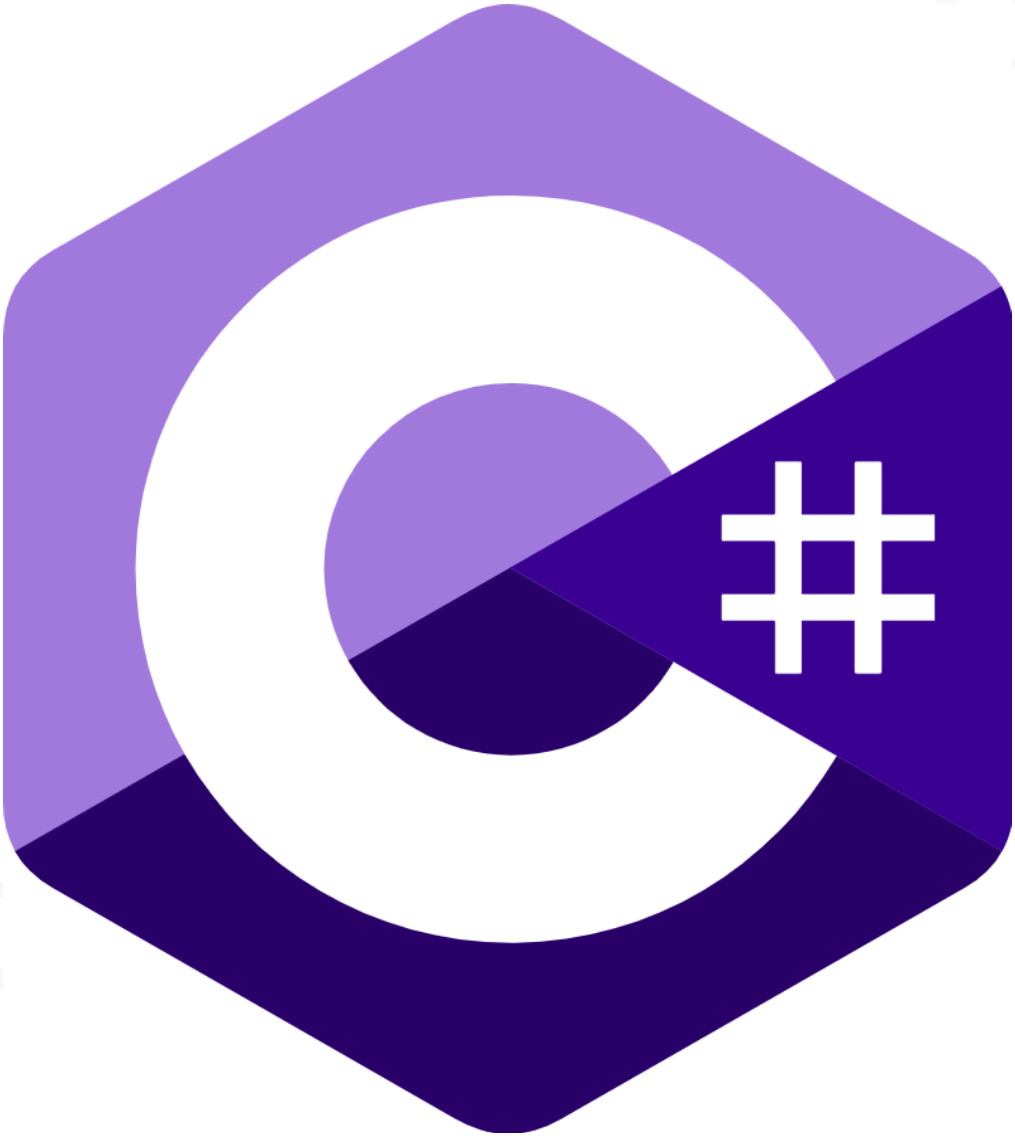
LẬP TRÌNH C# - SOLOLEARN

Student : Green Wolf

Date : 27 / 12 / 2020



1. Basic Concepts

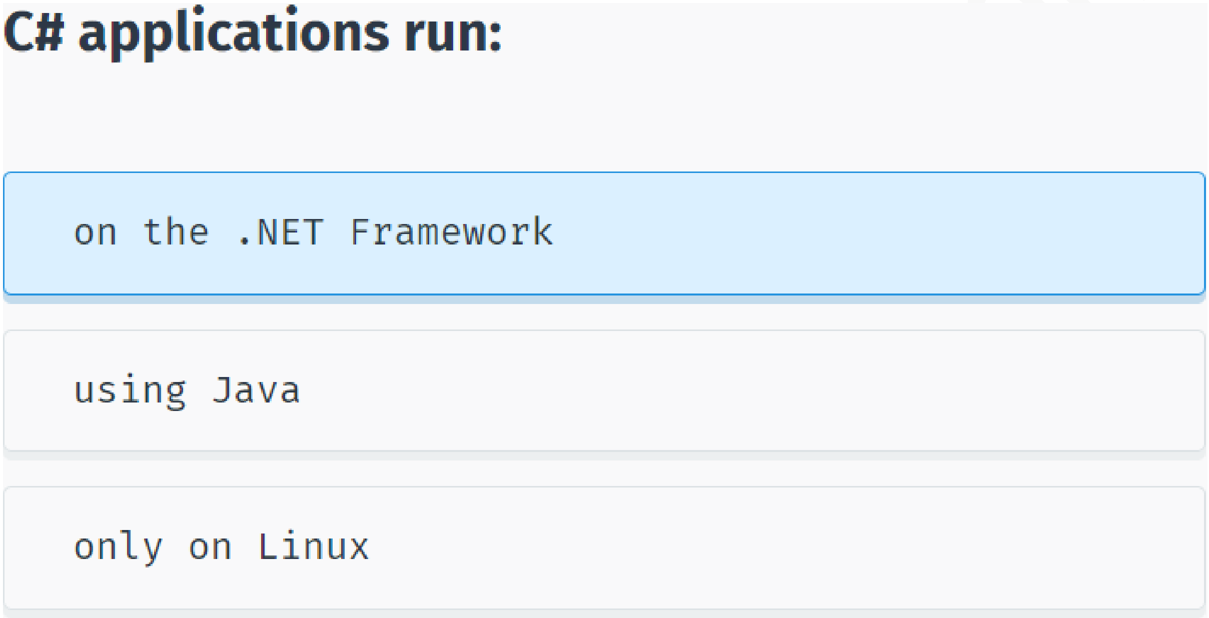
1. What is C#?

Welcome to C#

C# is an elegant object-oriented language that enables developers to build a variety of secure and robust applications that run on the **.NET Framework.**

You can use C# to create Windows applications, Web services, mobile applications, client-server applications, database applications, and much, much more.

You will learn more about these concepts in the upcoming lessons!



C# applications run:

on the .NET Framework

using Java

only on Linux

The .NET Framework

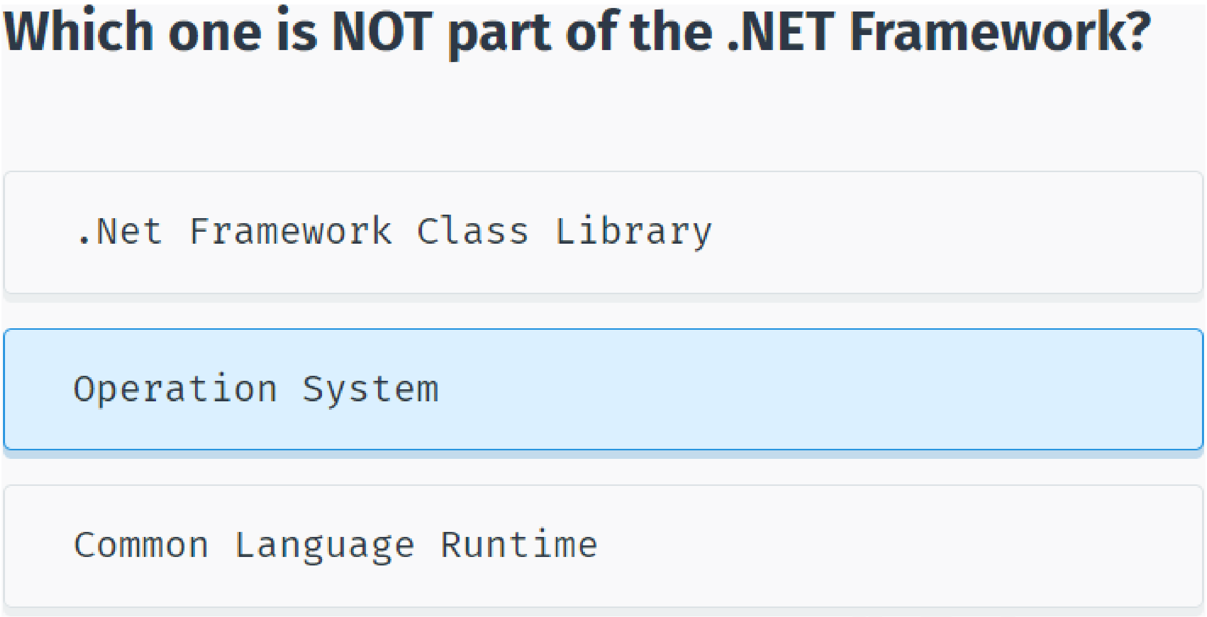
The .NET Framework consists of the **Common Language Runtime (CLR)** and the .NET Framework **class library**.

The **CLR** is the foundation of the .NET Framework. It manages code at execution time, providing core services such as memory management, code accuracy, and many other aspects of your code.

The **class library** is a collection of classes, interfaces, and value types that enable you to accomplish a range of common programming tasks, such as data collection, file access, and working with text.

C# programs use the .NET Framework class library extensively to do common tasks and provide various functionalities.

*These concepts might seem complex, but for now just remember that applications written in C# use the **.NET Framework** and its components.*



Which one is NOT part of the .NET Framework?

.Net Framework Class Library

Operation System

Common Language Runtime

2. Variables

Variables

Programs typically use data to perform tasks.

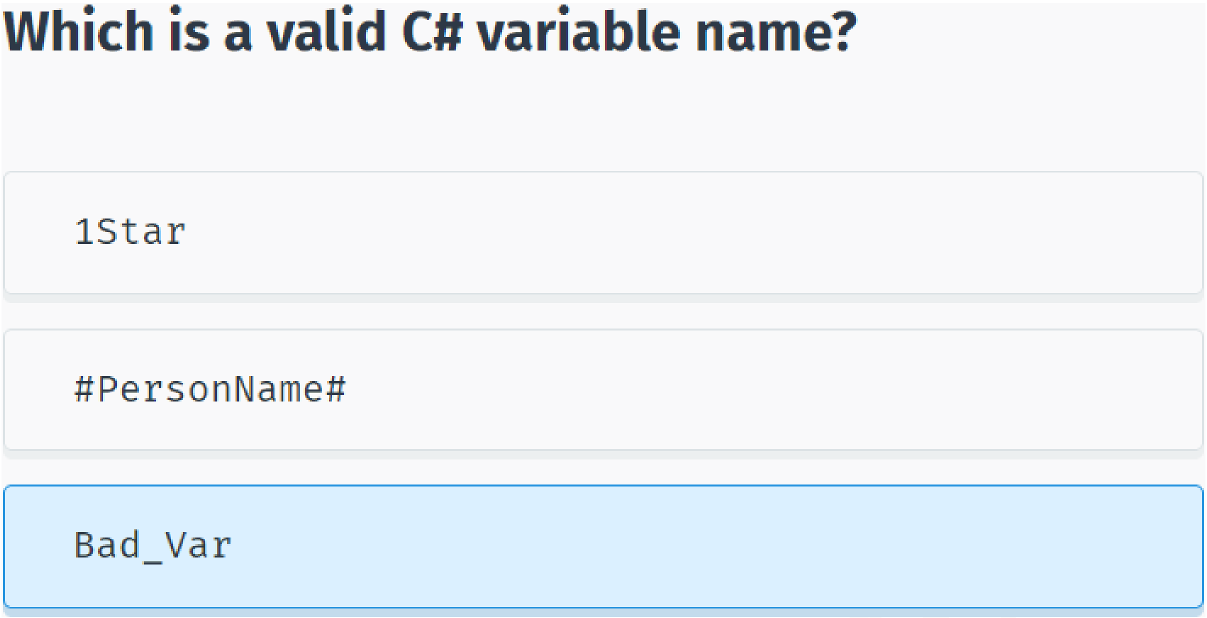
Creating a **variable** reserves a memory location, or a space in memory, for storing values. It is called **variable** because the information stored in that location can be changed when the program is running.

To use a variable, it must first be declared by specifying the **name** and **data type**.

A variable name, also called an **identifier**, can contain letters, numbers and the underscore character (_) and must start with a letter or underscore.

Although the name of a variable can be any set of letters and numbers, the best identifier is descriptive of the data it will contain. This is very important in order to create clear, understandable and readable code!

*For example, **firstName** and **lastName** are good descriptive variable names, while **abc** and **xyz** are not.*



Which is a valid C# variable name?

1Star

#PersonName#

[✓ Bad_Var]

Variable Types

A **data type** defines the information that can be stored in a variable, the size of needed memory and the operations that can be performed with the variable.

For example, to store an integer value (a whole number) in a variable, use the **int** keyword:

```
int myAge;
```

The code above declares a variable named **myAge** of type **integer**.

*A line of code that completes an action is called a statement. Each statement in C# must end with a **semicolon ";"**.*

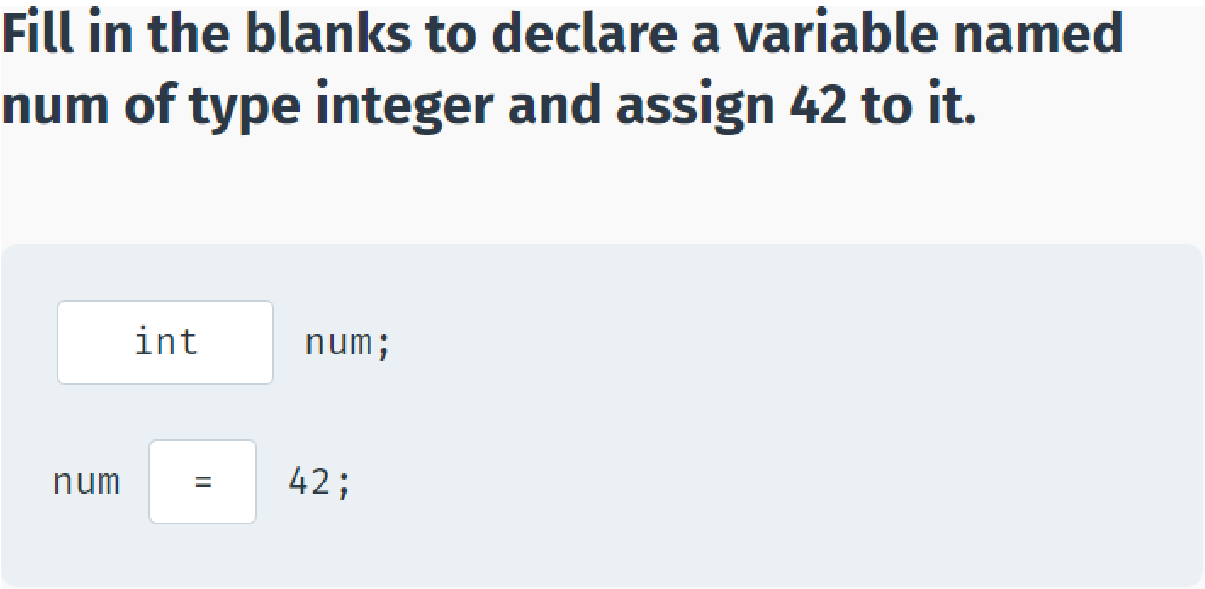
You can assign the value of a variable when you declare it:

```
int myAge = 18;
```

or later in your code:

```
int myAge;
myAge = 18;
```

Remember that you need to declare the variable before using it.



Fill in the blanks to declare a variable named num of type integer and assign 42 to it.

```
int num;

num = 42;
```

Built-in Data Types

There are a number of built-in data types in C#. The most common are:

int - integer.

float - floating point number.

double - double-precision version of float.

char - a single character.

bool - Boolean that can have only one of two values: True or False.

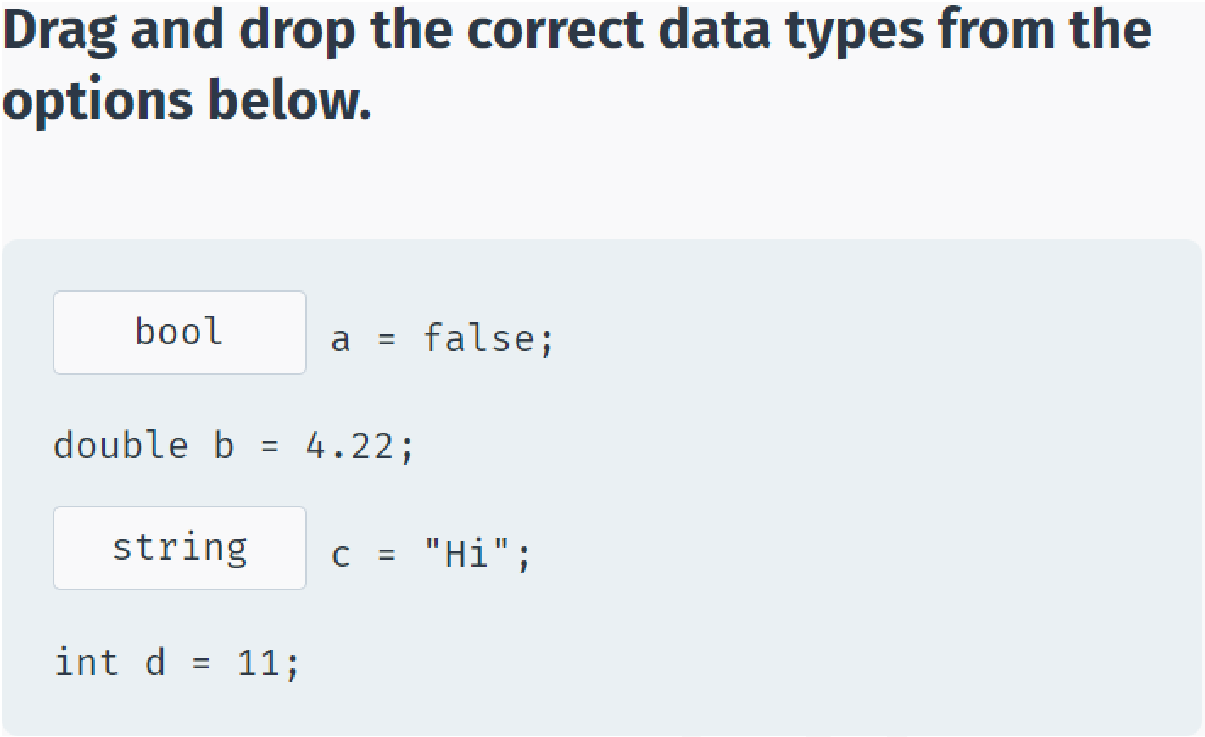
string - a sequence of characters.

The statements below use C# data types:

```
int x = 42;
double pi = 3.14;
char y = 'Z';
bool isOnline = true;
string firstName = "David”;\
```

*Note that **char** values are assigned using single quotes and **string** values require double quotes.*

You will learn how to perform different operations with variables in the upcoming lessons!



Drag and drop the correct data types from the options below.

```
bool    a = false;
double b = 4.22;
string  c = "Hi";
int d = 11;
```

3. Your First C# Program

Your First C# Program

You can run, save, and share your C# codes on our **Code Playground**, without installing any additional software.

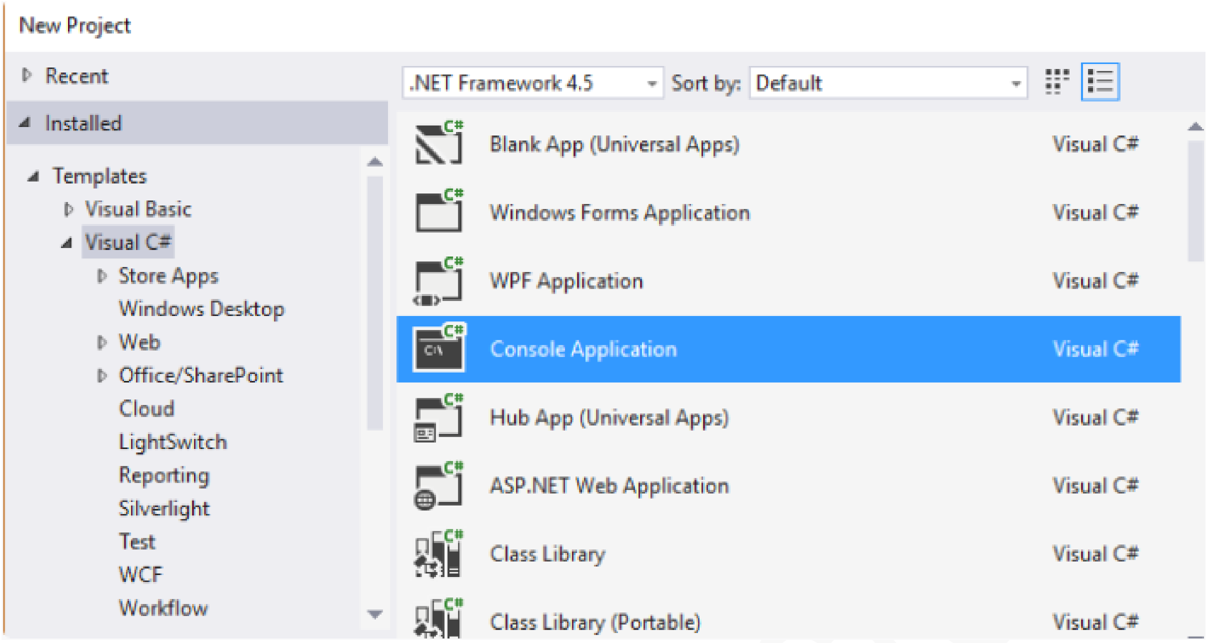
Reference this lesson if you need to install the software on your computer.

To create a C# program, you need to install an integrated development environment (IDE) with coding and debugging tools.

We will be using **Visual Studio Community Edition**, which is available to download for free.

After installing it, choose the default configuration.

Next, click **File->New->Project** and then choose **Console Application** as shown below:



New Project

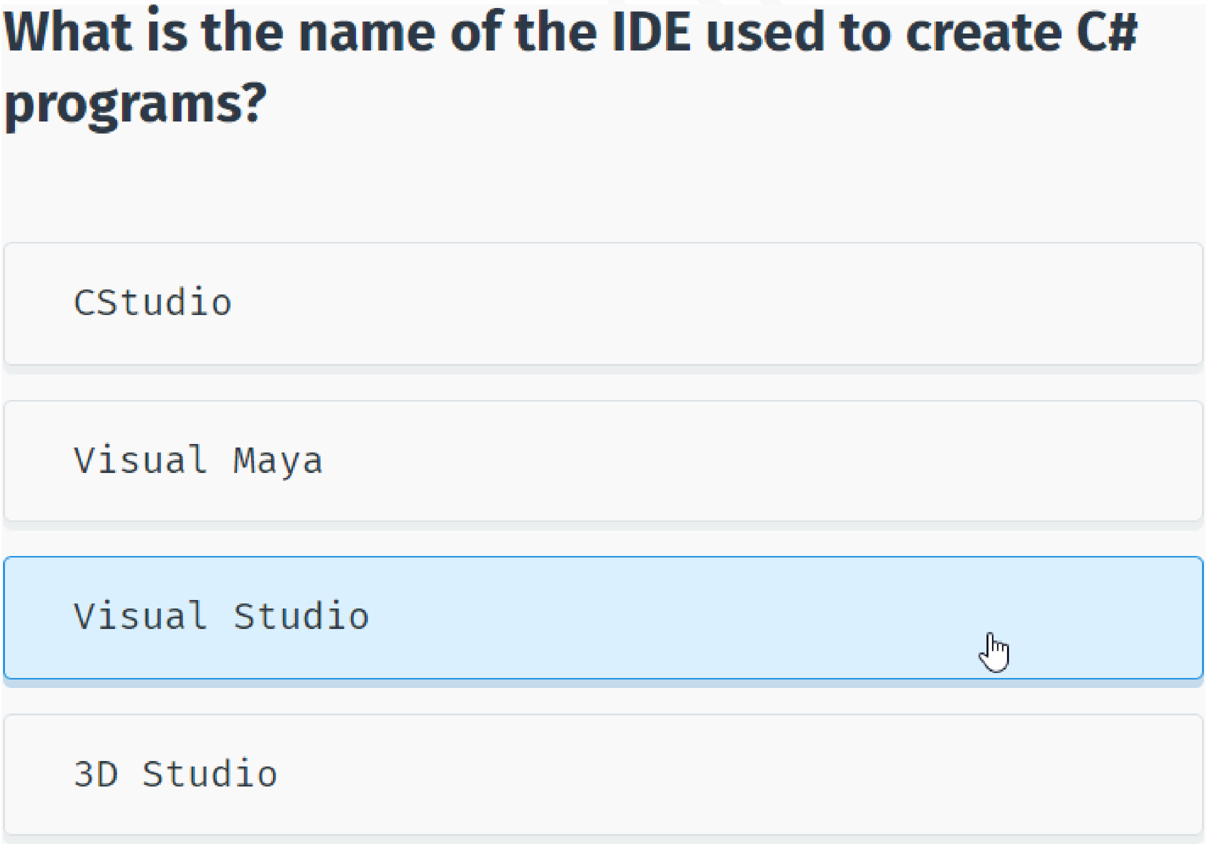
- Recent
- Installed
  - Templates
    - Visual Basic
    - Visual C#
      - Store Apps
      - Windows Desktop
      - Web
      - Office/SharePoint
      - Cloud
      - LightSwitch
      - Reporting
      - Silverlight
      - Test
      - WCF
      - Workflow

.NET Framework 4.5 Sort by: Default

Blank App (Universal Apps)	Visual C#
Windows Forms Application	Visual C#
WPF Application	Visual C#
Console Application	Visual C#
Hub App (Universal Apps)	Visual C#
ASP.NET Web Application	Visual C#
Class Library	Visual C#
Class Library (Portable)	Visual C#

Enter a name for your Project and click OK.

Console application *uses a text-only interface. We chose this type of application to focus on learning the fundamentals of C#.*



What is the name of the IDE used to create C# programs?

CStudio

Visual Maya

Visual Studio

3D Studio

Visual Studio will automatically generate some code for your project:

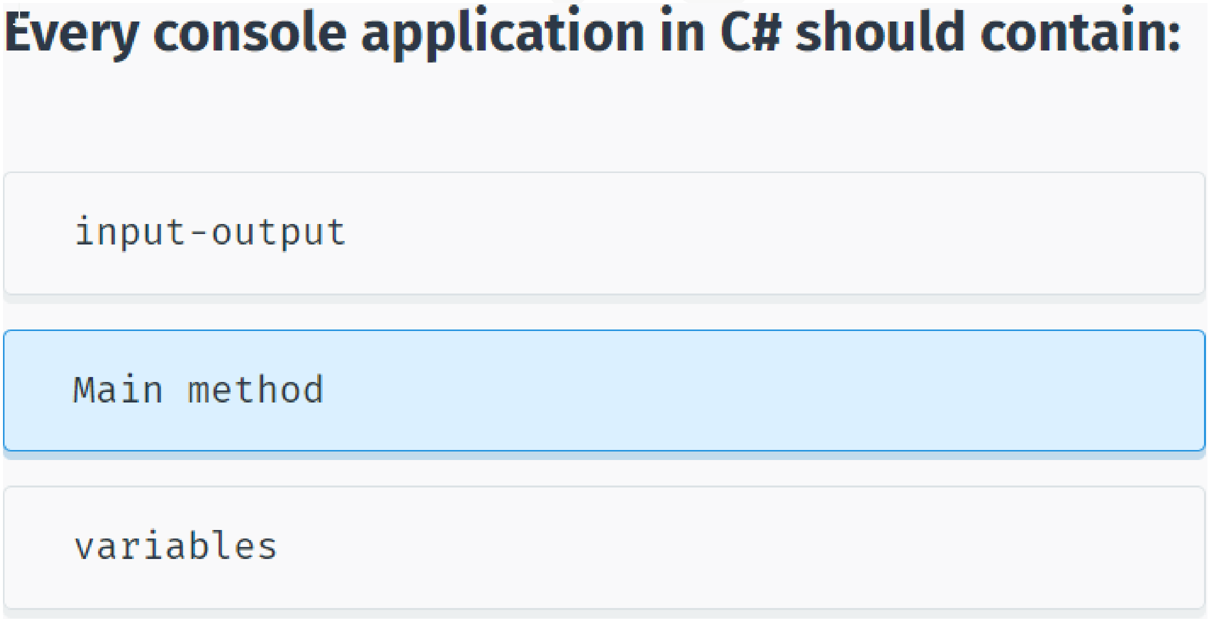
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace SoloLearn
{
    class Program
    {
        static void Main(string[] args)
        {
        }
    }
}
```

You will learn what each of the statements does in the upcoming lessons.

For now, remember that every C# console application must contain a **method (a function) named Main**. Main is the starting point of every application, i.e. the point where our program starts execution from.

We will learn about classes, methods, arguments, and namespaces in the upcoming lessons.



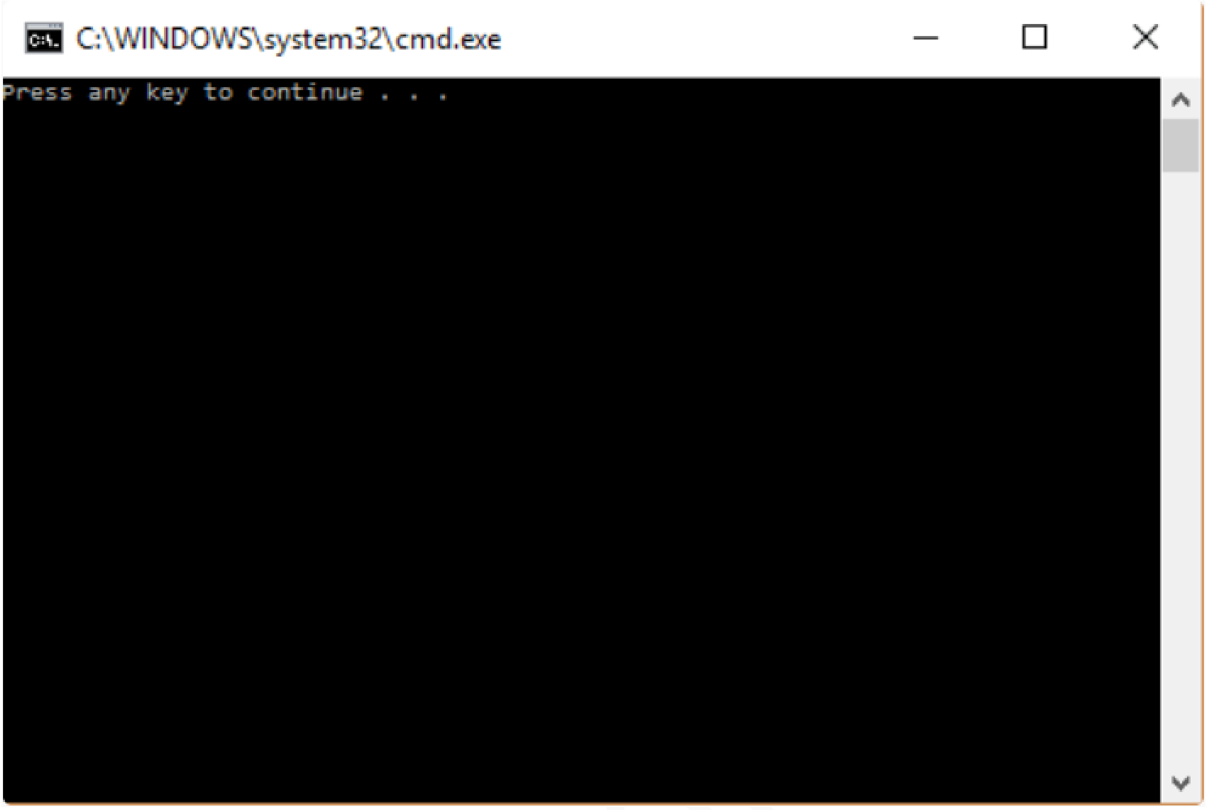
Every console application in C# should contain:

input-output

Main method

variables

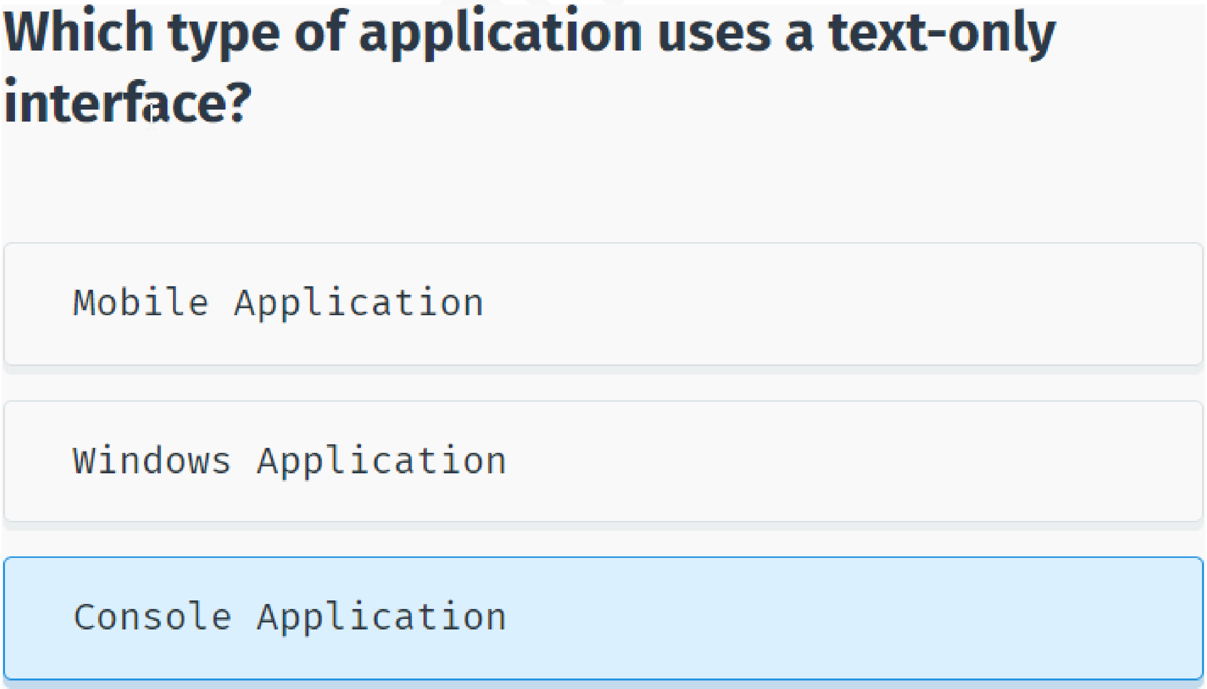
To run your program, press **Ctrl+F5**. You will see the following screen:



```
Press any key to continue . . .
```

This is a console window. As we did not have any statements in our **Main** method, the program just produces a general message. Pressing any key will close the console.

Congratulations, you just created your first C# program.



Which type of application uses a text-only interface?

Mobile Application

Windows Application

[✓ Console Application]

4. Printing Text

2. Conditionals and Loops

3. Methods

4. Classes & Objects

5. Arrays & Strings

6. More On Classes

7. Inheritance & Polymorphism

8. Structs, Enums, Exceptions & Files

9. Generics

Dictionary

Noun

basic concepts : những khái niệm cơ bản

language : ngôn ngữ

applications : những ứng dụng

Web services : dịch vụ web

client-server : máy khách-máy chủ

database : cơ sở dữ liệu

.NET Framework : là một nền tảng lập trình

Common Language Runtime : ngôn ngữ thực thi tổng quát

class library : lớp thư viện

foundation : nền tảng

memory management : quản lý bộ nhớ

collection : bộ sưu tập

aspects of code : các khía cạnh của code

execution time : thời gian thực hiện

accuracy : sự chính xác

core services : những dịch vụ cốt lõi

task : nhiệm vụ

various functionalities : các chức năng khác nhau

components : các thành phần

variables : biến (ví dụ f(x) = x^2 ->x chính là biến trong hàm)

memory location : vị trí bộ nhớ

name : tên

data type : kiểu dữ liệu

underscore character : “_” dấu gạch dưới

semicolon : “;” dấu chấm phẩy

single quotes : “ ‘ “ dấu nháy đơn

information : thông tin

identifier : sự định danh

statement : câu lệnh

operation : sự điều hành

sequence : sự liên tục

character : chữ cái

additional software : phần mềm bổ sung

classes : các lớp

methods : các phương thức

arguments : các đối số

namespaces : không gian tên

general message : thông báo chung

Verb

enable : cho phép, kích hoạt

build : xây dựng

create : chế tạo

run : chạy

consists of : bao gồm

collect : sưu tầm

accomplish : đạt được

file access : truy cập file

reserves : dự trữ

store : lưu trữ

declare : khai báo

specify : xác định

complete : hoàn thành

descriptive of : mô tả về

contain : lưu trữ

perform : biểu diễn

install : cài đặt

Pressing any key : nhấn phím bất kỳ

Adjective

elegant : thanh lịch

variety of : đa dạng

secure : an toàn

robust : mạnh mẽ

upcoming : sắp tới

complex : phức tạp

important : quan trọng

understandable : có thể hiểu được

readable : có thể đọc được

True : đúng

False : sai

different : khác nhau

Other

such as : như là

extensively : một cách chuyên sâu

automatically : một cách tự động

END!