****

**نبذة مختصرة عن البرمجة:**

**البرنامج**:

هو عبارة عن تطبيق سوفت وير تم أنشاؤه لأجراء عملية معينة أو تحقيق هدف معين

مثل مجموعة برامج Office (Word,Excel,PowerPoint,…....,…...,.....) تستخدم لإنشاء وتنفيذ بعض الأعمال المكتبية مثل طباعة أوراق معينه أو كتابة نص لغرض ما أو مشابه الخ..

مثال بسيط :

تخيل انك راجع من شغلك ورايح على بيتك وما فيش حد في البيت يعملك أكلتك أو الغداء وذهبت أنت إلى المطبخ ثم فكرت في إحضار أكلة معينة ووجدت كل ماتريده من أدوات أو خضروات لإحضار تلك الأكلة .

والآن أصبح دورك فأن تفكر كيف تعد هذه الأكلة

ببساطه شديدة تخيل انك قدامك كتاب فيه تعليمات ازاى تحضر الأكلة وأنت الآن كل معليك هو انك تقرأ ثم تنفذ.

الأكلة التي أحضرتها هو..................البرنامج

الغرض من الأكلة إن تشبع جوعك والبرنامج يقوم بحل مشكلة معينه بطريقة متطورة.

الذي قام بإحضار الأكلة هو..............المبرمج

عن طريق استخدام الأدوات والخضروات استطعت إن تحضر أكلتك

الأدوات.....................................جهاز كمبيوتر وبرامج معينه مثل Visual Studio

.........................................................................................................................

بيئة Visual Studio.NET

أولا: Console Application

A project for creating a command-line application

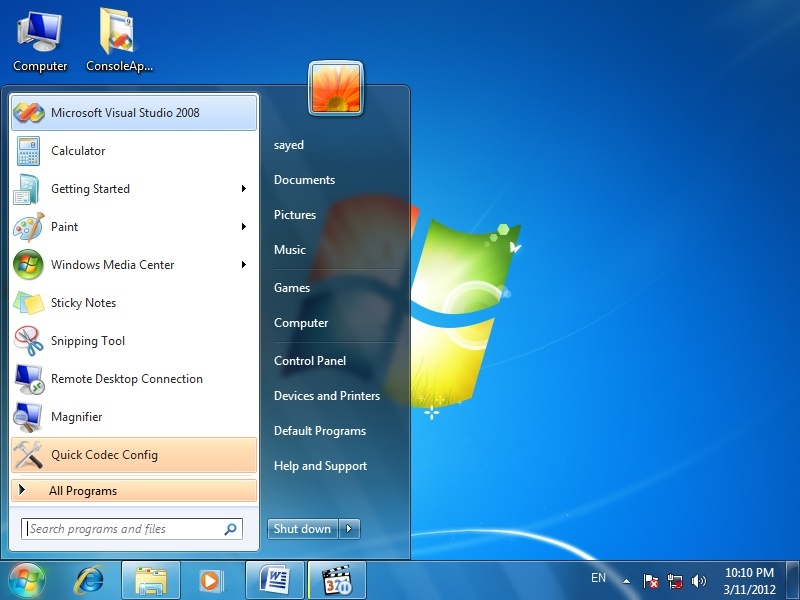
OR the console window is the ***MS-DOS prompt***.

***كلمة DOS اختصار لي* Disk Operating System**

بمعنى انه يستخدم لإنشاء أوامر ***DOS*** أو للتعامل من خلال الشاشة السوداء

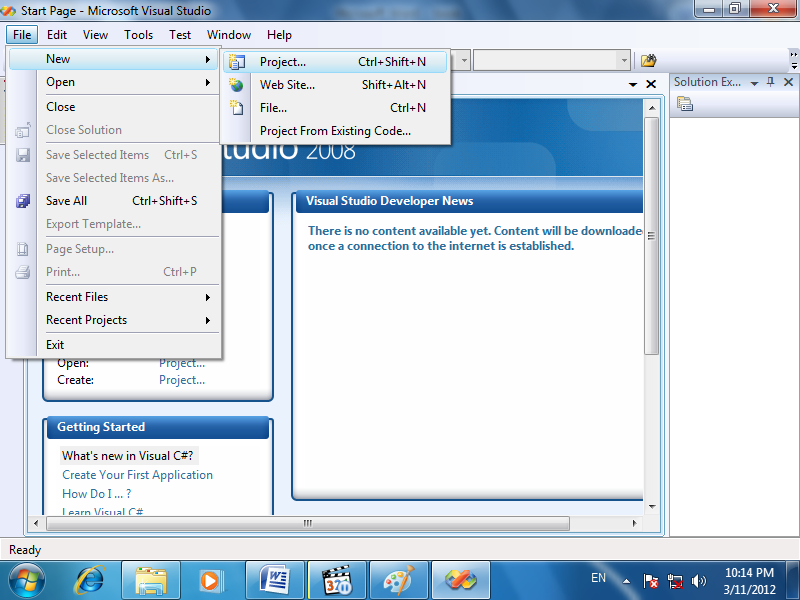
كيف تقوم بعمل برنامج بسيط باستخدام Console

تتبع الخطوات التالية:



1

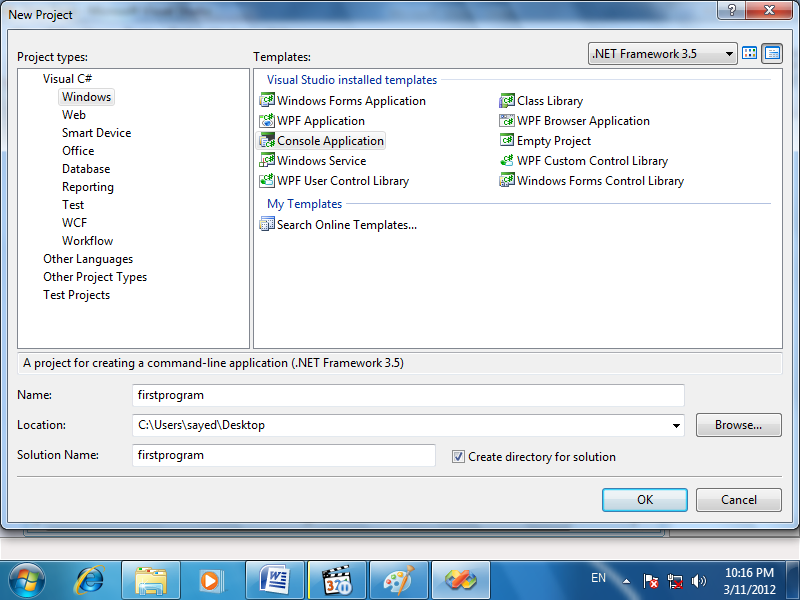
ثم بعد ذلك



(3)أنواع المشاريع التي

يمكن أنشاؤها هنا c#

وبعد كده من الشاشة اللي هاتظهر هامشي حسب تسلسل الأرقام



(6) أسم المشروع

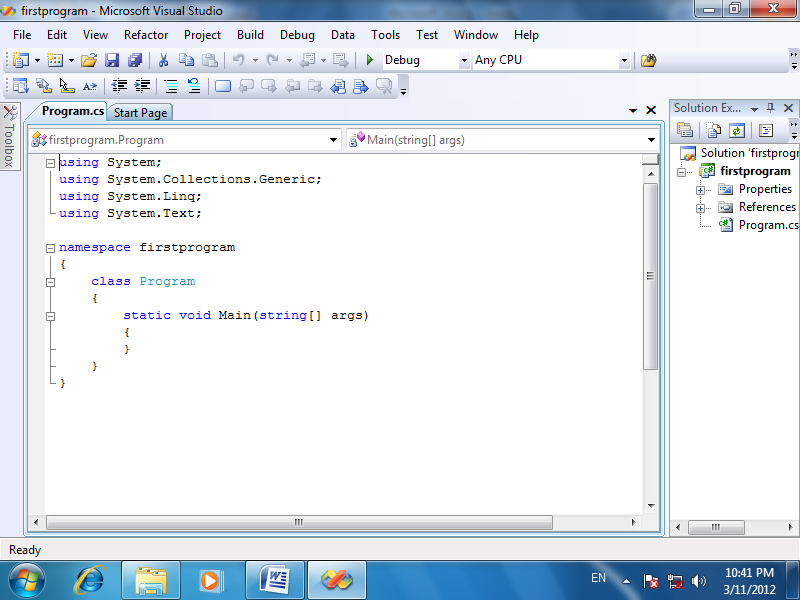
المكان الذي يتم فيه حفظ المشروع

4))

اختار windows

(5) تطبيقات الConsole

والآن تظهر نافذة البرنامج النهائية التي ستقوم فيها بكتابة الكود



مكان الكود

يحتوى عل كل ملفات المشروع

**الآن سنتعرف على كيفية كتابة أكواد بسيطة باستخدام Console**

**جملة الطباعة ( Write Statement)**

تستخدم للطباعة على الشاشة السوداء أو لإظهار معلومات معينة للمستخدم

شكلها:

Console.WriteLine();

تنفذ أمر الطباعة ثم ينتقل المؤشر الى السطر التالى

او

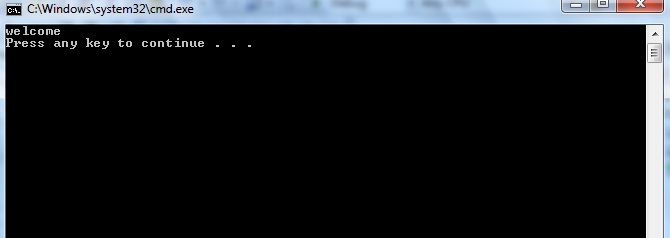
Console.Write();

تنفذ أمر الطباعة ولكن يظل المؤشر واقف على نفس السطر

**مثال:**

Console.WriteLine("welcome");

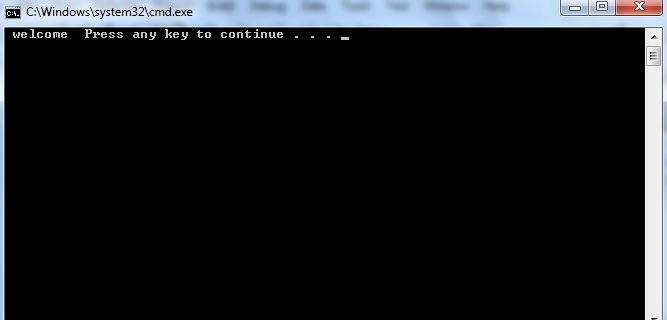
ناتج التنفيذ:

****

**ولكن**

Console.Write("welcome");

ناتج التنفيذ:

****

**ملحوظات:**

* لاحظ إي سطر في الكود لازم ينتهي بعلامة( semicolon) ;
* لاحظ أيضا كلمة Console ده عبارة عن class ومكتوب بالأزرق لأنها كلمة محجوزة او (reserved ward)يعنى ماينفعش تستخدمه انت كمتغير
* أيضاً كلمة WriteLineحرفى(w,l) كابيتل
* فى حالة طباعة حروف او string توضع الجملة بين علامتى

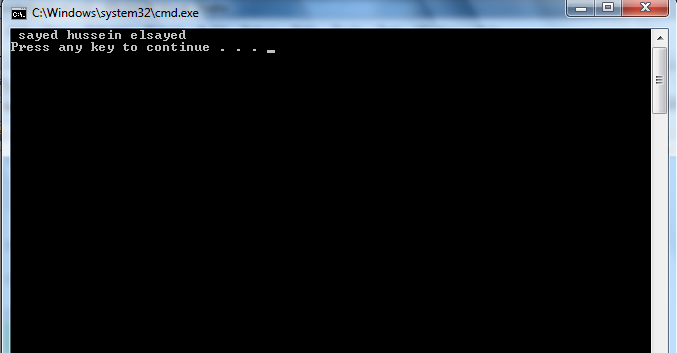
"الجملة " لو ارقام(2125) من غير " "

**مثال:**

استخدم جملة الطباعة لطباعة أسمك؟

Console.WriteLine(" sayed hussein elsayed");

عند عمل تنفيذ للبرنامج من خلال الضغط على (ctrl+f5) ترى هذه الشاشة

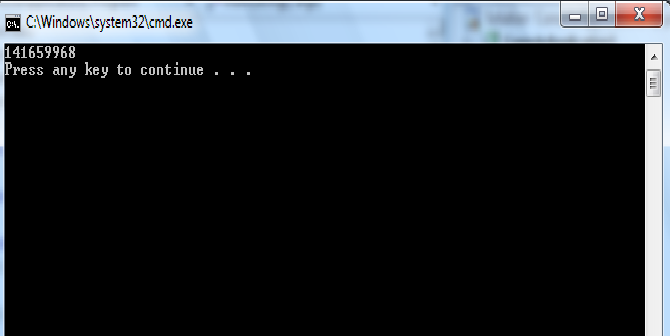


فى حالة لو طباعة ارقام:

**مثال:**

Console.WriteLine(0141659968);

ناتج التنفيذ



**جملة القراءة (Read Statement)**

Console.ReadLine();

تستخدم لأخذ قيم من المستخدم من خلال الشاشة السوداء

مثال:

أنشئ برنامج باستخدام console يطلب من المستخدم أن يدخل اسمه ثم يقوم البرنامج بطباعة الاسم الذي أدخلته هو ...

البرنامج:

Console.WriteLine("please enter your name ...");

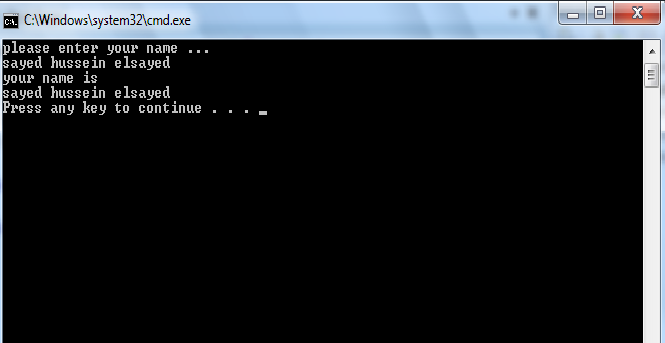
string name =Console.ReadLine();

Console.WriteLine("your name is");

Console.WriteLine( name);

عند التنفيذ يطبع على الشاشة من فضلك ادخل اسمك ويقوم المستخدم بكتابة اسمه وبما ان الاسم من النوع string قمنا بتعريف متغير اسمه name يستقبل مايقوم المستخدم بأدخاله

ناتج التنفيذ هو:



**ملحوظة:**

* جملة القراءة لا تستقبل من المستخدم إلا مدخلات نصية فقط ولكن أذا أردنا استقبال قيم رقميه من المستخدم ما الحل

باستخدام هذا الكود يتم التحويل بين أنواع المتغيرات

int num;

num = Convert.ToInt32(Console.ReadLine());

جملة القراءة (class: convert from data type to another)

**مثال:**

Console.WriteLine(" enter your number");

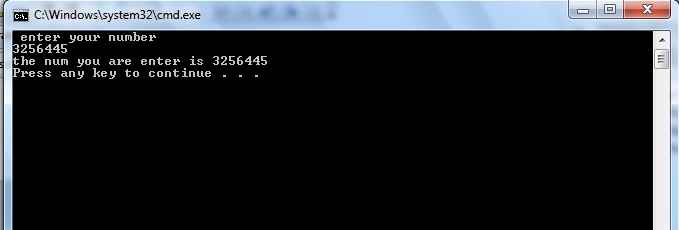
int num;

num = Convert.ToInt32(Console.ReadLine());

Console.Write("the num you are enter is ");

Console.WriteLine(num(;

ناتج التنفيذ:



* عند تعريف أكثر من متغير من النوع int or float أو اى نوع خاص بالأرقام ونريد طباعة تلك الأرقام على الشاشة

**مثال:**

عرف متغيرين من النوع int ثم قم بطباعة على الشاشة الرقم الأول هو .... والرقم الثاني هو .....

int frstnum = 20; int secnum = 30;

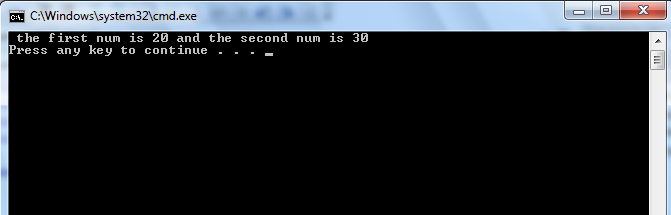
Console.WriteLine(" the first num is {0} and the second num is {1}",frstnum,secnum );

للإشارة إلى أول متغير

للإشارة إلى ثاني متغير

( ويكون ذلك على حسب ترتيب المتغيرات إثناء التعريف)

ناتج التنفيذ:



**Operators**

عند كتابة جملة لعمل مقارنة بين قيمتين أو إعطاء قيمة معينه لمتغير أو لأداء عملية حسابية فأنك تستعمل بعض الرموز التي توضح الفعل الذي تريد حدوثه في الجملة.

من أهم تلك الرموز المستخدمة داخل لغة C# هي .



وسيظهر استخدام تلك الرموز عند دراسة جمل

IF,For,While,Do while

**Types of errors:**

Syntax error:

**هو عبارة عن خطأ يظهر لو في أخطاء في كتابة الكود أو إعطاء متغير قيمة غير صحيحة أو انك تنسى تضع ()بتلاقي الخطأ تحته خط احمر**

**Example:**

int x = "sayed";

or

Console.WriteLine()

Logical error:

هو عبارة عن خطأ غير متوقع اى لا يوجد أخطاء أثناء نتيجة تنفيذ البرنامج ولكن نتيجة البرنامج ما بتكونش النتيجة المطلوبة أو غير متوقعة هنا أنا مثلا عاوز ناتج جمع

**Example:**

int x = 5; int y = 10;

int sum=x+y;

Console.WriteLine(sum-y);

Exception error(runtime error):

ده بيبقا خطأ كبير يخلى البرنامج يهنج مثلا لما بقسم حاجه على صفر أو أقوم بتغيير قيمة متغير null إلى lower case

Example:

string s = null;

s.ToLower();

**في حالة وجود برنامج أو مشروع بيه اكواد كنير ما ينفعش عشان وجود خطأ معين البرنامج كله يقف وعشان كده اخترعوا حاجه اسمها Exception handling فدى عبارة عن طريقة معينة توضع في المكان الذي يشك البرمج بوجود الخطأ به**

**عن طريق استخدام الكود try and catch**

**يتم وضع الكود بين**

**Syntax:**

**Try**

**{code}**

**Catch(exception any\_char)**

**{**

**Cosole.writeline(any\_char);**

**}**

**Example:**

try

{

string s = null;

s.ToLower();

}

catch(Exception ex) عبارة عن متغير يستقبل فيه رسالة الخطأ

{

Console.WriteLine(ex);

}

**ملاحظات:**

* يمكن تغير قيمة اى متغير إثناء البرنامج ولكن يوجد نوع من المتغيرات تظل قيمته ثابتة إثناء البرنامج .

لتعريف متغير من ذلك النوع نضع كلمة const أمام

المتغير

مثال:

const double r = 3.14;

Console.WriteLine(r);

* لزيادة قيمة متغير ما بمعدل 1 فى كل مرة هناك طريقتين

P++ and ++p

ولكن يوجد فرق بينهما

أولا:(p++)

لو p = 4 وعند طباعة p++ يطبع أول مرة 4 ثم بعد ذلك 5 ثم 6 وهكذا

ثانياً:(++p)

لو p = 4 وعند طباعة ++p يطبع أول مرة 5 ثم بعد ذلك 6 ثم 7 وهكذا

**مثال: على p++**

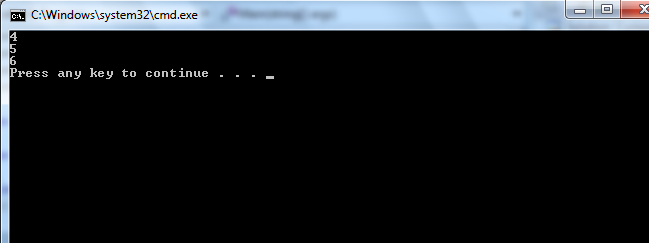
int p = 4;

Console.WriteLine(p++);

Console.WriteLine(p++);

Console.WriteLine(p++);

ناتج التنفيذ:



**مثال: على ++p**

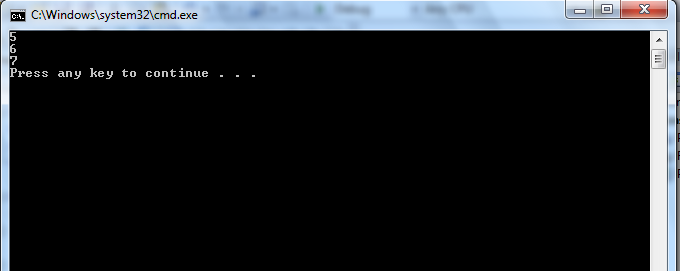
int p = 4;

Console.WriteLine(++p);

Console.WriteLine(++p);

Console.WriteLine(++p);

ناتج التنفيذ:



* أولوية تنفيذ العمليات الحسابية المختلفة:

التنفيذ من اليسار إلى اليمين

من الداخل إلى الخارج بالنسبة إلى الأقواس

الضرب ثم القسمة ثم الجمع أو الطرح

مثال:

x=(3\*(6+9))+(5+(20-5));

+

الخطوات المرقمه هى التى يسير عليها

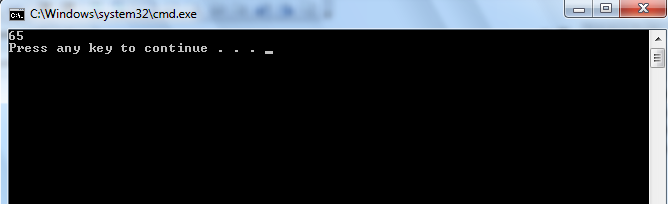
(المترجم من لغة البرمجة الى لغة الجهاز)compiler

ناتج العملية الاولى(9+6)=15 ثم بعد ذلك الخطوة الثانية (3\*15)= 45

ناتج العملية الثالثة(5-20)=15 ناتج العملية الرابعة(15+5)= 20

الخطوة الخامسة(15+45)=65

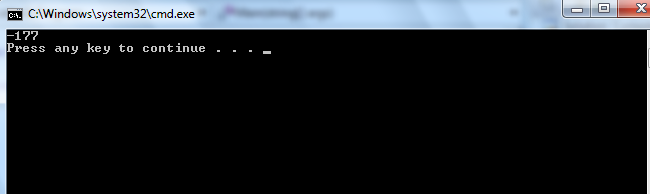
التنفيذ:



y=(6-(3+10/2))-(7\*(9+4\*4));

بأتباع طريقة التسلسل الرياضى تجد

الناتج =-177



**التحويل بين المتغيرات:**

**1-Implicit Conversion**

**2-Explicit Conversion**

**Implicit Conversion**

يتم التحويل أوتوماتيكيا بين الأنواع المختلفة عند ضمان عدم فقد اى معلومات بسبب التحويل

وهو دائما يحصل عند التحويل من النوع الأصغر إلى النوع الأكبر اى وضع قيمة صغيره داخل مخزن كبير

وهو مثل:

Int x =123456 ; // int is 4 byte integer

Long y = x; // implicit conversion to long(لا يتم فقد اى معلومات لأن )

**Explicit Conversion**

يتم أخبار compiler بعملية التحويل

Int x =472 ;

Short z = (short) x ; //explicit conversion to short

في هذا النوع من التحويل يتم فقد بعض المعلومات وذلك لأنه يقوم بوضع قيمة كبيرة داخل مخزن صغير.

يقوم إل compiler بتحويل قيمة المتغير x من النوع int إلى النوع short ثم وضعه في المتغير z .

**مثال :**

int x = 12345;long y;

y = x; // implicity conversion

short z;

int w=10;

z=(short)w; // explicity conversion

/ / هنا نرى كيف يتم فقد معلومات عند التحويل من النوع explicity

double c = 3.5;

int t;

t = (int)c;

Console.WriteLine(t); // يطبع 3 ويفقد .5

ناتج التنفيذ:



**if statement**

من الجمل الشرطية التي تتحكم في مسار البرنامج If تعتبر جملة

**If- statement**

**If else- statement**

**If- else if- statement**

**(If statement)**

**Syntax:**

If (Boolean-expression)

{

Statements

}

Example:

if (x > 50)

{

Console.WriteLine(" value of x is greater than 50");

}

عند أدخال قيمة > 50 يطبع الرسالة غير ذلك لم يفعل شئ

**(If- else statement)**

Example:

if (x > 50)

{

Console.WriteLine(" value of x is greater than 50");

}

else

{

Console.WriteLine("value of x is less than 50");

}

عند أدخال قيمة > 50 يطبع الرسالة الاولى غير كده هايطبع الرسالة الثانية

(If- else if statement)

if (x > 50)

{

Console.WriteLine(" value of x is greater than 50");

}

else if(x<50)

{

Console.WriteLine("value of x is less than 50");

}

else

{

Console.WriteLine("x equal 50");

}

عند أدخال قيمة > 50 يطبع الرسالة الاولى لو أقل يطبع الثانية لو غير كده يطبع الثالثة وهكذا......

**مثال:**

قم بعمل برنامج يطلب من المستخدم إدخال الدرجة التي حصل عليها ثم يطبع التقدير المقابل لتلك الدرجة(راسب,مقبول,جيد,جيد جدا,امتياز) باستخدام جملة if ؟

البرنامج:

Console.WriteLine(" enter ur grade");

int x = Convert.ToInt32(Console.ReadLine());

if (x < 50)

{

Console.WriteLine(" you are fail");

}

else if (x >= 50 && x < 65)

{

Console.WriteLine(" your grade is accept");

}

else if (x >= 65 && x < 75)

{

Console.WriteLine(" your grade is good");

}

else if (x >= 75 && x < 85)

{

Console.WriteLine(" your grade is very good");

}

else if (x >= 85 && x <= 100)

{

Console.WriteLine(" your grade is very excellent");

}

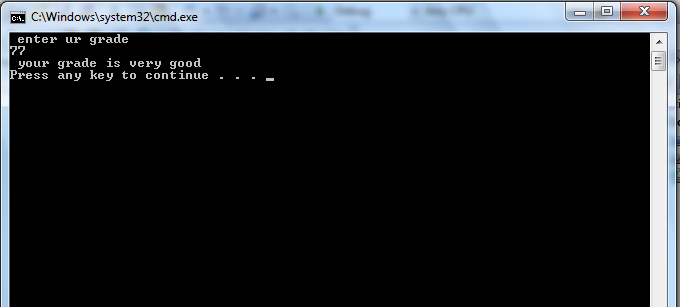
else

{

Console.WriteLine(" not exist");

}

ناتج التنفيذ عند إدخال الدرجة(77)



Switch statement:

It is similar to if statement but in another syntax .

Syntax:

Switch(variable or expression)

{

Case constant 1:

Statements

في حالة تحقق هذه الحالة يخرج من جملة switch Break;

Case constant 2:

Statements

Break;

يمكن وضعه أو لا Default:

Statements

Break;

}

Example:

Write a program by using switch case statement that ask user about the size of a sandwich (large,med,small) and then program write to him the cost of it.

Program:

string size;

Console.WriteLine(" enter sandwich size(large,med,small) you want..and then press enter");

size = Console.ReadLine();

switch(size)

{

case "large":

{

Console.WriteLine("it cost 20$");

break;

}

case"med":

{

Console.WriteLine("it cost 16$");

break;

}

case"small":

{

Console.WriteLine("it cost 10$");

break;

}

default: //(it is like else in if statement)

{

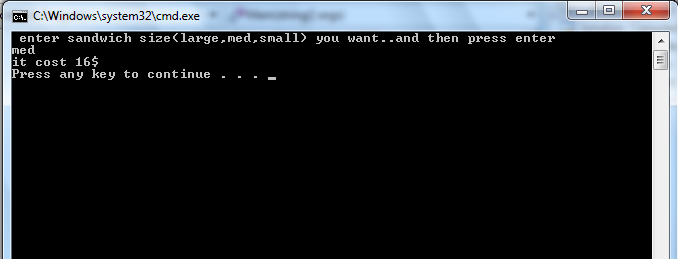
Console.WriteLine(" wrong choice");

break;

}

Output:

If you enter med



**For statement**:

تستخدم لتكرار عدد معين من الكود عدد معين من المرات على حسب الشرط الموجود

It used for repeating block of statements .

Syntax:

For(initial;condition;increment)

{

statements

}

Example:

for (int i = 1; i <= 10;i++ )

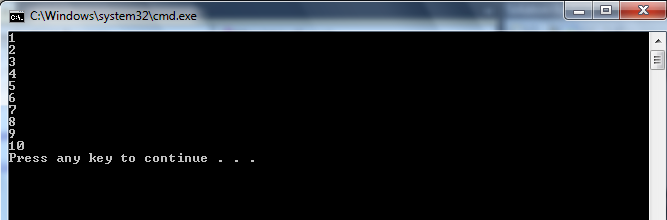
{

// take the first value of i and then print it and soon...

Console.WriteLine (i);

}

Output result:



Excer\_

write a program by using for statement to sum the numbers from 1 to 100 and then print the summation on screen?

While statement:

It tests the condition if the condition is true go to the loop and do it if else get out the while

تتم باختبار شرط معين لو صحيح تتدخل loop وتنفذه لو مش صحيح تخرج خارج جملة while

**Syntax**:

While(Boolean-expression)

{

statements

}

Example:

int i = 0 ;// initial value

while(i<5) // while I <5 enter the loop

{

Console.WriteLine(i);

i++;

}

**Do-while statement:**

Syntax:

الفرق بينها وبين جملة while أنها تقوم بتنفيذ ما بداخلها على الأقل مرة واحده Do

{

statements

}

While(Boolean-expression);

Example:

int x = 1;

do

{

Console.WriteLine(x);

x++;

}

while (x <=5);

الفرق بين break ,continue , goto :

تستخدم تلك الكلمات المحجوزة غالباً داخل جمل التكرار مثل for,while

**أولا: break**

تخبر compiler بالخروج من loop عند الوصول إليها داخل البرنامج

**ثانياً: Continue**

عند الوصول إليها لم يستكمل ما بعدها ويقوم بالرجوع إلى بداية loop

**ثالثاً: goto**

وهى قليلة الاستخدام عند الوصول إليها تقوم بنقل compiler

إلى مكان معين في البرنامج .

Example(break):

for (int i = 1; i <= 10;i++ )

{

Console.WriteLine(i);

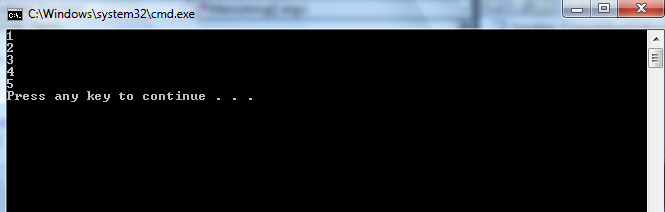
if (i== 5)

{

break;

}

Output:

  
example(continue)

for (int i = 1; i <= 10;i++ )

{

if (i == 5)

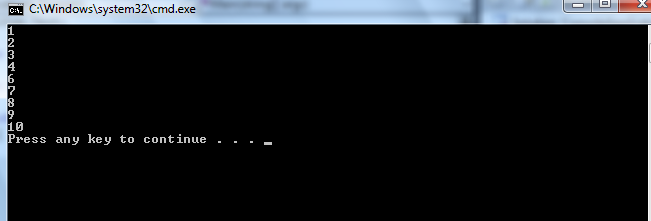
{

continue;

}

Console.WriteLine(i);

}

Output: نلاحظ عدم وجود 5

Example:

for (int i = 1; i <= 10;i++ )

{

if (i == 5)

{

goto p;

}

Console.WriteLine(i);

p:Console.WriteLine("goto statement");

}

**Methods**

عند كتابة كود يقوم بأداء وظيفة معينه مثل الجمع أو الطرح أو اى وظيفة أخرى ونريد استخدام هذا الكود أو هذه الوظيفة أن تنفذ أكثر من مره عند الحاجة إليها فليس من الاحتراف كمبرمج أن تقوم بكتابة هذا الكود في كل مرة أنت محتاج إليه فهنا يأتي دور ال method .

وتوجد أنواع مختلفة لتسمية أو لطريقة تعريف method ولكن سنلجأ إليها فيما بعد:

ملاحظات:

أولا: تكتب ال method خارج main أو داخل class بالشكل التالي.

class Program

{

public static int sum(int a)من النوع int لأنها بترجع قيمة

{

int s = 0;

for (int i = 1; i <= a;i++ )

{

s += i;

}

return s;يظل محتفظ بيها ويعطيه عند استدعاء المسود

}

static void Main(string[] args)

{

Console.WriteLine(" enter any numer");

int x = Convert.ToInt32(Console.ReadLine ());

Console.WriteLine(sum(x)); هنا طريقة استدعاء method لما بترجع حاجة أو لما تكون من الأنواع int ,float,…….

}

}

ويوجد نوع أخر من method هو void وده غالبا يستخدمه لما بكون مش عايز ارجع حاجه مثل جمل الطباعة

class Program

{

public static void print(int a)

{

for (int i = 1; i <= a;i++ )

{

Console.WriteLine("i="+i);

}

}

static void Main(string[] args)

{

Console.WriteLine(" enter any number");

int x = Convert.ToInt32(Console.ReadLine ());

print(x); طريقة استدعاء المسود لما تكون من النوع void

}

}

* عند وجود أكثر من method داخل البرنامج فأنا compiler يفرق بينهما من حيث

اسم method

عدد parameters

وأنواع parameters من حيث data type

عند أعطاء قيم parameters ل method فهناك ثلاث طرق

Passing by value, passing by reference, passing by out:

وتلك الطرق تستخدم أكثر عند استخدام method من النوع void

Example:

class Program

{

public static void increment(int y)// by value

{

y++; لا تستطيع هذه المسود أن ترى مكان الذاكرة لقيمة

}

//=============================================

public static void increment(ref int y)//br ref

{

y++;

}

//=================================================

public static void increament1(out int y)//by out

{

y = 3;

y++;

}

static void Main(string[] args)

{

int x = 5;

increment(x);

Console.WriteLine(x);//x=5

//===========================================

int x2 = 5;

increment(ref x2);

Console.WriteLine(x2);//x=6

//====================================================

int x3 = 5;

increament1(out x3);

Console.WriteLine(x3);//x=4

}

}

**Arrays**

**Syntax:**

**Data type [ ] name = new data type [ size] ;**

**Example:**

int[] x = new int[4];دى عبارة عن عدد العناصر داخل الأراى

**array تستخدم في مواضع كثيرة في البرمجة ويتم استخدامه دائما عند وضع مجموعة من المتغيرات أو مجموعة من العناصر المتشابهة في النوع داخل مكان واحد ويتم ترتيبه بطرق مختلفة.**

**Example:**

**Define an array that allow to the user to enter five elements of string and then print these elements to the user?**

string[] name = new string[5];

for (int i = 0; i < 5; i++)

{

Console.WriteLine("enter name number {0} ", i + 1);

name[i] = Console.ReadLine();

}

for (int i = 0; i < 5; i++)

{

Console.WriteLine("the name number {0} is {1} ", i + 1, name[i]);

}

**يوجد أنواع أخرى من array تستطيع آن تحتوى على أنواع متغيرة من العناصر مثل string,int,…,…**

**وعدد العناصر بها غير محدود اي ليس لها طول معين.**

**Array List:**

**Different data types and unknown it’s size.**

**هي من أنواع المصفوفات التي تحتوي على أنواع متغيره في النوع من المتغيرات أو العناصر وهى أيضا غير معلومة الطول اى أن طولها هو عبارة عن عدد العناصر التى يتم وضعها داخل arraylist**

**Syntax:**

**ArrayList name = new ArrayList();**

**وعشان اعرف arraylist لازم فى الأول استدعى namespace أسمها**

**Collections عن طريق**

using System.Collections;

example:

ArrayList arr = new ArrayList(); //define arraylist

arr.Add("sayed");

arr.Add(014); طريقة الإدخال داخل الاراى لست arraylist\_name.Add(string,int,….,…..)

arr.Add(362.353);

foreach(object i in arr)

{

Console.WriteLine(i); المتغيرات يستخدم مع الأراى لست فى حالة تنوع

}

List:

List have the same data type but unknown it’s size

**Syntax:** the same data type

**List<data type> name = new List<data type>();**

**Example:**

List<string> x = new List<string>();

x.Add("ghada");

x.Add("ali");

x.Add("gdfgdg");

foreach (string i in x)

{

Console.WriteLine(i);

}

**OOP(Object Oriented Programming)**

**طريقة إنشاء أل classes**

كل class يحتوى على خصائص معينه يتميز بها عن غيره وهناك بعض الخصائص المشتركة**.**

من أهم هذه الخصائص:

Example:

**engineer**

**doctor**

class person contain this information

**teacher**

**farmer**

Field:(body,arm,leg,noise,eye,……etc)

Property:(tall,short,size,face,……etc)

Method:(work in accompany,hospital,school,……etc)

Constructor:(initialize the class’s instance members)

To create a new class go under the namespace in your application and .

Example:

class car

{

public int x; لو المتغير هنا مش ببلك ماكنش هاينفع تستخدمه لازم يكون ببلك

}

Here you are create a new class car how can you use it?

In the main program you can take an object from this class ….how?

static void Main(string[] args)

{

car c = new car();// الشكل العام لإنشاء اوبجيكت من اى كلاس

c.x = 4;

Console.WriteLine(c.x);

You can create another object from the class

car r = new car();

r.x = 2;

Console.WriteLine(r.x);

لاحظ انه عند تعريف المتغيرات من النوع ببلك فأى حد فى المين ممكن يستخدمها ويغير عليها طيب لو انا محتاج اخلى الحاج هدى انا اللى اتحكم فيها يعنى ماينفعش المستخدم يدخل اى حاجه وخلاص.

Example:

class age

{

public int urage;

}

………….main……………

age g = new age();// object

Console.WriteLine("pls, enter ur age");

int x = Convert.ToInt32(Console.ReadLine());

g. urage = x;

Console.WriteLine(g. urage);

معنى كده ان المستخدم ممكن يدخل اى سن او اى عمر وانا مثلا مش عايز غير اللى سنهم اكبر من ال 18 وهنا بقا بستخدم حاجه اسمها

Property

Example:

private int urage;//field

public int Age//property

Change the type of age from public to private and property must be public

{

get

{

if (urage >= 18)

{

return urage;

}

else

{

return -1;

}

}

set

{

urage = value;

}

}

}

//====================================================

class Program

{

static void Main(string[] args)

{

age g = new age();

Console.WriteLine("pls, enter ur age");

int x = Convert.ToInt32(Console.ReadLine());

g.Age = x;// take the value from property.

Console.WriteLine(g.Age);

Access modifiers

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| internal | protected | private | public |  |
| yes | no | yes | yes | The same class |
| yes | no | no | yes | The same name space out the class |
| no | no | no | yes | Another name space |
| yes | yes | yes | yes | Derived class in the same name space |
| no | yes | yes | yes | Derived class from another name space |

**Constructor**

It is a method in every class gets the variable initial values by zero or null.

Example:

class example

{

public int x, y;

}

class Program

{

static void Main(string[] args)

{

**constructor**

example ex = new example();

Console.WriteLine(ex.x);

Console.WriteLine(ex.y);

}

If I want to change the values of x or y …..create a constructor and change the values.

Syntax:

Public class\_name()

{

Change the values you want…..

}

Example:

class example

{

public int x, y;

public example()//constructor

{

x = 6; //change the value of x only but y still by zero

}

}

class Program

{

static void Main(string[] args)

{

example ex = new example();

Console.WriteLine(ex.x);

Console.WriteLine(ex.y);

}

The output: 

You can pass values to the constructor like the method how..?

Example:

Write an application that take the name and the age from user and then print them on the screen by using constructor:

class info

{

public string name;

public int age;

public info (string n,int a)// instructor have two parametrs

{

name = n;

age = a;

}

}

class Program

{

static void Main(string[] args)

{

Console.WriteLine("enter ur name...");

string name = Console.ReadLine();

Console.WriteLine("enter ur age...");

int age = Convert.ToInt32(Console.ReadLine());

info d = new info(name, age);

Console.WriteLine("ur name is "+ d.name);

Console.WriteLine("ur age is " + d.age);

}

}

inheritance

Name

Age

City

Department

Name

Age

City

Degree

Name

Age

City

doctor

engineer

employee

That mean the employee have the similar properties of the engineer and the doctor how you do this by programming…?

Example:

namespace inheritance

{

class emp //base class or the father class

{

public string name, city;

public int age;

} that mean the class eng have properties of emp

class eng : emp

{

public string dep;

} that mean the class doc have properties of emp

class doc :emp //u can write class doc:eng

{

public string degree;

}

class Program

{

static void Main(string[] args)

{

eng e = new eng(); //object

e.name = "ghada";

e.age = 28;

e.city = "alex";

e.dep = "software eng.";

Console.WriteLine("data:\n{0}\n{1}\n{2}\n{3}",e.name,e.age,e.city,e.dep);

}

}

}

Abstract

Example:

abstract class a

{

public abstract void s();//if method is abstract it must haven’t abody and also class must be abstract class

}

class b:a //class b inherit from a and allow to make changes

{

public override void s()

{

Console.WriteLine("ghada");

}

}

Example:

Write application by using abstract class animal have the method eat() must be abstract and then create classes(elephent,cat,mouse) use the method eat and make changes.

Application:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace @abstract

{

abstract class animal

{

public abstract void eat();

}

class eleghent : animal

{

public override void eat()

{

Console.WriteLine("elephent eat crosses");

}

}

class cat : animal

{

public override void eat()

{

Console.WriteLine("cat eat fish");

}

class mouse : animal

{

public override void eat()

{

Console.WriteLine("mouse eat cheese");

}

}

class Program

{

static void Main(string[] args)

{

cat a = new cat();

a.eat();

}

}

}

}