

تعلم لغة سي++ مع روح سامية : الدرس الثاني

بمعنا : المتغيرات و أنواع البيانات و نتعلم في هذا الدرس ما يلي :

١. معنى المتغير و كيفية إعلانه.

٢. أنواع المتغيرات الأساسية.

٣. المتغيرات ذات الإشارة و عديمة الإشارة.

البرنامج :

سنبدأ هذا الدرس أيضا بكتابة برنامج ، افتح برنامج code::blocks ثم أنشئ ملفا جديدا و احفظه أولا باسم "variables.cpp" ، فلنكتب (نعم عليك أن تكتب) المثال التالي في الملف و نفهمه :

```
/*=====
      المتغيرات
=====*/
/*=====*/
#include <iostream>
using namespace std;
/*=====*/

// الدالة الرئيسية
int main () {
    int x;           //x الإعلان عن المتغير
    x=100;           //x تعيين قيمة للمتغير
    cout << x << endl; //x طباعة
    x=200;           //x تعيين قيمة للمتغير
    cout << x;       //x طباعة
    return 0;        //إنهاء البرنامج
}
```

قم بتشغيل البرنامج بزر (F9) من لوحة المفاتيح أو من القوائم العلوية بالضغط على خيار build->build and run ، بعد تشغيل البرنامج ستظهر طرفية و ستجد مكتوبا عليها العدد ١٠٠ و تحته العدد ٢٠٠ ، ستجد أيضا عبارات أخرى ، برنامج code::blocks هو الذي وضعها فلا تفكر فيها كثيرا ، ستجدها بعد نهاية كل برنامج تكتبه و تشغله بـ code::blocks.

100
200

الخرج

الشرح :

يقوم برنامجنا بما يلي :

١- يعلن المتغير `x`.

٢- يعين قيمة `١٠٠` للمتغير `x`.

٣- يطبع المتغير `x`.

٤- يعين قيمة `٢٠٠` للمتغير `x`.

٥- يطبع المتغير `x`.

٦- يخرج.

هذا البرنامج يبين الاستخدام الأساسي للمتغير ، المتغير عبارة عن مساحة في الذاكرة يمكن تعيين قيمتها و تغيير هذه القيمة متى نشاء ، لننظر إلى البرنامج سطرا سطرا :

-١

```
/*=====
    المتغيرات
=====*/
```

هذا تعليق متعدد الأسطر مكتوب فيه عنوان الدرس ، حاولت أن أجعله مرتبا قليلا ، ليس أمرا معقدا
أليس كذلك ؟

-٢

```
/*=====*/
#include <iostream>
using namespace std;
/*=====*/
```

على الأغلب تتذكر من الدرس السابق أن هذين السطرين مهمان ، تلاحظ أنني وضعت فوقهما و تحتهما تعليقين متعددي الأسطر (و لكنني لم أضع أي تعليق منهما في أكثر من سطر) ، كل ما في الأمر أنني أردت أن يكون الأمر مرتبا قليلا و يبدو جميلا إلى حد ما ، مرة أخرى أقول لا تفكر كثيرا في هذين السطرين ، سيأتي شرحهما فيما بعد.

-٣

```
//x الدالة الرئيسية  
int main () {
```

بداية الدالة الرئيسية مع تعليق يبين ذلك ، تذكر أن كل الأوامر التي في وسط الدالة الرئيسية هي الأوامر التي ستنفذ.

-٤

```
int x; //إعلان عن المتغير x
```

هذا السطر عبارة عن إعلان عن متغير من نوع **عدد صحيح** معرفه x ، بعد هذا السطر يمكننا أن نعين قيمة للمتغير و نغير هذه القيمة ، بإمكاننا أن نستخدم هذا المتغير في أمور كثيرة مثل الحساب و الطباعة.

-٥

```
x=100; //تعيين قيمة للمتغير x
```

هنا تم تعيين قيمة ١٠٠ للمتغير x ، تستطيع أن تقول بأن x يساوي ١٠٠ ، ما رأيك أن أسألك سؤالاً يوضح ذلك ، كم قيمة x زائدا ١٤ ؟ هذا صحيح !! x زائدا ١٤ يساوي ١١٤ .

-٦

```
cout << x << endl; //طباعة x
```

هذا أمر الطباعة مرة أخرى ، و لكن هذه المرة لم نطبع نصا نكتبه بين علامتي تنصيص ، لقد طبعنا المتغير x ، إذا شاهدت الخرج تجد أن العدد الذي طبع هو ١٠٠ لأن x يساوي ١٠٠ ، تلاحظ أننا وضعنا بعد المتغير x سهمين آخرين و كتبنا بعدهما endl ، تستطيع أن تطبع أي عدد من النصوص و المتغيرات باستخدام أمر cout مع السهمين ، ضع بعد كل شيء تريد أن تطبعه سهمين آخرين ثم الشيء الآخر الذي تريد طباعته ، مثلا تستطيع أن تكتب :

```
cout << "hi " << "my name is TheBSOM" << endl;
```

الكلمة endl تعني سطرا جديدا ، أي أنك إذا استخدمت أمر cout مرة أخرى ستم الطباعة في السطر الثاني و ليس في نفس السطر ، كلما أردت أن تطبع في سطر جديد اكتب أمر الطباعة و اجعل ما تطبعه endl ، احذفها إن أردت و سترى الفرق.

-٧

```
x=200; //تعيين قيمة للمتغير x  
cout << x; //طباعة x
```

هنا نعين قيمة أخرى للمتغير x ، المتغير x يساوي الآن ٢٠٠ ، القيمة القديمة و كانت ١٠٠ لم تعد موجودة ، سأسألك مرة أخرى : كم قيمة x زائدا ١٤ ؟ أصبت !! القيمة الآن ٢١٤ ، بعد تعيين القيمة نقوم بطباعة x مرة أخرى ، تجد أن العدد الذي انطبع هذه المرة في السطر الثاني هو ٢٠٠ ، هذا لأن

x يساوي ٢٠٠ الآن بعد أن عينا قيمة جديدة له.

-٨

```
return 0; //إنهاء البرنامج  
}
```

إنهاء البرنامج و إغلاق الدالة الرئيسية ، ليس شيئا جديدا أليس كذلك ؟

معنى المتغير و كيفية إعلانه :

في الدرس السابق طبعنا عبارة "hello world" ، قد لا يكون برنامج كهذا مفيدا جدا لأنه يطبع عبارة واحدة فقط ، هذه العبارة لا تتغير أبدا ، المتغيرات هي مساحة في الذاكرة يمكن أن نحفظ فيها بيانات معينة ، هذه البيانات يمكن أن نغير قيمتها متى أردنا ، في المثال أعلننا عن متغير من نوع **عدد صحيح** ، نستطيع أن نعين أي قيمة لهذا المتغير و طباعته ، بهذه الطريقة نستطيع أن نطبع أي رقم نريد بحفظ القيمة التي نريد طباعتها في المتغير ثم طباعة المتغير ، مثالنا هذا لا يقدم لنا شيئا مختلفا في النتائج و لكننا سنتعلم إن شاء الله كيفية قراءة البيانات في المستقبل ، نستطيع مثلا أن نقرأ ملفا و نحفظ الأرقام الموجودة فيه بداخل متغير ، بهذه الطريقة نستطيع أن نطبع أي قيمة و يمكن استخراج هذه القيمة من ملف ، طريقة الإعلان عن متغير هي كتابة نوع المتغير ثم المعرف ، في مثالنا كتبنا نوع المتغير **int** ثم المعرف **x** ، نوع المتغير هو نوع البيانات ، مثلا **int** تعني **عددا صحيحا** ، **العدد الصحيح** هو عدد موجب أو سالب لكنه لا يحتوي على كسر عشري ، أي أنك لا تستطيع أن تضيف فاصلة للعدد ، مثلا لا تستطيع أن تكتب **x=12.435** لأنه عدد يحتوي على كسر عشري ، تستطيع أن تكتب **x=100** أو **x=64** أو **x=-23** لأنها كلها أعداد صحيحة ، المعرف هو اسم يرمز للمتغير ، مثلا في الرياضيات نستخدم كثيرا الحرف 'س' كمعرف لمتغير ، نستطيع أن نستخدم معرفا مختلفا ، مثلا نستطيع أن تكتب **int id** أو **int age** أو **int pool_1** ، كلما أردت أن تضع قيمة المتغير في عملية ما (الطباعة مثلا) تكتب المعرف ، مثلا **x << cout** لطباعة قيمة **x** ، إذا أردت اختيار معرف ما لمتغير تريد إعلانه عليك أن تراعي بعض القواعد :

أولا : لا يمكن استخدام أي رمز أو مسافة أو علامة ترقيم في المعرف ، نستطيع فقط استخدام الأحرف (x مثلا) و الأرقام (2 مثلا) و الرمز السفلي (_) ، الجدول التالي يوضح أمثلة عن المعرفات الصحيحة و الخاطئة.

معرف خاطئ	معرف صحيح
وجود علامة ترقيم (فاصلة)	age
وجود علامة ترقيم (نقطتان رأسيان)	x1
وجود رمز حسابي (ناقص)	abc_1
وجود رمز حسابي (يساوي)	k1c_c
وجود مسافة	a_1

ثانيا : لا يمكن أن يبدأ المعرف برقم ، أول رمز من المعرف يجب أن يكون حرفا ، يمكن أيضا أن يبدأ المعرف برمز سفلي (_) و لكن الأفضل عدم استخدام هذا ، الجدول التالي يوضح أمثلة عن المعرفات الصحيحة و الخاطئة :

معرف صحيح	معرف خاطئ	سبب الخطئ
x123	123x	يبدأ برقم
ab1c	1abc	يبدأ برقم
1	1	يبدأ برقم
abc	2abc9	يبدأ برقم
__what	123what	يبدأ برقم

ثالثا : لا يمكن أن يكون المتغير كلمة محجوزة في لغة سي ++ ، يجب أن لا تكون كلمة ذات معنى في لغة سي ++ ، بإمكانك أن تلاحظ الكلمات المحجوزة باللون الأزرق في برنامج code::blocks ، الجدول التالي يوضح الكلمات المحجوزة في لغة سي ++ التي يمنع استخدامها كمعرفات :

asm	auto	bool	break	case	catch
char	class	const	const_cast	continue	default
delete	do	double	dynamic_cast	else	enum
explicit	export	extern	false	float	for
friend	goto	if	inline	int	go
mutable	namespace	new	operator	private	protected
public	register	reinterpret_cast	return	short	signed
sizeof	static	static_cast	struct	switch	template
this	throw	true	try	typedef	typeid
typename	union	unsigned	using	virtual	void
volatile	wchar_t	while	and	and_eq	bitand
bitor, compl	not	not_eq	or	or_eq	xor
xor_eq					

أنواع المتغيرات الأساسية :

أعطينا مثالا عن متغير من نوع **عدد صحيح** ، النوع هو معلومات عن البيانات التي نريد حفظها ، مثلا **عدد صحيح** يعني أننا نستطيع أن نحفظ عددا صحيحا فقط في المتغير ، لا نستطيع أن نحفظ عددا عشريا ، ماذا لو احتجنا أن نحفظ عددا عشريا أو حرفا أو غير ذلك ؟ لكي نقوم بهذا علينا إعلان متغير من النوع الذي نحتاجه ، هنا سنشرح أربعة أنواع من أنواع المتغيرات الأساسية كثيرة الاستخدام و سنذكر المتغيرات الأخرى.

أولا : **الرمز (char)** : **الرمز** هو عبارة عن متغير يمكنك أن تحفظ فيه رمزا واحدا ، الأحرف مثلا تحفظها في متغير من نوع **رمز** ، 'A' ، 'a' و غيرها من الأحرف و حتى الرموز العددية مثل رمز '1' و '2' تستطيع أن تخزنها في متغير من نوع الرمز ، تذكر أن رمز '1' ليس رقم **واحد** و إنما هو رمز فقط ، في البرمجة هناك

فرق بين رمز واحد و عدد **واحد** ، صحيح أننا في العادة نحفظ في متغير الرمز حرفا و لكن لأن الحاسب لا يفهم إلا الأرقام و الأرقام فقط بإمكاننا أيضا أن نخزن في متغير الرمز عددا صحيحا ما بين -١٢٨ و ١٢٧ .

هنا مثال عن متغير من نوع **رمز** نعين قيمته ثم نطبعه :

```
/*=====
الرمز
=====*/
/*=====*/
#include <iostream>
using namespace std;
/*=====*/

//الدالة الرئيسية
int main () {
    char character;           //إعلان المتغير character
    character='A';           //تعيين قيمة للمتغير character
    cout << character << endl; //طباعة character
    character='a';           //تعيين قيمة للمتغير character
    cout << character;       //طباعة character
    return 0;               //إنهاء البرنامج
}
```

ثانيا : **العدد الصحيح (int)** : **العدد الصحيح** هو عبارة عن عدد صحيح ليس فيه كسور ، أي ١ ، ٢ ، ٣ ، وهكذا ، لا يمكن أن تحفظ عددا عشريا مثل ١,٥ أو ٢,٠٠٩ أو ٢٠٣,٤٠١ ، **العدد الصحيح** يمكن أن يحفظ عددا ما بين -٢١٤٧٤٨٣٦٤٨ و ٢١٤٧٤٨٣٦٤٧ ، صحيح أنك تستطيع أن تستخدم متغير الرمز لحفظ الأرقام لكن عادة ما يستخدم العدد الصحيح لهذا ، البرنامج الذي عرضناه كمثال في بداية الدرس يوضح طباعة **عدد صحيح**.

ثالثا : **العدد العشري (float)** : متغير **العدد العشري** هو متغير تستطيع أن تحفظ فيه رقما بكسر عشري ، مثلا ٢,٥ و ٣,٠٠٩٢ و ١٩٨,٢٠٣١ و هكذا ، بإمكانك أن تحفظ في متغير **العدد العشري** عددا ما بين $3,4 \times 10^{28}$ و $-3,4 \times 10^{28}$ ، الأس للعشرة يمكن أن يكون موجبا أو سالبا ، يمكنك حفظ عدد صحيح في المتغير العشري ، هنا مثال عن متغير من نوع **عدد عشري** نعين قيمته ثم نطبعه :

```
/*=====
العدد العشري
=====*/
/*=====*/
#include <iostream>
using namespace std;
/*=====*/

//الدالة الرئيسية
int main () {
    float number;           //الإعلان عن المتغير number
    number=1.982043;        //تعيين قيمة للمتغير number
    cout << number << endl; //طباعة number
    number=1029.4430012;    //تعيين قيمة للمتغير number
    cout << number;         //طباعة number
    return 0;              //إنهاء البرنامج
}
```

رابعا : **المنطقي (bool)** : المتغير **المنطقي** هو متغير تستطيع أن تحفظ فيه إحدى قيمتين : إما **صائب** أو **خاطئ** ، لا تستطيع أن تحفظ فيه أي قيمة أخرى ، هنا مثال عن متغير **منطقي** نعين قيمته ، بإمكانك أن تجرب طباعته لكنني لن أقوم بذلك الآن لأننا سنعود للمتغير المنطقي في دروس لاحقة :

```

/*=====
المتغير المنطقي
=====*/
/*=====*/
#include <iostream>
using namespace std;
/*=====*/

// الدالة الرئيسية
int main () {
    bool logical;           //الإعلان عن المتغير logical
    logical=true;           //تعين قيمة صائب للمتغير logical
    number=false;           //تعين قيمة خاطئ للمتغير logical
    return 0;               //إنهاء البرنامج
}

```

هنا قائمة بأنواع المتغيرات الأساسية في لغة سي++ :

النوع	الوصف	الحجم	نطاق القيم
char	رمز	بايت واحد	رمز واحد أو ١٢٧ إلى -١٢٨
short int	عدد صحيح قصير	بايتان	-٣٢٧٦٨ إلى ٣٢٧٦٧
int	عدد صحيح	٤ بايتات	-٢١٤٧٤٨٣٦٤٨ إلى ٢١٤٧٤٨٣٦٤٧
long int	عدد صحيح طويل	٤ بايتات	-٢١٤٧٤٨٣٦٤٧ إلى ٢١٤٧٤٨٣٦٤٧
long long int	عدد صحيح طويل جدا	٨ بايتات	-٩٢٢٣٣٧٢٠٣٦٨٥٤٧٧٥٨٠٨ إلى ٩٢٢٣٣٧٢٠٣٦٨٥٤٧٧٥٨٠٧
float	عدد عشري	٤ بايتات	$\pm 1.0 \times 10^{-38}$ إلى $\pm 3.4 \times 10^{38}$
double	عدد عشري دقيق	٨ بايتات	$\pm 1.0 \times 10^{-308}$ إلى $\pm 1.7 \times 10^{308}$
long double	عدد عشري دقيق طويل	١٢ بايتات	$\pm 1.0 \times 10^{-4932}$ إلى $\pm 1.1 \times 10^{4932}$
bool	قيمة منطقية	بايت واحد	صائب أو خاطئ
wchar_t	رمز عريض	٤ بايتات	رمز عريض واحد

بعض المتغيرات تشبه بعضها مع اختلاف النطاق و البعض الآخر مشابه تماما لبعض ، الخصائص و الأحجام المذكورة في الجدول السابق هي الأكثر استخداما لكن بعض الأنظمة تستخدم أحجاما مختلفة للمتغيرات ، **العدد الصحيح** و **العدد الصحيح الطويل** مثلا متشابهان تماما لكن بعض الأنظمة تجعل **العدد الصحيح الطويل** أكبر حجما و يتحمل نطاقا أكبر .

المتغيرات ذات الإشارة و عديم الإشارة :

ذكرنا في ما سبق نطاق المتغيرات و أن لها قيمة تقع في نطاق محدود كي تعطي نتيجة خالية من الأخطاء ، تستطيع أن تحدد ما إذا كان المتغير **ذو إشارة** أو **عديم الإشارة** و عندها يتغير النطاق و يكون أقل عدد في النطاق ٠ و يصبح أعلى عدد في النطاق هو أعلى عدد في نطاق المتغير **ذو الإشارة** + القيمة العددية لأقل عدد في نطاق المتغير **ذو الإشارة** ، مثلا متغير **الرمز** يحفظ عددا ما بين -١٢٨ و ١٢٧ ، **الرمز عديم الإشارة** يحفظ

رقما ما بين ٠ و (١٢٧+١٢٨)=٢٥٥ ، نفس الشيء بالنسبة **للعدد الصحيح** ، **العدد الصحيح عديم الإشارة** يحوي عددا ما بين ٠ و ٤٢٩٤٩٦٧٢٩٥ ، لتعريف متغير بدون إشارة تضيف كلمة **unsigned** قبل النوع ، مثلا **unsigned int** **للعدد الصحيح** و **unsigned char** **للمرئ** ، لا يمكن إعلان متغير عشري أو منطقي بدون إشارة بل يجب أن يكون بإشارة ، إذا أردت تعريف متغير بإشارة فتستطيع أن تكتب نوع المتغير مباشرة كما فعلنا في ما سبق و تستطيع أيضا أن تكتب كلمة **signed** قبل نوع المتغير ، مثلا **signed int** و **int** تعني نفس الشيء.

```

unsigned int u; //الإعلان عن متغير من نوع عدد صحيح عديم الإشارة
signed int s; //الإعلان عن متغير من نوع عدد صحيح ذي الإشارة
int i; //الإعلان عن متغير من نفس نوع المتغير s تماما
قيمة المتغيرات تدور إذا زدت عليها أو نقصت عليها إلى قيمة خارج نطاقها ، مثلا ، أقل قيمة في العدد
الصحيح عديم الإشارة هي ٠ ، ماذا لو كتبنا المقطع التالي :

```

```

unsigned int u; //الإعلان عن متغير من نوع عدد صحيح عديم الإشارة
u=-1; //تعيين قيمة خارج نطاق المتغير
ما يحصل هنا أن القيمة المخزنة في المتغير u هي أعلى قيمة في النطاق أي ٤٢٩٤٩٦٧٢٩٥ ، جرب اطبع
قيمة المتغير و ستجد أنها كذلك ، ماذا لو زدنا واحدا على أعلى قيمة و هي ٤٢٩٤٩٦٧٢٩٥ ؟ ستجد أنك تعود
إلى العدد ٠ ، نفس الشيء بالنسبة للمتغيرات ذات الإشارة ، لو حفظنا عدد -٢١٤٧٤٨٣٦٤٩ و هو أقل من أقل
قيمة في النطاق بواحد كما يلي :

```

```

int s; //الإعلان عن متغير من نوع عدد صحيح عديم الإشارة
s=-2147483649; //تعيين قيمة خارج نطاق المتغير
تجد إذا طبعت s أن قيمته أصبحت أعلى قيمة في النطاق و هي ٢١٤٧٤٨٣٦٤٧ ، نفس الشيء إذا حفظت قيمة
٢١٤٧٤٨٣٦٤٨ في المتغير s و طبعته فستجد قيمته أقل قيمة في النطاق و هي -٢١٤٧٤٨٣٦٤٨ .

```

تعلمنا اليوم :

١- معنى المتغير و كيفية إعلانه :

● المتغير مساحة في الذاكرة يمكن أن نحفظ فيها بيانات معينة.

● نعلن عن المتغير بكتابة نوع المتغير ثم مسافة ثم معرف المتغير ، مثلا :

```

int x;

```

٢- أنواع المتغيرات الأساسية و مثلنا على أهمها و هي **العدد الصحيح int** و **العدد العشري float** و

الرمز char و **المنطقي bool**.

٣- المتغيرات ذات الإشارة و عديمة الإشارة.

حاول القيام بالآتي :

- ١- عرف عدة متغيرات من أنواع مختلفة.
- ٢- عين قيما مختلفة لعدة متغيرات.
- ٣- أطبع عدة متغيرات و حاول أن تطبعها بترتيب جميل.