تمرينات للبرمجة بفورترون

قسم الفيزياء
كلية العلوم الدقيقة
جامعة الشهيد حمه لخضر بالوادي

السنة أولى
ماستر فيزياء
2020-2019

Becer Zoubir

كتابة هذه البرامج وتنفيذها تحتاج الى محرر نصوص ومترجم. توجد العديد من الادوات التى تساعدك على كتابة البرامج بشكل منظم والعديد منها تحوى مكتبات للغة ومترجمات compilers وما عليك فقط هو كتابة البرنامج ثم ترجمته وتنفيذه وتتميز هذه الادوات بسهولة اكتشاف الأخطاء. على سبيل المثال نذكر بعض هذه الادوات المستخدمة فى كتابة البرامج الظاهرة في هذه التمرينات: Programers's Notepad +gfortran, geany+gfortran, Plato, Silver Front, Code Blocks .

تمرين 1

نفذ هذا البرنامج في ذهنك وأكتب وضيفة كل سطر بين 2 و 10.

```
1  program First_Test
2  ! My first program
3  J = 1
4  K = 3
5  L = 2*J + K
6  J = 3*J + 2*L
7  K = K + 2
8  L = J + K + L
9  print*, J, K, L
10 stop
11 end program First_Test
```

الحل

عند تنفيذ البرنامج يقوم الحاسوب بما يلي:

السطر 2: يتخطى الحاسوب هذا السطر لأنه تعليق.

السطر 3: يخزن الحاسوب القيمة 1 في مكان في الذاكرة اسمه J. يعني اسناد القيمة 1 الى المتغير J: $1 \longrightarrow$ J.

السطر 4: يخزن الحاسوب القيمة 3 في مكان في الذاكرة اسمه K. يعني اسناد القيمة 1 الى المتغير K: $3 \longrightarrow$ K.

السطر 5: يقوم الحاسوب بحساب قيمة العبارة 2*J + K ثم يخزن النتيجة في مكان في الذاكرة اسمه L. بمعنى:
2*J + K $= 2 \cdot 1 + 3 = 5$ ثم يخزن النتيجة 5 في L : $5 \longrightarrow$ L.

السطر 6: يقوم الحاسوب بحساب قيمة العبارة 2*L + 3*J بإستخدام القيم الحالية لـ J و L ثم يستبدل القيمة الحالية لـ J بالناتج. بمعنى:
3*J + 2*L $= 3 \cdot 1 + 2 \cdot 5 = 13$
ثم يمسح الحاسوب القيمة الحالية لـ J ويستبدلها بـ 13 : $13 \longrightarrow$ ~~1~~ J.

السطر 7: يقوم الحاسوب بزيادة القيمةالحالية لـ K بمقدار 2 : $5 \longrightarrow$ ~~3~~ K.

السطر 8: يقوم الحاسوب بجمع القيم الحالية لـ J و K و L ثم يستبدل القيمة الحالية لـ L بالناتج. بمعنى:
J + K + L $= 13 + 5 + 5 = 23$
ثم يمسح الحاسوب القيمة الحالية لـ L ويستبدلها بـ 23 : $23 \longrightarrow$ ~~5~~ J.

السطر 9: يطبع الحاسوب في الشاشة القيم الحالية لـ J و K و L بالشكل:

```
13        5        23
```

السطر 10: تأمر التعليمة Stop الحاسوب بالتوقف.

تمرين 2

إشرح خطوات تنفيذ البرنامج المقابل وحدد ما يعرضه في الشاشة بعد التنفيذ.

```
1  ! Test of sum and product of
     numbers
2  I=19
3  J=5
4  K=3
5  MS=I+J+K
6  MP=I*J*K
7    print*,I,J,K,MS,MP
8    STOP
9  END
```

الحل

عند تنفيذ البرنامج يقوم الحاسوب بما يلي:

السطر 1: يتخطى الحاسوب هذا السطر لأنه تعليق بسبب وجود علامة التعجب !.

السطر 2: يخزن الحاسوب القيمة 19 في مكان في الذاكرة اسمه I. يعني اسناد القيمة 19 الى المتغير I: $19 \longleftarrow$ I.

السطر 3: يخزن الحاسوب القيمة 5 في مكان في الذاكرة اسمه J. يعني اسناد القيمة 5 الى المتغير J: $5 \longleftarrow$ J.

السطر 4: يخزن الحاسوب القيمة 3 في مكان في الذاكرة اسمه K: $3 \longleftarrow$ K.

السطر 5: يقوم الحاسوب بحساب قيمة العبارة I + J + K على يمين علامة التساوي ثم يخزن النتيجة 27 في مكان في الذاكرة اسمه MS. بمعنى:
$I + J + K = 27$ ثم يخزن النتيجة 27 في MS : $27 \longleftarrow$ MS.

السطر 6: يقوم الحاسوب بحساب قيمة الجداء I*J*K بإستخدام القيم الحالية لـ I و J و K ثم يسند (يخزن) النتيجة 285 للمتغير MP . بمعنى: $285 \longleftarrow$ MP

السطر 7: يطبع الحاسوب في الشاشة القيم المخزنة في المتغيرات I و J و K و MS و MP بهذا الترتيب كما يلي:

```
19        5        3        27        285
```

وهذا لمرة واحدة لانه تم اسخدام التعليمة print*, لمرة واحدة في هذا البرنامج.

السطر 8: تأمر التعليمة Stop الحاسوب بالتوقف. لكن وبسبب وجود التعليمة END بعدها يمكن الاستغناء عنها هنا والاكتفاء بـ END فقط.

تمرين 3

اشرح الغرض من هذا البرنامج.

```
program Test
read*,a,b,c,d,e
s=a+b+c+d+e
print*,a,b,c,d,e
print*,s
end program Test
```

تأمر التعليمة `read*,` بهذه الصياغة البرنامج بقراءة بيانات يختارها المستخدم من لوحة المفاتيح ثم تخزينها في المتغيرات الظاهرة بعدها حسب ترتيب ظهورها. هنا a ثم b ثم c ثم d ثم e. الميزة الاساسية لاستخدام التعليمة `read*,` هي تعميم البرنامج وجعله مستقلا عن بيانات الادخل وبذلك يمكن استخدامه لاي قيم ادخال يرغب بها المستخدم.

تمرين 4

اشرح الغرض من هذا البرنامج.

```
program Area
read*,B,H
Area=(B*H)/2.0
print*,"B=",B,"H=",H,"Area=",Area
end program Area
```

تمرين 5

حدد الغرض من البرنامج المقابل.

```
 program GuessMyAim
 integer :: I
    I=1
99  write(*,*) I
    I=I + 1
    go to 99
 end program GuessMyAim
```

تأمر التعليمة `go to` الحاسوب بالذهاب الى سطر معلم برقم معين (مثل السطر الرابع المعلم بالرقم 99 في هذا البرنامج) ثم مواصلة تنفيذ التعليمات انطلاقا منه نحو بقية السطور الموالية سطرا بسطر.

تمرين 6

إشرح خطوات تنفيذ البرنامج المقابل وحدد ما يعرضه في الشاشة بعد التنفيذ.

```
program GuessWhatIprint
   N=1
31 K=N*N
   print*,N,K
   N=N+2
   if(N<10) go to 31
end program GuessWhatIprint
```

لاحظ ان الجملة N=N+2 لاتعني التساوي بل تعني خذ القيمة الحالية المحفوظة في المتغير N وأضف لها 2 ثم خزن النتيجة في المتغير N. بعبارة اخرى ان القيمة الجديدة لـ N اصبحت تساوي اخر قيمة لـ N مضافا لها 2.

الحل

عند تنفيذ البرنامج يقوم الحاسوب بما يلي:

السطر 2: يخزن الحاسوب القيمة 1 في مكان في الذاكرة اسمه N. يعني اسناد القيمة 1 الى المتغير N: $N \longleftarrow 1$.

السطر 3: يقوم الحاسوب بضرب قيمة N في نفسها واسناد النتيجة الى المتغير K: $K \longleftarrow 1$.

السطر 4: يقوم الحاسوب بطبع القيم الحالية لـ N و K في الشاشة:

```
1        1
```

السطر 5: يقوم الحاسوب بجمع 2 على القيمة الحالية لـ N وتخزين نتيجة الجمع (3) في المكان N (بهذا تُمسح القيمة السابقة لـ N وتستبدل بـ 3) ونعبر عن هذه العملية كما يلي: 3 ⟵ ~~1~~ N.

السطر 6: يجرى السؤال التالي: هل N اقل من 10 ؟.
وحيث أن N=3، فإن الجواب نعم، اي أن الشرط بين القوسين صحيح لذلك يأتمر الحاسوب بالأمر go to ليذهب الى السطر المعلم برقم 31 اي الى السطر الثالث من البرنامج.ويستعد لتفيذ الجملة المكتوبة فيه وفي السطور الموالية له.

السطر 3: يقوم الحاسوب بتربيع القيمة الحالية لـ N ثم تخزين النتيجة (9) في المكان K: 9 ⟵ ~~3~~ K.

السطر 4: يقوم الحاسوب بطبع القيم الحالية لـ N و K:

```
3        9
```

السطر 5: يقوم الحاسوب بجمع 2 على القيمة الحالية لـ N لتصبح 5: 5 ⟵ ~~1~~ ~~3~~ N.

السطر 6: يعود الحاسوب الى السطر المعلم برقم 31 بسبب ان الجواب عن السؤال "هل N اقل من 10 ؟." هو نعم.

السطر 3: يربع الحاسوب القيمة الحالية لـ N ثم تخزن النتيجة (25) في المتغير K: 25 ⟵ ~~1~~ ~~9~~ K.

السطر 4: يطبع الحاسوب القيم الحالية المخزونة في N و K:

```
5        25
```

السطر 5: يقوم الحاسوب برفع القيمة الحالية لـ N بمقدار 2 لتصبح 7: 7 ⟵ ~~1~~ ~~3~~ ~~5~~ N.

السطر 6: يعود الحاسوب الى السطر الثالث بسبب ان N=7<10

السطر 3: يربع الحاسوب القيمة الحالية لـ N و يخزن النتيجة في K: 49 ⟵ ~~1~~ ~~9~~ ~~25~~ K.

السطر 4: يطبع الحاسوب قيم N و K:

```
7        49
```

السطر 5: يزيد الحاسوب قيمة N بمقدار 2: 9 ⟵ ~~1~~ ~~3~~ ~~5~~ ~~7~~ N.

السطر 6: يعود الحاسوب الى السطر الثالث بسبب ان N=9<10

السطر 3: تستبدل قيمة K الحالية بمربع N: 81 ⟵ ~~1~~ ~~9~~ ~~25~~ ~~49~~ K.

السطر 4: تطبع قيم N و K:

```
9        81
```

السطر 5: تزداد قيمة N بمقدار 2: 11 ⟵ ~~1~~ ~~3~~ ~~5~~ ~~7~~ ~~9~~ N.

السطر 6: ينفذ الحاسوب الجمل التالية في البرنامج حيث ان الجواب عن السؤال "هل N=11<10 " هو لا هذه المرة. اي أن الشرط بين القوسين غير محقق وبالتالي لا تتنفذ التعليمة go to.

السطر 7: يتوقف الحاسوب عن التنفيذ بسبب وجود عبارة End program.

وبالتالي تأخذ نتيجة الطبع الاجمالي الشكل التالي:

```
1        1
3        9
5        25
7        49
9        81
```

ومنه يتضح الغرض من البرنامج هو طباعة الاعداد الفردية من 1 الى 9 ومربعاتها جنبا الى جنب بحيث يظهر كل عدد فردي ومربعه في كل سطر.

تمرين 7

توجد مجموعة من الاخطاء في هذا البرنامج قم بتصحيحها.

```
1  program checkme
2  ! there is/are something(s)
     wrong with this program
3  ! find out what and correct it
4  real :: a, b, c
5   a = -10.0
6   b = 23.5
7   c = a*b
8   d = sqrt(a)
9
10  print *,'What is wrong with
      this?
11  print *, 'Nothing now..'
12  print *, c is, c
13  write(*,*) "d = ",d
14  !
15 end program checkme
```

تمرين 8

ماهي النتيجة التي يعرضها البرنامج المقابل.

```
1   program whatIprint
2   integer :: i,n=0
3      do i=1,6
4         n=n+2
5      enddo
6   write(*,*) "n = ",n
7   end program whatIprint
```

تمرين 9

اكتب برنامجا فرترون يطلب من المستخدم ادراج ثلاثة اعداد حقيقية A, B, C ثم يعرض الحلول الحقيقية للمعادلة من الدرجة الثانية : $Ax^2 + Bx + C = 0$.

> ! لاتستخدم متغيرات منطقية. يجب ان يعالج البرنامج حالة ادخال جميع القيم معدومة وكذلك حالة عدم توفر حلول.

تمرين 10: إستكشف

حدد في الحالات التالية خرج البرنامج.

> ! إذا تم تنفيذ التعليمة `cycle` داخل حلقة معينة فان البرنامج **يتخطى** تنفيذ الدورة الحالية ويتقدم مؤشر الحلقة بخطوة واحدة ثم يستأنف تنفيذ الحلقة.

> ! إذا تم تنفيذ التعليمة `exit` داخل حلقة معينة فان البرنامج **يتوقف** عن تنفيذ الحلقة وينتقل البرنامج الى تنفيذ التعليمات ابتداءا من اول سطر بعد الحلقة مباشرة.

.1

```
1  program test
2  integer :: i
3     do i = 1, 5
4        if ( i == 3 ) cycle
5        WRITE (*,*) i
6     enddo
7  write (*,*) 'End of loop!'
8  end program test
```

.2

```
1  program test
2  integer :: i
3     do i = 1, 5
4        if ( i == 3 ) exit
5        WRITE (*,*) i
6     enddo
7  write (*,*) 'End of loop!'
8  end program test
```

الحل

.1

```
1
2
3
5
End of loop!
```

.2

```
1
2
End of loop!
```

تمرين 11: إستكشف

حدد في الحالات التالية عدد مرات (دوران) الحلقة.

```
1. do index = 5, 10
2. do j = 7, 10, -1
3. do index = 1, 10, 10
4. do loop_counter = -2, 10, 2
5. do time = -5, -10, -1
6. do i = -10, -7, -3
```

تمرين 12: إستكشف

حدد في الحالات التالية ما اذا كانت تعليمات فورترون صحيحة او خاطئة . في حالة الخطاء حدد السبب.

.1

```
Loop1: do i = 1, 10
   loop2: do j = 1, 10
      loop3: do i = i, j
         ...
      end do loop3
   end do loop2
end do loop1
```

.2

```
loop1: do i = 1, 10
   loop2: do j = i, 10
      loop3: do k = i, j
         ...
      end do loop3
   end do loop2
end do loop1
```

.3

```
loopx: do i = 1, 10
   ...
      loopy: do j = 1, 10
      ...
      end do loopx
end do loopy
```

تمرين 13: إستكشف

حدد ما ستحويه المتغيرات a,b,c عند نهاية تنفيذ البرامج المقابلة.

```
1  program test
2  character(len=8) :: a, b, c
3  a = 'ABCDEFGHIJ'
4  b = '12345678'
5  c = a(5:7)
6  b(7:8) = a(2:6)
7  end program test
```

```
1  program test
2  character(len=10) :: a
3  character(len=8) :: b, c
4  a = 'ABCDEFGHIJ'
5  b = '12345678'
6  c = a(1:3) // b(4:5) // a(6:8)
7  end program test
```

! إذا كانت العبارة المسندة الى متغير حرفي **أطول** من **طوله** المصرح به تسند العبارة مع اسناد فراغات على يمينه مكان الاماكن المتبقية الشاغرة. في الحالة العكسية **تلغى** الاحرف الفائضة عن الطول المصرح به.

! يمكن **دمج** سلسلتي حروف معا في سلسلة واحدة عن طريق المؤثر // وتعرف العملية باسم **concatenation**.

تمرين 14: إستكشف

حدد في الحالات التالية عدد مرات (دوران) الحلقة.

```
1. do irange = -32768, 32767
2. do j = 100, 1, -10
3. do kount = 2, 3, 4
4. do index = -4, -7
5. do i = -10, 10, 10
6. do i = 10, -2, 0
7. do
```