الجزء الرابع

أساسيات وتطبيقات لغة سي

د. عمر زرتي
جامعة الفاتح
طرابلس – ليبيا

معلومات عن المؤلف

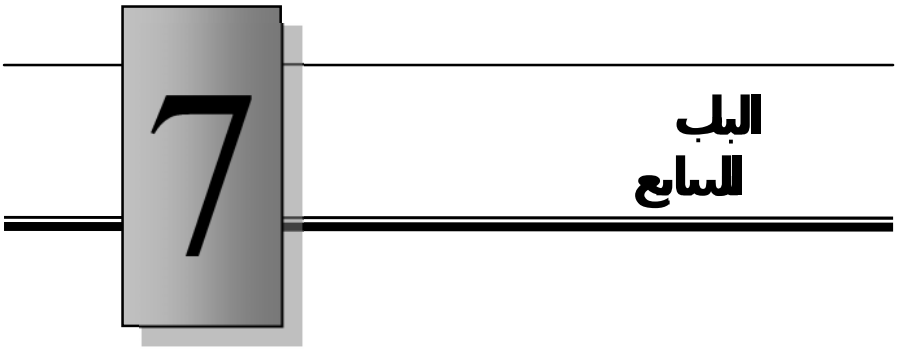
الدكتور **عمر زرتي** omarzarty@yahoo.com هو استاذ شرف بقسم الحاسب الالي بكليه العلوم جامعه الفاتح بطرابلس ليبيا.

تحصل على شهادة البكالوريوس سنة 1970 في الرياضيات من جامعة Colorado School of Mines الواقعة في مدينة Golden من ولايه كولورادو بالولايات المتحدة. تم نال شهادة الماجستير سنة 1972 من جامعه Case Western Reserve الكائنه بمدينه Cleveland بولايه اوهايو . تم تحصل على شهادة الدكتوراة سنة 1976 من جامعة Washington State University بمدينة Pullman بولايه واشنطن.

عند رجوعه الى جامعة الفاتح قام بتاسيس قسم الحاسب الالي وتراس القسم لمدة خمس سنوات، واستمر بعده كعضو هيئة تدريس في مجال (الطرق العددية والبرمجة) ابتداء من درجة (محاضر) وانتهاء إلى درجة (استاذ).

وقد اتجه إلى مجال تاليف وكتابة الكتب المنهجية في المستوى الجامعي والثانوي. ومن اهم كتبه كتاب (الطرق العددية بلغة فورتران) ، كما يعتبر كتابه (اساسيات الحاسوب والبرمجة) من الكتب التي لاقت استحسانا وقبولا واسعا. إضافة إلى دلك فقد قام بتاليف عدة كتب منها (البرمجة بلغة باسكال) و (البرمجة بلغة فورتران) و(اساسيات وتطبيقات لغة سي) ، وتمت طباعة هده الكتب عدة مرات.

وللدكتور عمر زرتي ايضا اهتمام بموضوع (بحوث العمليات) الدي يقوم بتدريسه في مستوى الدراسات العليا، ويقوم بالاشراف على رساله ماجستير في هدا المجال.



اختبـــار البيانات

Data Verification

7.1 مقدمه

في هدا الباب ، ندرس انواع البيانات وطرق اختبارها في لغه سي بمزيد من التفصيل عما درسناه حتى الآن . ويمكن تقسيم هده الانواع إلى الاتي :

1 . النوع الصحيح دو الإشارة (signed int) الدي يمكن ان يكون سالبا او موجبا او صفرا .
2 . النوع الصحيح بدون إشارة (unsigned int) الدي يمكن ان يكون موجبا او صفرا فقط.
3 . النوع الصحيح الطويل long .
4 . النوع الصحيح القصير short .
5 . النوع char وهو ايضا صحيح ولكن يتكون من بايت واحدة فقط .
6 . النوع الكسري (العائم) float دو النقطة العائمة .
7 . النوع المضاعف double وله ضعف دقة النوع العائم .

ويلاحظ ان الرموز characters في لغه سي يشار إليها بارقامها في جدول ASCII (او حسب الشفرة المستخدمة لتمتيل الرموز) . وبالتالي فإن النوع النضيد string ما هو إلا مصفوفه من الارقام تناظر مصفوفه ارقام الرموز في هدا النضيد بجدول الشفرة المستخدمه لتمتيل الرموز .

7.2 النوع char

إذا كان المتغير x من النوع الصحيح بحيث لا يتجاوز

$$\{ -128 \leq x \leq 127 \}$$

فقد يكون من الانسب تخصيص بايت واحدة لهذا المتغير ، فهي كافيه لتخـــزين اي عدد x في النطاق المذكور، وذلك لان اكبر رقم يمكن تمثيله بالإشارة فـــي بايت واحدة هو

$$2^7 - 1$$

اي 127 في النظام العشري .

وبالتالي تمكننا لغة سي من إعلان هذا النوع من المتغيرات الـــذي كثيـــرا مـــا نستخدمه في التطبيقات الإداريه مثل :

1 . المتغير الذي يعبر عن احد الاختيارات في القائمة الرئيسية.
2 . المتغير الذي يعبر عن رقم صفحة في تقرير صغير.
3 . المتغير الذي يرمز لعدد الطلبه في فصل دراسي صغير .

ولكن قد يلاحظ الدارس انه لم تتم في جميع هذه الامثلة الاستفادة مـــن الاعـــداد السالبه، وان الحد الاعلى هو 127 ، اي اننا لم نستفد إلا من 127 رقـــم. فـي الحقيقة ، يمكننا حل هذه المشكلة في لغة سي باستخدام معدل النوع:

unsigned

(اي بدون إشارة) . اي باستطاعتنا تعديل النوع char (او اي نـــوع اخـــر) بواسطة معدل النوع unsigned على النحو التالي :

```
unsigned  char  x ;
```

بهذه الطريقة يصبح النطاق للمتغير x هو :

$$0 \leq x < 255$$

وهو يحتوي على عدد من الارقام يساوى

$$2^8 - 1 = 255$$

اي عدد الارقام التي يمكن تمثيلها في 8 خانات تنائيه (بايت واحدة). والسبب في دلك ان خانه الإشارة اضيفت إلى العدد واصبح لدينا 8 خانات تنائيه (بايت واحدة) بدلا من 7 خانات تنائية في حالة وجود خانة للإشارة .

مثل : مادا يطبع البرنامج التالي ؟ ومادا يحدث إدا تم تغيير الشرط $x < 127$ بالشرط $x <= 127$

```
main()
{
        char x;
        for(x=-127 ; x<127; x++)
                printf(" %d ",x);
}
```

الشكل (7.2.1) النوع char

يطبع البرنامج الارقام من 127- إلى 126 بزيادة 1 في كل مرة. وهو في داخل النطاق المسموح به للمتغير x من النوع char .

لاحظ ان عدم تحديد الإشارة signed او unsigned يعنى في لغة سي انه من النوع الاول اي singned .

إدا تم تغيير الشرط

$$x < 127$$

بالشرط

$$x <= 127$$

فمعنى دلك ان الخروج من الحلقه يتطلب وصول x إلى القيمه 128 وهدا لن يتحقق لان الحد الاقصى للنوع char هو 127 مما يؤدى إلى حلقة لانهائيه .

لاحظ ان جملة التعيين

$$x = 128 ;$$

يقبلها مصرّف لغة سي حتى لو كانت x من النوع char ، ولا يصدر رساله خطا error massage كما نتوقع ، بل يعتبرها

$$x = -128 ;$$

اي ان إضافـة 1 إلى اكبر قيمة لها تاتير البداية من اول قيمة في النطاق المحدد .

مثل (7.2.2) : اعد كتابة الـبرنامج (7.2.1) مع تحـ ـديد x مـــن النـــوع unsigned char، اي بـدون إشارة، وذلك لطباعة الاعداد من 0 إلى 254 .

يبين الشكل (7.2.2) البرنامج المطلوب ، وفيه نلاحظ ما يلي :

1 . عدم تجاوز x النطاق المسموح به للنوع unsigned char ، حيث تتوقف دورة for عندما تصبح قيمة x هي 255 .
2 . إذا استخدمنا النضيد " %c " بدلا من النضيد " %d " في طباعة عدد من النوع char سنحصل على الرمز المقابل لهذا العدد في جدول اسكى .

```
main()
{
        unsigned char x;
        for(x=0;x<=255;x++)
                printf(" %d ",x);
}
```

الشكل (7.2.2) النوع unsigned char

فمثلا إذا كان المتغير x من النوع char فإن جمله التعيين

x = 65 ;

تكافئ الجملة

x = 'A' ;

مثل (7.2.3) : اكتب برنامجا لطباعة جميع الرموز في جدول اسكى بمعـــدل 10 رموز في كل سطر بحيث يطبع الرمز إلى جانب رقمه في الجدول .

```
main()
{       unsigned char k;
        printf("\n ");
        for(k=32; k < 255 ; k++)
        {       if(k%10==0)
                {       printf("\n\n");
                        getch();
                }
                printf("%3d %c  ",k,k);
        }
        printf("%3d %c",255,255);
}
```

الشكل (7.2.3) برنامج جدول اسكى

لاحظ ان الرموز الـقابلة للطباعة في جدول اسـكى تبدا من الـــرقم 32 ، وان اخر رمز قد تمت طباعته في البرنامج كان خارج حلقة for (لمادا ؟) .

7.3 معدلات النوع short و long

يستخدم معدل النوع short لتعديل النوع الصحيح ليصبح دا سعة 2 بايت (اي 16 خانة ثنائية) ، اي انه يكافئ في معظم الحاسبات النوع int ، حيث الحد الاعلى لهدا النوع هو العدد

$$2^{15} - 1 = 32767$$

والحد الادنى هو 32768 - .

إدا كانت هده السعة لا تكفى لتخزين العدد (متلا في تعداد السكان) يمكننا زيادة سعة الكلمة إلى 4 بايت (اي 32 بت) باستخدام المعدل long . اي ان العدد في هدا النوع يمكن ان يصل إلى

$$2^{31} - 1 = 2147483647$$

ملاحظات :

1 . عند قراءة أو طباعة عدد من النوع الصحيح الطويل long int نستخدم النضيد " %ld " وليس " %d " .

2 . عند قراءة أو طباعة عدد من النوع الصحيح غير السالب unsigned int تستخدم النضيد " %u " .

مثل : البرنامج التالي يوضح طريقة إعلان وطباعة متغير صحيح من النوع الطويل long ومتغير من النوع غير السالب .

```
main()
{       unsigned  int x;
        long y;
```

```
        x=65000;
        y=2134567890;
        printf("\n %u ", x);
        printf("\n %ld",y);
}
```

الشكل (7.3.1) طباعه متغير من النوع long والنوع unsigned

7.4 النوع المضاعف double

سبق وان درسنا النوع float ، وهو من النوع المستخدم عادة في تمثيل الاعداد الكسرية، حيث نقسم الكلمة ذات السعة 4 بايت (32 بت) إلى جزءين : الجـــزء الكسرى والاس . ولكن هدا التقسيم قد يؤدي إلى حيز غير كاف لتمثيـــل كـــل الاعداد في الجزء الكسري.

للحصول على دقه اكثر من النوع float يمكن للمبرمج بلغه سي استخدام النوع المضاعف double الذي يوفر سعة 8 بايت .

وطبقا لمرشد توربو سي فإن نطاق العدد من النوع double يصل إلى

1.7E+308

اي ان الاس يصل إلى 308 وهو رقم كبير جدا. اما اصغر رقم فهو - 1.7 E 308 وهو رقم صغير جدا يكاد يكون صفرا .

وإذا ظلت هناك حاجة إلى ارقام اكبر يمكن استخدام النوع long double الذي يسع 10 بايت ، وهذا النوع لا يحتاجه إلا بعض علماء الفلك او الذرة وغيرهـــا من التخصصات الدقيقه .

ملاحظات :

1 . لاحظ أن استخدام النوع double يعطى نتائج أكثر دقة من النوع float ولكن على حساب سرعة التنفيذ.

2 . عـــــند قـــراءة أو طـــباعة عـــدد من النوع double استخدم النضيد " % f ".

3 . عند قراءة أو طباعة عدد من النوع long double استخدم النضيد " % Lf ".

مثل : الـبرنامج بالشكل (7.4.1) يبين الفرق في دقة النـــوعين float و double .

```
main()
{
        long double x;
        float y;
        x=1234567.8901234;
        y=x;
        printf("\n %Lf  %f",x,y);
}
```

الشكل (7.4.1) مقارنة بين النوعين float و double

عند تنفيذ هدا البرنامج نحصل على

1234567.875000 1234567.890123

حيث نلاحظ ان العدد الواقع على اليسار لا يوجد به خطا ، اي انه هـو نفـس العدد المدخل ، اما العدد الاخر ففيه خطا واضح ، حيث هناك فرق بين العـدد الاصلي والمطبوع .

والسبب في ذلك طبعا هو ان الاول من النوع double والثاني من النوع float ، وكما نعلم فإن النوع double دو سعه اكبر من النوع float.

7.5 التحويل من نوع إلى اخر

إدا كانت x من النوع float وكانت y من النوع double واستخدمنا الجملة

```
y = x ;
```

فإن ذلك يؤدى إلى عملية تحويل من النوع float إلى النـوع double، وهـى عملية لا ينتج عنها فقدان لاي اعداد لانها تمت من النوع الصغير (4 بايت) إلى النوع الاكبر (8 بايت) . ولكن إدا كانت العملية عكس ذلك ، اي

```
x = y ;
```

حيث تتم عملية التحويل من double إلى النوع float فينتج عنها فقد لـــبعض الاعداد، لان حيز النوع float لا يكفى للنوع double، ولابد من قطع العدد y بما يكفى لسعة x .

مادا يحدث في هده الحالة عند تنفيد عبارة متل x + y ؟

هل يكون الناتج من النوع float او double ؟

القاعدة العامه هي :

عند وجود نوعين أو أكثر فى عبارة حسابية يكون الناتج من النوع الأكبر

وعلى دلك فإن ناتج x + y سيكون من النوع double .

امـا إدا كانت x و y من نفس النوع فيكون الناتج طبعا من النوع نفسه . مثل :

6 / 5 = 1

حيث نفقد الجزء الكسري لان 6 و5 من النوع الصحيح ، اما إدا كان احـــدهما كسريا والاخر صحيحا فيكون الناتج كسريا ، متل :

6.0 / 5 = 1.2

لدلك إدا كانت k و m النوع الصحيح int واردنا ان يكون ناتج قسمه k علـى m من النوع الكسري ، يمكننا كتابة القسمة على النحو التالي :

(float) k / (float) m

او ببساطه اكتر:

(float) k / m

اي ان k (float) هي عملية تحويل من النوع الصحيح إلى الكسرى float .

مثال (7.5.1) : مادا يطبع البرنامج التالي ؟

```
main()
{
    int k=6,  m=5;
    float x;
    double y=1/3.  ;
    x=k/m;
    printf("\n %f", x);
    printf("\n %f",x+y);
    printf("\n %f", (float)k/m);
}
```

الشكل (7.5.1)

ناتج التنفيذ هو

1.000000
1.333333
1.200000

7.6 اختبار البيانات

من مشاكل استخدام الحاسوب في مختلف التطبيقات مشكلة إدخال بيانات خاطئة للحاسوب ، وهو خطا بشري لا نملك تفاديه بالكامل، و لكن يمكننا التقليل منه قدر الإمكان ، ودلك بالتاكد مما يلي :

1 . القيمة المدخلة تقع في النطاق المتوقع .
2 . القيمة المدخلة من النوع المطابق للمطلوب .

ويمكن للمبرمج بلغة سي ان يستعين بعدد من الدوال الجاهزة المعرفة في ملف ctype.h التي تساعد في التحقق من نوع البيانات كما هو مبين بالجدول (7.6.1).

مثال اكتب برنامجا لقراءة اسم بحيث

1 . لا يزيد عدد حروفه عن 20 حرفا .
2 . لا يحتوى إلا على حروف او فراغات .

اي ان البرنامج لا يقبل إدخال غير الحروف والفراغات ، و إدا خالف مشغل البرنامج دلك ، يصدر إندار ولا يقبل الرمز المدخل .

الدالة	الشرط (لتكون True)
isalnum(c)	c حرف او رقم (من 0 إلى 9)
isalpha(c)	c حرف
isdigit(c)	c رقم (من 0 إلى 9)
iscntrl(c)	c رمز تحكم او إلغاء delete
isascii(c)	c احد رموز جدول اسكى
isprint(c)	c رمز قابل للطباعة
isgraph(c)	تكافئ isprint باستثناء الفراغ
islower(c)	c حرف صغير Lower case
isupper(c)	c حرف كبير Capital
isspace(c)	c فــراغ اوtab او carraige return او newlineاوtab او vertical او form feed
isxdigit(c)	c رقم سادس عشر (اي 0 , 1 , 2 , ... , 9 , A , B , C , D , E , F)

جدول (7.6.1) دوال لاختيار البيانات

```
#include <stdio.h>
#include <ctype.h>
#define N 20
main()
{
        char name[N];
        int i;
        printf("\n Please enter your name-->");
        for(i=0;i<=N-1  ;i++)
        {
                for(;;)
                {       name[i]=getch();
                        if(name[i]==13)break;
                        if( isalpha(name[i]) || name[i]==' ' )
                        {
                                putchar(name[i]);
                                break;
                        }
                        else
                        printf("\a");
                }
                if ( name[i]==13) break;
        }
        name[i]='\0';
        printf("\n The name entered is %s \n ", name);
        getch();
}
```

الشكل (7.6.1) برنامج إدخال الاسم بعد التاكد من صحة الإدخال

ملاحظات عن البرنامج (7.6.1)

1 . ضرورة التوجيه

```
# include  < ctype.h >
```

نظرا لاستعمال الدالة isalpha( )

2 . استخدام حلقة لانهائية لقراءة كل حرف من حروف الاسم لا خروج منها إلا بقراءة حرف او فراغ.

3 . استخدام الدورة الخارجية لقراءة رموز الاسم (الحروف والفراغات) وعددها لا يتجاوز 20 رمزا، ويمكن إنهاء الحلقه بالمفتاح enter (رقمه 13 في جدول اسكى).

4 . لإصدار جرس التنبيه الخطا نستخدم

```
printf( " \a " ) ;
```

5 . وضع رمز الإنهاء ' 0\ ' كاخر رمز في النضيد كما هو متعارف عليه في لغة سي.

6 . اخر جملة في البرنامج

```
getch( ) ;
```

هـ-ي فقط لغرض الإبقاء على شاشة الإخراج إلى حين إدخال اي رمز .

7.7 تمارين

1 . مادا يطبع البرنامج التالي

```
main( )   ;
{   char    k   ;
for (k = 1 ; k < 200 ,k++)
printf( " %d " , k )  ;
}
```

2 . اكتب البرنامج الذي يطبع الحروف الابجدية

a b c d ..... x y z

ودلك بالاستعانة بارقام هده الحروف في جدول اسكى .

3 . ما هو الفرق بيم النوع char والنوع unsigned char ؟

4 . اكتب برنامجا لحساب !n حيث

$$n! = 1 * 2 * 3 * ..... * (n-1) * n$$

ونظرا لان الناتج قد يكون اكبر من سعه النوع الصحيح int ، خاصه إذا كان n عددا كبيرا ، فمن الضروري في هذه الحالة استخدام المعدل long . مـا هـي القيمه n التي يبدا عندها هدا الشرط ضروريا ؟ اختبر برنامجك بـبعض القـيم حتى تجد القيمة الحرجة.

5 . اكتب برنامجا لحساب المجموع

$$sum = 0.001 + 0.001 + ... + 0.001$$

ودلك باستخدام النوع float مرة تم النـوع double مـرة اخـرى . احـسب المجموع لعدد 1000 حد . كم يجب ان يكون الناتج ؟ وماهي القيمة التي يطبعها البرنامج في الحالتين ؟ كيف تفسر النتيجه ؟

6 . يريد مبرمج ان يعد برنامجا للحسابات الجارية في مصرف ، حيث قد يصل رصيد الزبون إلى ملايين الدنانير . ما نوع المتغير الدي يجب ان يستخدمه لهدا الرصيد؟ وإدا كان عدد الزبائن قد يصل إلى مئات الالاف ، ما هو نوع المتغير الدي قد يستخدمه لرقم الزبون المتسلسل؟

7 . إذا كان

i من النوع الصحيح int

n من النوع الصحيح long int

x من النوع العائم float

y من النوع المضاعف double

ما هو نوع الناتج للعبارات التالية :

(a) i + n
(b) i / n
(c) x + n
(d) x* y
(e) n *y
(f) (float) y
(g) (double) y
(r) (float) i / n

8 . ما هي الانواع المناسبة لاختبار البيانات التالية :

(a) رقم الطالب (ويتكون من 5 خانات رقميه)

(b) التاريخ على الصورة : اليوم / الشهر / السنه

حيث اليوم : من 1 إلى 31

الشهر : من 1 إلى 12

السنه : من 1 إلى 9999

(c) الاسم : ويتكون من 20 حرفا (بما في ذلك الفراغات) .

9 . اكتب برنامجا يقوم بقراءة نضيد يتكون من حروف وارقام بحيث :

(1) لا يزيد عدد رموزه عن 15 رمزا .
(2) لا يحتوى إلا على حروف وارقام وفراغات .

وبحيث لا يقبل البرنامج اي رمز اخر ويصدر جرس إندار بذلك .

10 . اكتب برنامجا يقوم بقراءة نضيد تم يقوم بتحويل جميع الحروف الصغيرة (a ,b ,c ,...) إلى حروف كبيرة (A ,B ,C ,...) ويطبع النضيد.
ملاحظة : استخدم الدالة islower واستعن بجدول اسكى في عملية التحويل من حرف صغير إلى حرف كبير