, Objeto m)
c. public static Objeto operator +=(Objeto c, Objeto m)
d. public Objeto operator +=(Objeto c, Objeto m)
e. public static object operator +(Objeto c, Objeto m)
f. Ninguna de las anteriores.

16. ¿Qué fragmento de código no tiene error de sintaxis?

a. public override int miMetodo();
b. public static int miMetodo();
c. public virtual int miMetodo();
d. public abstract int miMetodo();
e. Todas las anteriores.
f. Ninguna de las anteriores.

17. **Seleccione la opción correcta y desarrolle en C# la porción de código correctamente.**

Realizar la propiedad **"PrecioDeCosto"** que solo permita asignar un valor (entero) al atributo **"_precioDeCosto"** y además que asigne el precio de venta al atributo **"_precioDeVenta",** sumando un **27%** al precio de costo. ¿Cómo debo hacerlo?

a. Dentro del set asigno el valor al atributo _precioDeCosto que trae "value" y llamo al método de instancia sin parámetros llamado "CambiarElPrecio()".
b. Dentro del set asigno el valor al atributo _precioDeCosto que trae "value" y coloco esta línea "this._precioDeVenta = this._precioDeCosto * 0.27".
c. Dentro del get asigno el valor al atributo _precioDeCosto que trae "value" y coloco esta línea "this._precioDeVenta = this._precioDeCosto * 1.27".

d. Dentro del set asigno el valor al atributo _precioDeCosto que trae “value” y llamo al método de clase sin parámetros llamado “CambiarElPrecio()”.

18. Sabiendo que estas líneas de código son correctas, que los atributos de las clases son **públicos** y además que todas las clases poseen un **único constructor**, realice los constructores, de cada una de las clases, sabiendo que ProdVendido hereda de ProExport, que ProdExport gereda de ProdImpuesto y que este último hereda de Producto.

```
Producto pro = new Producto("TV Led");
        ProdImpuesto pl = new ProdImpuesto(pro.nombbre, 5988.33f);
        ProdExport pEX = new ProdExport(pl, "Argentina");
        ProdVendido p = new ProdVendido(pEX, "Cliente en Uruguay");
```

19. ¿Qué muestra el siguiente bloque de código?

```
int i = 20;
for(; ; )
{
  Console.Write(i + " ");

  if (i >= -10)
      i -= 4;
  else
      break;
}
```

a. 16 12 8 4 0 -4 -8
b. 20 16 12 8 4 0 -4 -8
c. 20 16 12 8 4 0 -4 -8 -12
d. 16 12 8 4 0 -4 -8 -12
e. Ninguna de las anteriores
f. Sintácticamente está mal.
g. Arroja una excepción.

20. ¿Cuál es el resultado de la siguiente línea de C#?

```
public class D
  {
    public virtual void DoWork()
    {
      Console.WriteLine("DoWork");
    }
  }

  public abstract class E : D
  {
    public abstract override void DoWork(int i);
  }


public class F : E
  {
    public override void DoWork()
```

```
    {
        Console.WriteLine("DoWork Two");
    }
}

class Program
{
    static void Main(string[] args)
    {

        F objeto = new F();

        objeto.DoWork();


    }
}
```

a. DoWork y luego DoWork Two.
b. DoWork.
c. DoWork Two.
d. Ninguna de las anteriores.
e. Sintácticamente está mal.
f. Arroja una excepción.

RESPUESTAS

1. **D**
2. **H**
3. **G**
4. **H**
5. **E**
6. **D**
7. **C**
8. **D**
9. **E**
10. **C**
11. **G**
12. **H**
13. **C**
14. **C**
15. **A**
16. **D**
17. **A**
18.
19. **C**
20. **E**