

# Tarea 1 Programación Orientada a Objetos en C#

Integrantes:

Alexis Brayan López Matías

Arturo Castillo Valles

Carlos Naranjo Robledo

Ricardo Martínez Ríos

1. ¿Qué es un algoritmo y cuáles son sus características?

Un algoritmo es un conjunto de pasos o receta que se usa para llevar una tarea a cabo, en la mayoría de los casos necesita una entrada y regresa un resultado, también tiene complejidad de tiempo y espacio/memoria.



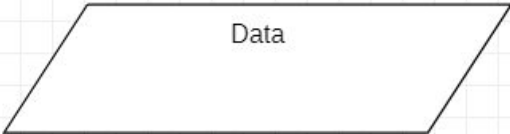
2. ¿Qué es un diagrama de flujo?

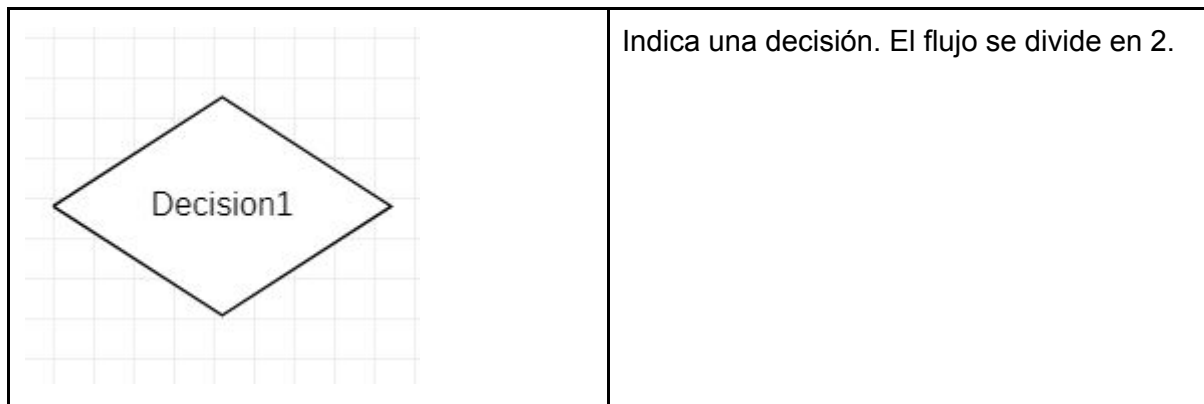
Diagrama que describe un proceso, sistema o algoritmo informático. Los diagramas de flujo emplean rectángulos, óvalos, diamantes.

3. ¿Qué es un programa?

Un programa es una secuencia finita de instrucciones que permiten la realización de una tarea en específico.

4. Realiza una tabla explicando la simbología de los diagramas de flujo.

	Indica un proceso a realizar. Suma dos números por ejemplo.
	Indica el flujo a seguir
	Indica una entrada o salida de un proceso.



5. ¿Qué es un lenguaje de programación?

Un lenguaje que nos ayuda a expresar instrucciones de manera detallada a una computadora electrónica, ya sea sustituyendo lenguaje ensamblador a lenguaje máquina, o traduciendo un lenguaje de alto nivel a ensamblador.

6. ¿Qué es el lenguaje máquina?

Es el sistema de códigos directamente interpretable por un circuito microprogramable, como el microprocesador de una computadora o el microcontrolador de un autómata.

7. ¿Qué es el lenguaje ensamblador?

Es un lenguaje de bajo nivel conformado por un conjunto de instrucciones específico para cada tipo de procesador.

8. ¿Qué es un lenguaje de alto nivel?

Lenguajes que permiten programar independientemente del conjunto de instrucciones de cada procesador. Son mucho más parecidos al lenguaje humano que al de máquina.

9. ¿Qué es un paradigma de programación?

Un paradigma de programación es una técnica/estilo que tienen los lenguajes de programación, y son la base del diseño de estos lenguajes, aunque un lenguaje puede “pertenecer” a varios paradigmas, o al menos tener la capacidad de usarlos.

10. ¿Cuál es la clasificación de los lenguajes de alto nivel? Da ejemplos de cada clasificación

La mayoría de los paradigmas son variantes de la programación, imperativa y declarativa, en la programación imperativa se describe paso a paso cada instrucción del programa, y en la programación declarativa se describe el problema y por mecanismos de inferencia se encontrarán las instrucciones para solucionarlo.

- Programación Imperativa: C, BASIC, Pascal
  - Programación Estructurada: ALGOL, Ada
  - Programación Orientada a Objetos: Java, C#
- Programación Declarativa:
  - Programación funcional: Haskell.
  - Programación Lógica: Prolog

- Programación multiparadigma: Python, Lisp

11. Explica:

- 1. Paradigma orientado a objetos.

Paradigma que se basa en la abstracción y modelamiento de objetos para la resolución de un problema, donde cada objeto posee atributos (características) y métodos (funcionalidad). Este paradigma cuenta con cuatro características fundamentales:

- Abstracción: Aislamiento de un elemento de su contexto para el modelamiento de sus propiedades y comportamiento.
- Encapsulamiento: Restricción de acceso de funciones y atributos internos de una clase.
- Herencia: Permite que una clase derivada posea las características y comportamiento de la clase padre, así como también poder extender su funcionalidad.
- Polimorfismo: Permite enviar mensajes sintácticamente iguales a objetos de distintos tipos de tal forma que estos sean capaces de entenderlos.

- 2. Paradigma orientado a eventos.

Paradigma en el cual el flujo del programa está determinado por la ocurrencia de eventos. Este tipo de paradigma se caracteriza por efectuar una acción después de que un evento se presente.

- 3. Paradigma funcional.

Se basa en la resolución de problemas mediante el uso de funciones que se centran en qué se debe hacer y no en cómo hacerlo. La composición de funciones, la transparencia referencial y las funciones puras son los tres conceptos fundamentales que componen a este tipo de paradigma.

12. Menciona 5 lenguajes que usan el paradigma orientados a objetos

- C++
- Java
- C#
- Python
- Kotlin

13. Menciona 5 lenguajes que usan el paradigma orientados a eventos

JavaScript, Swift, C#, Java, Python

14. ¿Qué es un compilador?

Traductor que transforma un programa en un lenguaje, a otro. reúne los elementos para su ejecución y los almacena para hacer uso de ellos en el futuro.

15. ¿Qué es un intérprete?

Es un programa que recibe código de alto nivel, lo analiza y posteriormente lo ejecuta línea por línea.

16. ¿Qué es el código de bytes?

Es un código intermedio que se acerca al lenguaje de máquina. Se produce en lenguajes interpretados como Java o Javascript para optimizar su ejecución en el intérprete, y permite pasar a lenguaje de máquina más fácil y de manera portable.

17. ¿Qué es un IDE y menciona alguno que podemos usar para C#?

Un IDE es un ambiente de trabajo con una interfaz gráfica que nos ayuda en el desarrollo de aplicaciones, lo más básico que tiene es un editor de texto que puede adaptarse al lenguaje de programación que estemos utilizando, también puede traer herramientas para realizar ingeniería inversa sobre base de datos, creación de ventanas, etc.

18. ¿Qué es un Framework?

Esquema o estructura que se establece y que se aprovecha para desarrollar y organizar software, es el entorno pensado para hacer más sencilla la programación de cualquier aplicación o herramienta actual, automatiza muchos procesos y facilita el conjunto de la programación.

19. Menciona algún Framework para C#

.Net framework, como su nombre lo dice, es un framework de C# que permite crear aplicaciones Form-based y Web-based, aunque también es posible crear lo que se llaman Web-Services. Además, proporciona un entorno de desarrollo sencillo, con mayor seguridad y por lo tanto menor cantidad de vulnerabilidades.

20. ¿Qué es Sublime Text?

Un editor de texto muy chulo, portable, ligero y con soporte de muchos plugins para facilitar el desarrollo en cualquier lenguaje.

21. ¿Qué es Java?

Es un lenguaje de programación orientado a objetos, muy popular debido a su habilidad de correr en “donde sea” gracias a su máquina virtual, también aumentó su popularidad al ser la base para el Sistema Operativo Android y por su sintaxis fácil de entender.

22. ¿Qué es C#?

Lenguaje de programación multiparadigma desarrollado y estandarizado por Microsoft como parte de su plataforma .NET, diseñado para desarrollar programas de la misma.

23. Escribe la historia de C#.

Anders Hejlsberg formó un equipo en enero de 1999, con el objetivo de diseñar un nuevo lenguaje de programación, que inicialmente fue nombrado como Cool (C-like Object Oriented Language). No fue hasta julio de 2002 que el nuevo lenguaje fue liberado por Microsoft, aunque bajo el nombre de C#.

La versión 1.0 de C# fue liberada en enero de 2002 junto a Visual Studio .Net 2002. Posteriormente en el año 2005 se dio a conocer la versión 2.0 de C# que incluía nuevas características, como lo son genéricos, tipos parciales y métodos anónimos.

La siguiente versión, C# 3.0, llegó a finales del 2007 con una gran cantidad de nuevas características, algunas de las más destacables son los tipos anónimos, expresiones lambda, propiedades auto-implementadas y expresiones de consulta.

La versión 4.0 se liberó en el 2010 y proveía de tipos dinámicos, argumentos opcionales y con nombre y tipos *interop* embebidos.

En 2012 salió la versión 5.0 que añadía programación asincrónica y atributos *Caller info*.

En la siguiente versión, 6.0, se introdujeron muchas características que hacían del lenguaje un poco más cómodo, como por ejemplo las importaciones estáticas, filtros de excepciones, interpolación de cadenas y propagador de *NULL*.

C# 7.0 incluyó nuevas características para que los desarrolladores tuvieran la capacidad de escribir código más limpio. Algunas características son variables *out*, tuplas y deconstrucción, funciones locales y pattern matching.

Finalmente la última versión, C# 8.0, fue liberada en el año 2019. En esta nueva versión se agregaron miembros de sólo lectura, mejoras en pattern matching, flujos asíncronos, índices y rangos, entre otras características.

24. Menciona 5 características del lenguaje C# (Da una descripción de ellas).

- Orientado a objetos: Sigue el paradigma OOP, cumpliendo con abstracción, encapsulamiento, herencia y polimorfismo.
- Tiene recolector de basura: Libera automáticamente la memoria del heap que ya no se esté usando.
- Manejo de excepciones: Mecanismo que permite manejar errores de manera estructurada y cómoda.
- Sistema de tipos unificado: Todos los tipos, incluyendo los primitivos como int y double heredan de una sola clase object.
- Tipado estático: El tipo de cada variable se define en tiempo de compilación y no cambia en el transcurso de la ejecución del programa.

25. ¿Por qué aprender C#?

Se pueden hacer varias cosas con este lenguaje, desde aplicaciones Web, aplicaciones de escritorio, juegos, aplicaciones de consola, etc., es muy versátil, también pertenece al framework .NET, que facilita la producción de tecnologías/aplicaciones Web, independientemente del hardware usado, es un “producto” de Microsoft. Algunos lo consideran el rival de Java, por sus similitudes.

26. Menciona algunas diferencias entre Java y C#

C#

- Sobrecarga de operadores
- Tipos Anónimos
- Los tipos básicos también derivan de la clase Object
- Tipos Genéricos en tiempo de Ejecución
- Permite el uso de goto
- Permite el uso de structures y unions

Java

- Más adecuado para el uso de concurrencia

- Java admite la palabra clave `strictfp`, lo que significa que los resultados para un punto flotante serán los mismos para diferentes plataformas.
- compatibilidad y robustez de plataforma

27. ¿Cuándo usar programación orientada a objetos y cuándo usar programación estructurada? Dar 3 ejemplos.

El uso de un paradigma está ligado a los requerimientos del problema a solucionar, como por ejemplo el lenguaje a emplear, la facilidad de desarrollo o la curva de aprendizaje.

Programación estructurada:

- Programación de sistemas embebidos.
- Programación de sistemas operativos.
- Mantenimiento a sistemas bancarios antiguos (Fortran, C, Cobol).

Programación orientada a objetos:

- Desarrollo de videojuegos.
- Desarrollo de aplicaciones de escritorio.
- Desarrollo de aplicaciones web (JSP, JSF, PHP).

28. Definir concretamente los siguientes conceptos:

1. Clase: Descripción de un conjunto de atributos y métodos.
2. Mensaje: Información que se envía desde un emisor a un receptor.
3. Objeto: Instancia de una clase.
4. Atributo: Información que pertenece a cada objeto.
5. Método: Procedimiento unido a un objeto.
6. Instancia: Algo concreto.
7. Referencia: Valor que apunta a otro.

29. Menciona 20 palabras reservadas de C# y breve descripción.

- `public`: declara un elemento que todos pueden usar libremente
- `protected`: declara un elemento que se puede usar libremente en una clase y en sus clases heredadas.
- `private`: declara un elemento que se puede usar solo en la clase en el que fue declarado.
- `if`: control de flujo IF.
- `else`: control de flujo ELSE.
- `switch`: control de flujo SWITCH.
- `case`: control de flujo CASE.
- `null`: valor "nulo"
- `for`: control de flujo FOR
- `while`: control de flujo WHILE
- `new`: creación de objetos
- `int`: tipo INTEGER
- `float`: tipo FLOAT
- `double`: tipo DOUBLE
- `bool`: tipo BOOL (boolean)
- `char`: tipo CHAR
- `string`: tipo STRING (cadenas)
- `override`: indica sobreescritura

- void: tipo VOID (vacío)
- true: valor “verdadero”

### 30. ¿Qué es un identificador en C#?

Los nombre que reciben las variables, deben comenzar por alguna letra o por `_`, no deben tener espacios intermedios, pueden contener caracteres de formato Unicode, se pueden declarar identificadores que coincidan con la palabra clave de C# mediante el prefijo `@`, con algunas convenciones como:

- Los nombres de interfaz empiezan por una `I` mayúscula.
- Los tipos de atributo terminan con la palabra `Attribute`.
- Los tipos de enumeración usan un sustantivo singular para los que no son marcas y uno plural para los que sí.
- Los identificadores no deberían contener dos caracteres `_` consecutivos. Esos nombres están reservados para los identificadores generados por el compilador.

### 31. ¿Qué es un dato?

Es la unidad mínima de información que tiene una representación de algo, como por ejemplo la edad o un nombre.

### 32. ¿Qué es un tipo de dato?

El conjunto de valores a los cuales el dato puede tomar como valor.

### 33. ¿Qué es una literal?

Son valores fijos, las constantes apuntan a estas, el programa no puede alterarlas en tiempo de ejecución.

### 34. ¿Qué es una variable?

Símbolo que puede tomar un valor de un conjunto de valores (dominio de la variable).

### 35. ¿Qué es una constante y cómo se declara en C#?

Son valores inmutables conocidos desde el momento de compilación y cuyo valor no cambia durante la ejecución del programa. Para declarar una variable constante se debe colocar la palabra reservada **const** antes del tipo de la variable, por ejemplo: `const int i = 0;`

### 36. Tipos de datos primitivos en C# y sus características.

- bool: Representa un valor verdadero o falso
- byte: Entero sin signo de 8 bits.
- sbyte: Entero con signo de 8 bits.
- char: Representa un caracter de UTF-16.
- decimal: Flotante con 16 bits de precisión,
- double: Flotante con 8 bytes de precisión.
- float: Flotante con 4 bytes de precisión.
- int: Entero con signo de 32 bits.
- uint: Entero sin signo de 32 bits.
- long: Entero con signo de 64 bits.
- ulong: Entero sin signo de 64 bits.

- short: Entero con signo de 16 bits.
- ushort: Entero sin signo de 16 bits.
- string: Secuencia de char.

37. Menciona todos los operadores en C# (aritméticos, booleanos, unarios, binarios, etc.).

+, -, /, \*, %, ., &, &&, |, ||, ?, ??, ^, =, ==, \*=, +=, -=, /=, %=, ^=, !=, ~, ++, --, ->, !, <, <=, >, >=, <<, >>, is, as, new, &=, |=, <<=, >>=, [], (), new, typeof, checked, unchecked, default, nameof, delegate, sizeof, stackalloc.

38. ¿Qué es un casting en C#?

Forma de informar explícitamente al compilador que tiene la intención de realizar la conversión y que sabe que puede producirse una pérdida de datos, generalmente se usa el tipo al que se quiere convertir entre paréntesis delante del valor o variable a convertir.

39. Explicar la visibilidad de los miembros de una clase:

- public: Acceso al tipo o miembro desde cualquier clase.
- private: Acceso al tipo o miembro desde la clase interna.
- protected: Acceso al tipo o miembro desde la clase interna o clases derivadas.

40. Escribe las convenciones de escritura de un código en C#.

Según

<https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/inside-a-program/coding-conventions>

Usar el indentado del editor que use el equipo.

Sólo una expresión por línea.

Sólo una declaración por línea.

Divide con al menos 1 línea blanca las propiedades de los métodos.

Usa siempre paréntesis para dividir expresiones booleanas.

Comenta con buena ortografía

Comenta siempre en una nueva línea.

Usa StringBuilder cuando requieres construir al vuelo cadenas muy grandes.

Sólo usa tipado implícito (**var** en vez del tipo) solamente si el lado derecho de la declaración es completamente clara.

Prefiere datos con signo.

Usa inicializadores. (Seguir con un bloque de definición de cada atributo {})

Usa lambdas para eventos de un sólo uso.

41. Escribe diferencias entre C y C# (principalmente en manejo de memoria, arreglos y tipos básicos).

C# "recoge" la basura automáticamente, mientras que en C se tiene que hacer a mano para que no se desperdicie memoria, entre los tipos básicos en C#, se encuentran las cadenas (string), mientras que su uso y creación en C son mucho más complejas, los demás tipos primitivos son los mismos, los arreglos en C# no pueden ser manipulados como punteros en C, de hecho no es recomendable y se tiene que usar palabras reservadas para indicar que

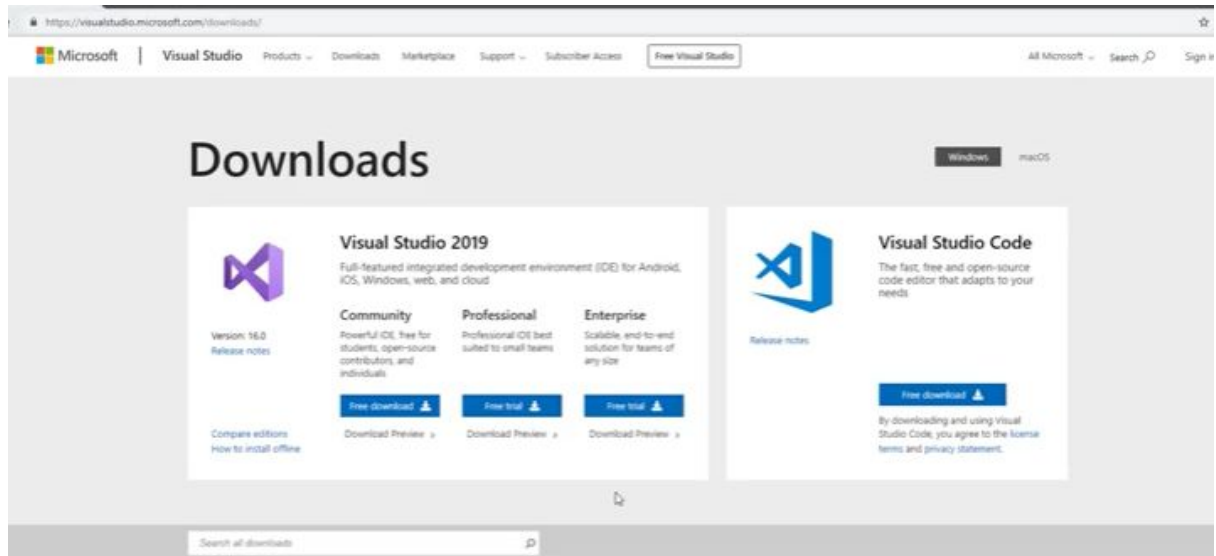


se están usando punteros para manipulación de arrays, en C# se usa generalmente un arreglo con la notación de paréntesis cuadrados, nada más.

42. Realiza un tutorial con capturas de pantalla de:

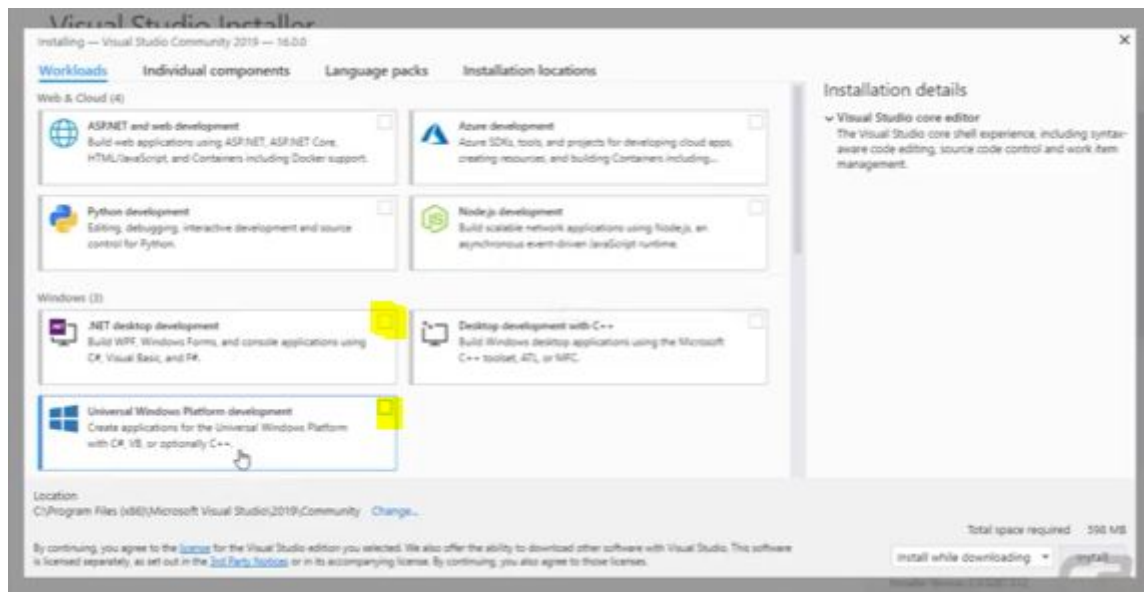
1.- ¿Cómo instalar el visual studio?

- Descargamos visual studio de la página de microsoft



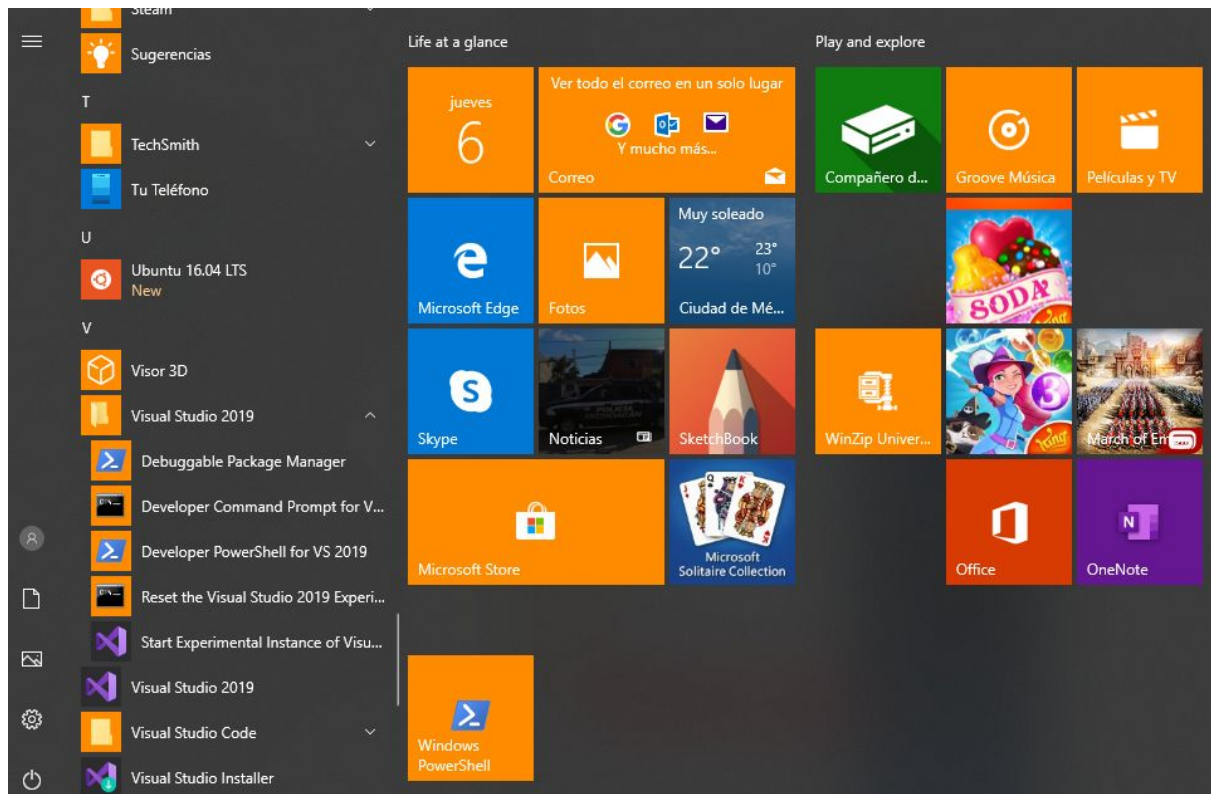
All downloads

- Marcamos las casillas como se indica abajo



2.- ¿Cómo compilar un programa desde consola?

Se puede abrir la terminal desde el menú de windows



o añadir el programa csc.exe al path, que podría estar en la ubicación, dependiendo de la versión y la arquitectura,

C:\Windows\Microsoft.NET\Framework64\v4.0.30319, (ejecutar con csc File.cs)

3.- ¿Cómo hacer un hola mundo en C#?

```

1  using System;
2
3  namespace Hello
4  {
5      0 references
6      class Program
7      {
8          0 references
9          static void Main(string[] args)
10         {
11             Console.WriteLine("Hello World!");
12         }
13     }

```

43. ¿Cómo se comenta en un código de C#?

- Comentarios de línea: // Esto es un comentario.
- Comentarios de bloque: /\* Esto es un comentario de bloque. \*/

- Comentarios de etiquetas XML: `/// <summary> Comentario de XML. </summary>`

44. ¿Por qué es importante comentar un código? (En cualquier lenguaje).

Principalmente para documentar el por qué de cada decisión, evitando que se hagan cambios incompatibles en futuras versiones.

45. Escribe la definición de los siguientes conceptos.

- Herencia: es pasar el comportamiento y características de una clase a otras.
- Abstracción: obtener el concepto base de algo, sin importar su significado “complejo” en el mundo real.
- Polimorfismo: se tiene polimorfismo cuando se tiene un mismo método definido, pero con un número diferente de parámetros o de un tipo diferente, esto hace que un objeto pueda tener un comportamiento similar en un método, sin importar los datos que se le pasen, esto también aplica cuando un objeto de una clase que heredó de otra utiliza un método de la clase padre o propia dependiendo de los parámetros usados.
- Upcast: es un “casteo” de una clase padre a una clase heredada, esto siempre es permitido, por ejemplo, un “Animal” base puede ser un “Gato”, el comportamiento y atributos básicos de un Animal están presentes en un Gato.
- Downcast: es lo contrario a un upcast, se quiere castear de una clase heredada a una padre, pero esto puede causar errores, ya que un Gato tiene comportamientos y atributos extras que un Animal base no tiene, no los puede perder así como así.
- Interfaces: indica/restringe un comportamiento específico en las clases que la implementan, parecido a la herencia, aunque puede ser múltiple.
- Encapsulamiento: es la protección de datos y su integridad, de esta manera evitamos acceso no autorizado a datos que son de vital importancia en un programa, se usan las palabras reservadas `protected` y `private`.

46. ¿Qué es la sobrecarga de métodos en C#?

Utilizar el mismo identificador de método para definir dos o más métodos, pero cada uno con diferentes parámetros (diferente firma del método)

47. ¿Qué es un constructor?

Es el método de una clase que se ejecuta de forma automática al generar una instancia de ésta. El constructor tiene el mismo nombre que la clase, permite inicializar los atributos y no retorna nada.

48. Menciona las diferencias entre los tipos de tipado: fuerte, débil, estático, dinámico.

- Débil: hace conversión de tipos implícitamente (javascript).
- Fuerte: Requiere hacer conversiones explícitas (casting) entre tipos. (Python)
- Estático: Los tipos de cada variable se establecen en tiempo de compilación. (C)
- Dinámico: Los tipos de cada variable se conocen hasta el tiempo de ejecución y pueden cambiar.

49. ¿Qué tipo de tipado tiene C#?

En un principio era de tipado fuerte, pero la capacidad de tipado dinámico se agregó al usar la palabra reservada “dynamic”.

50. ¿Qué es un tipo de dato anónimo en C# y para qué sirve?

Los tipos anónimos proporcionan una forma conveniente de encapsular un conjunto de propiedades de solo lectura en un solo objeto sin tener que definir explícitamente un tipo primero. El nombre del tipo es generado por el compilador y no está disponible en el nivel del código fuente. El compilador infiere el tipo de cada propiedad.

51. ¿Para qué sirve la palabra reservada using?

- Permite el uso de tipos en un namespace sin tener que calificar el uso de dicho tipo.
- Permite el acceso a miembros estáticos y tipos anidados de un tipo sin tener que calificarlo.
- Permite crear alias para un namespace o un tipo.

52. ¿Qué es un espacio de nombres?

En C# .Net los usa para organizar algunas clases

- Organizan grandes proyectos de código.
- Se delimitan mediante el uso del operador “.”
- La directiva using evita el requisito de especificar el nombre del espacio de nombres para cada clase.
- El espacio de nombres global es el espacio de nombres "raíz": global :: System siempre se referirá al espacio de nombres del sistema .NET.

53. ¿Cuál es la función de los siguientes espacios de nombres?

- System
- System.Collections.Generic
- System.Linq
- System.Text
- System.Threading.Tasks

54. ¿Qué es el Garbage Collector y cómo funciona en C#?

Es parecido a lo que hace la máquina virtual de Java al detectar que una locación de memoria ya no está siendo referenciada en ninguna parte, cuando esto pasa, el recolector “limpia” de datos esa locación y hace posible que se pueda usar en otro proceso, también ayuda a que un objeto no trasape su bloque de memoria con otro, y acomoda eficientemente los objetos en el heap.

55. Menciona la sintaxis en C# para las siguientes estructuras de control y da un ejemplo de cada una:

- Ciclo for:

```
for (initializer; condition; iterator){  
    statement;  
}
```

Ejemplo:

```
for (int i = 0; i<10; i++)  
{  
    System.Console.WriteLine(i);  
}
```

- **Ciclo while:**

```
while (statement){  
    statement;  
}
```

- **Ejemplo:**

```
while (true)  
{  
    System.Console.WriteLine("Loop infinito");  
}
```

- **Switch case:**

```
switch (expression){  
    case value1: // statement sequence  
        break;  
    .  
    .  
    .  
    case valueN: //statement sequence  
        break;  
    default: //statement sequence  
        break;  
}
```

- **Ejemplo:**

```
int option  
switch (option){  
    case 1:  
        Console.WriteLine("1");  
        break;  
    default:  
        Console.WriteLine("default");  
        break;  
}
```

- **Condiciona! if:**

```
if (statement){  
    statement;  
}  
else {  
    statement;  
}
```

- **Ejemplo:**

```
if ( 1 < 2){  
    System.Console.WriteLine("OMG!, 1 < 2");  
}  
else {  
    System.Console.WriteLine("Wait... what?");  
}
```

## 56. Nombrar Aplicaciones "famosas" hechas en C#.

- CodeHub, <https://github.com/CodeHubApp/CodeHub>
- ShadowSocks, <https://github.com/shadowsocks/shadowsocks-windows>
- [Banshee](#) reproductor de archivos multimedia

- [Beagle](#) buscador de información del sistema
- [Colectica](#) estadísticas
- [Docky](#) lanzador de aplicaciones
- [FlashDevelop](#) desarrollo de aplicaciones flash
- [HandBrake](#) transcoder de video
- [KeePass](#), administrador de contraseñas
- [Low Orbit Ion Cannon](#) (LOIC), una aplicación de código abierto de prueba de estrés de red y ataque de denegación de servicio
- [Lphant](#), un cliente [peer-to-peer file sharing](#)
- [Microsoft Visual Studio](#)
- [MonoDevelop](#), an [open source integrated development environment](#).
- [Open Dental](#), a [dental practice management software](#).
- [Pinta](#), an [open-source](#), [cross-platform bitmap](#) editor de imagenes

57. ¿Qué tipos de errores existen en la programación.

Errores lógicos, errores del tipo semánticos, en los que el programa no hace lo que se espera que haga

Errores sintácticos, errores que no siguen las reglas del compilador o intérprete, por lo que no tienen una ejecución exitosa

58. ¿Qué es un controlador de versiones?

Un controlador de versiones nos permite llevar un historial detallado de un proyecto de programación, viendo los cambios hechos a archivos mediante commits, con esto podemos guardar un proyecto en una versión “estable” sin tener que preocuparnos de romperlo/desestabilizarlo con la inclusión de mejoras o actualización de herramientas, también podemos separar y compartir proyectos por medio de ramas y forks.

59. ¿Qué es git?

Git es un sistema de control de versiones gratis de código libre que permite el manejo de archivos, proyectos, etc., de una forma más organizada, rápida y eficiente. Algunas de sus características son:

- Facilidad en el desarrollo no lineal.
- Gestión distribuida.
- Publicaciones de almacenes de información por medio de HTTP, FTP, rsync o ssh.
- Gestión eficiente.

60. ¿Qué es github?

Un servidor para repositorios git.

61. ¿Cuál es la diferencia entre git y github?

Git es un sistema de control de versiones, github es una plataforma para compartir tus proyectos a los que git les da seguimiento

62. ¿Qué es un commit?

Es cuando añadimos un conjunto de cambios a un repositorio usando un controlador de versiones, el commit puede hacerse dirigido a una rama en específico, tomando en cuenta todos los archivos o unos en específico, contiene una descripción y mensaje para identificarlo en el historial, con estos se puede regresar a una versión diferente fácilmente.

## Secciones individuales

### Alexis Brayan López Matías

62. Crea un repositorio en github, y ve subiendo todo el material que vayas haciendo durante el curso, todos los commits se deben hacer desde consola (individual).

<https://github.com/AlexisLM/CSharpCERT>

63. Menciona lenguajes de programación en los que hayas programado y cuál es tu favorito (individual).

He programado en C, Java, Python, PHP, JavaScript, Haskell y Prolog. Mi lenguaje favorito es JavaScript.

64. Empieza el curso de SoloLearn de C#. Terminalo antes del sábado a medianoche, mandar certificado (Individual).

En progreso...

65. Menciona cuál es tu concepto de “profesor” y de “alumno”. ¿Qué relación debe haber entre ellos? (individual).

Profesor es aquella persona que tiene como objetivo transmitir el conocimiento que este tiene a sus alumnos de la mejor forma posible.

Alumno es aquella persona que tiene como propósito aprender a través de un profesor.

Considero que debe haber una relación de respeto desde ambas partes, aunque es posible llevar una convivencia menos formal (si ambas partes están de acuerdo) ya que el ambiente se vuelve más ameno y el aprendizaje se facilita mucho más.

66. ¿Cuánto crees que vale tu tiempo? ¿Por qué? (Individual).

Considero que mi tiempo es muy valioso, y no sólo el mío sino el de cualquier persona ya que todos poseemos un tiempo finito que no debemos desaprovechar.

67. Responde a las siguientes preguntas (Individual)

- ¿Cómo te llamas?

Alexis Brayan López Matías

- ¿Qué carrera estás estudiando?

Ciencias de la computación (Pasante).

- ¿Qué piensas de esta revolución tecnológica que estamos viviendo?

Me agrada bastante ya que muchas de las ideas que hace tiempo se pensaban imposibles o muy lejanas, ahora no lo son. El avance tecnológico abre un sinfín de posibilidades para la mejora del bienestar de todos, aunque es necesario llevar a cabo todo con responsabilidad. Me parece muy interesante los temas de IoT, Computación Cuántica e Inteligencia Artificial porque estos campos prometen mucho a futuro.

- Cuéntame un poco de tu experiencia como programador/desarrollador.

Tengo un buen nivel de Java dado que fue el lenguaje que utilicé durante toda mi carrera. Manejo C a un nivel básico/intermedio y Python a un nivel intermedio.

He utilizado Haskell y Prolog para tareas académicas, aunque Prolog sólo vi la parte introductoria.

También utilicé un poco Unity para el desarrollo de un simulador de aspiradora automática, aunque los conocimientos que tengo son muy básicos.

En el caso de Desarrollo Web tengo conocimientos de HTML5, PHP, CSS3 y JavaScript avanzados. En cuanto a frameworks para Front-End he utilizado Bootstrap y VueJS, y para Back-End he utilizado Laravel.

- ¿Qué tanto esperas aprender en este curso? (Individual).

Sonará a cliché, pero espero aprender mucho de C# ya que, aunque anteriormente lo utilicé de forma muy básica en Unity, considero que no tengo conocimiento de este lenguaje. Además, C# es un lenguaje muy popular en la industria por lo que siempre es bueno aprender algo que de verdad se utiliza en el mundo laboral.

## Ricardo Martínez Ríos

62. Crea un repositorio en github, y ve subiendo todo el material que vayas haciendo durante el curso, todos los commits se deben hacer desde consola (individual).

<https://github.com/M0rn1ngSt4r/CSharpCERT>

63. Menciona lenguajes de programación en los que hayas programado y cuál es tu favorito (individual).

C, C++, C#, Java, Python, Ruby, Go, Coq, Haskell, Bash, mi favorito es Python por la facilidad de escribir, aunque demasiada azúcar sintáctica te hace mal.

64. Empieza el curso de SoloLearn de C#. Terminalo antes del sábado a medianoche, mandar certificado (Individual).

...

65. Menciona cuál es tu concepto de “profesor” y de “alumno”. ¿Qué relación debe haber entre ellos? (individual).

Profesor es alguien que tiene conocimientos extensos o avanzados a comparación de un estudiante sobre temas específicos, y está dispuesto a compartir esa información a estos



estudiantes, un estudiante en cambio es una persona dispuesta a aprender sobre algún tema en específico, sin perder el tiempo propio ni del profesor.

66. ¿Cuánto crees que vale tu tiempo? ¿Por qué? (Individual).

Es algo invaluable, nada te lo regresa (aún), y creo que se debe respetar el de los demás de igual manera.

67. Responde a las siguientes preguntas (Individual)

- ¿Cómo te llamas?

Ricardo Martínez Ríos

- ¿Qué carrera estás estudiando?

Ciencias de la Computación (Terminada).

- ¿Qué piensas de esta revolución tecnológica que estamos viviendo?

Creo que es emocionante, aunque un poco preocupante como puede influir en nuestras vidas de manera destructiva, por el alcance de la influencia de las redes sociales, la pérdida de privacidad y la poca seguridad en los sistemas que ahora son parte integral de nuestras vidas, aún así, espero que los avances tecnológicos que vienen puedan revertir daños ambientales y acabar con la falta de alimento y energía que afectan a nuestra sociedad.

- Cuéntame un poco de tu experiencia como programador/desarrollador.

He utilizado por mucho tiempo Java, desde la prepa, y alguna vez use C# para un proyecto de reproducción de música hace tiempo, y Python lo use mucho al final de la carrera y en el curso actual de becario, aun así no sabría cómo medir el nivel en alguno de los lenguajes que se “usar”, aunque creo que me adapto rápidamente.

- ¿Qué tanto esperas aprender en este curso? (Individual).

Espero ver un poco de conexión a base de datos, aplicaciones de escritorio y tal vez algo de .NET, también algunas de las características nuevas en C# como el tipado dinámico o nueva sintaxis, los “get” y “set” que vimos al inicio no los había visto antes.

## Arturo Castillo Valles

62. Crea un repositorio en github, y ve subiendo todo el material que vayas haciendo durante el curso, todos los commits se deben hacer desde consola (individual).

<https://github.com/acv629>

63. Menciona lenguajes de programación en los que hayas programado y cuál es tu favorito (individual).

- C
- Java
- Javascript
- PHP

- Haskell
- C#
- C++ **Favorito**
- Python
- Lua (poquito)
- Ruby
- Bash
- Powershell

64. Empieza el curso de SoloLearn de C#. Termínalo antes del sábado a media noche, mandar certificado (Individual).

...

65. Menciona cuál es tu concepto de “profesor” y de “alumno”. ¿Qué relación debe haber entre ellos? (individual).

Un profesor es una persona experimentada en el área de estudio y que encamina al alumno. El alumno es alguien dispuesto a obtener conocimientos del área de estudio.

66. ¿Cuánto crees que vale tu tiempo? ¿Por qué? (Individual).

Mucho. Mi tiempo es finito y hay muchas cosas que lo ocupan, por lo que espero que cada actividad que hago sea de provecho.

67. Responde a las siguientes preguntas (Individual)

- ¿Cómo te llamas?

Arturo Castillo Valles

- ¿Qué carrera estás estudiando?

Ciencias de la Computación (Terminada).

- ¿Qué piensas de esta revolución tecnológica que estamos viviendo?

Que necesitamos tener cuidado, y no por cosas de AI y escenarios tipo terminator, si no por la manipulación de los medios de información.

- Cuéntame un poco de tu experiencia como programador/desarrollador.

Tengo experiencia trabajando como desarrollador web freelance desde hace un par de años. He utilizado las siguientes tecnologías de manera extensiva:

- SDL2
- OpenGL (versión 3.0 y 2)
- Nodejs
- Expressjs
- Django
- JSF
- PostgreSQL
- MariaDB
- Maya
- Blender
- Unity3D

- Android
- ¿Qué tanto esperas aprender en este curso? (Individual).  
Sobretudo de Windows Forms, que nunca he usado y .NET.

## Naranjo Robledo Carlos

62. Crea un repositorio en github, y ve subiendo todo el material que vayas haciendo durante el curso, todos los commits se deben hacer desde consola (individual).  
<https://github.com/CarlosNaranjoMx/ProgramacionCSharp>

63. Menciona lenguajes de programación en los que hayas programado y cuál es tu favorito (individual).

- C
- Java
- Javascript
- Haskell
- C++
- Python
- Bash
- Powershell

64. Empieza el curso de SoloLearn de C#. Terminalo antes del sábado a medianoche, mandar certificado (Individual).

65. Menciona cuál es tu concepto de “profesor” y de “alumno”. ¿Qué relación debe haber entre ellos? (individual).

Profesor, persona que facilita el conocimiento sobre un tema a su alumno, el alumno debe poner de su parte para poder aprender

66. ¿Cuánto crees que vale tu tiempo? ¿Por qué? (Individual).

Es muy preciado para mi porque me gustaria gastarmelo en muchisimas cosas que se que ni en toda una vida podría terminar

67. Responde a las siguientes preguntas (Individual)

- ¿Cómo te llamas?

Naranjo Robledo Carlos

- ¿Qué carrera estás estudiando?

Ciencias de la Computación

- ¿Qué piensas de esta revolución tecnológica que estamos viviendo?

Que se le está dando mucho poder de opinión a personas que no les corresponde, porque pienso que no siempre la opinión de la mayoría es la correcta

- Cuéntame un poco de tu experiencia como programador/desarrollador.

- Javascript
- JSF

- ¿Qué tanto esperas aprender en este curso? (Individual).  
C# orientado a seguridad