الفصل الأول

مدخل إلى البرمجة بلغة فورتران

إعداد: م. سناء مهدي

Time	Subject	الموضوع
week	**1.** *Introduction to programming in Fortran* • *The intrinsic data types* • *Arithmetic operators* • *Relational operators* • *Simple Input and output*	1- مدخل إلى البرمجة بلغة فورتران • أنواع المعطيات • العمليات الحسابية • عمليات المقارنة • جمل الإدخال والإخراج
week	**2.** *Control constructs* • *If constructs* • *The case constructs* • *The DO constructs*	2- تركيبات التحكم • تركيبة الانتقاء الشرطية • تركيبة الحالات • تركيبة الدوران
week	**3.** *Procedures* • *External and internal procedures* • *Functions* • *Subroutine*	**3-** البرامج الفرعية • البرامج الفرعية الداخلية والخارجية • الدوال • الروتين
week	**4.** *Arrays* • *Arrays declaration* • *Dimension statement* • *Reading and writing of the arrays* • *Data statement* • *Intrinsic operators* • *Subprogram and array*	4- المصفوفات • تعريف المصفوفات • جملة البعد • قراءة وطباعة المصفوفات • جملة البيانات • إجراء العمليات الحسابية • استخدام البرامج الفرعية مع المصفوفات
week	**5.** *Numerical differentiation*	**5-** التفاضل العددي
week	**6.** *Numerical integration* • *Trapezoidal Rule* • *Simpson's Rule*	**6-** التكامل العددي • التكامل بطريقة شبه المنحرف • التكامل بطريقة سمبسون

المصادر ***references***

- فورتران 77 مدخل إلى برمجة الحاسبات د.محمد زكي محمد نبيل خليل
- المرجع الأساس في برمجة وتطبيقات فورتران 90 د. عوض منصور د. محمود أباظة
- Fortran 90 for scientists & engineering Brian D Hahn

الفصـــــل الأول

المقدمــة :- ***Introduction***

Programming Language

The particular set of rules for coding the instructions to computer called a programming language.

لغة البرمجة : هي مجموعة خاصة من القوانين لأوامر مقننة للحاسبة.

* هنالك العديد من اللغات مثل لغة فورتران وبيسك وباسكال و C^{++}.

Fortran, which stands for **For**mula **Tran**slation was the first "high level" programming language.

هنالك نوعين من اللغات هي :-

1-Low level language

- لغة الحاسب machine language

هي اللغة التي يستخدمها الحاسب في سائر أجهزته الداخلية , ويتعامل بها لمعالجة المعطيات والمعلومات الواردة إليه عبر أجهزة الإدخال والإخراج وكذلك لإخراج النتائج.

- لغة الرموز symbolic language

تتكون من مجموعة من الرموز الاصطلاحية التي تؤدي غرض معين وتمر بمراحل وهي :-

Source program → compiler → object program

(البرنامج المصدري) (المترجم) (البرنامج الهدف)

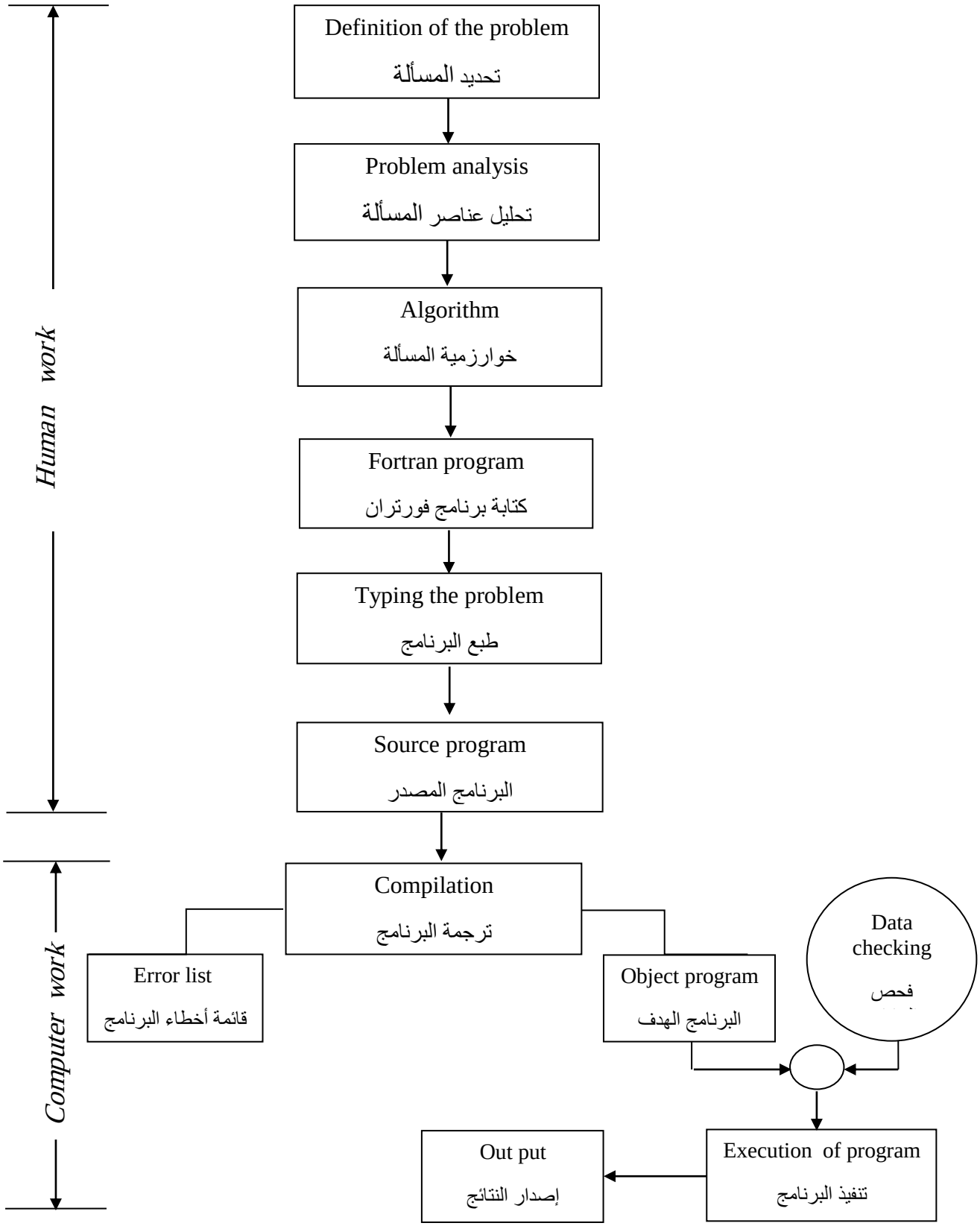
2-High level language

منها لغة فورتران وتكون سهلة التعلم والاستعمال وقابلة للتطور

How to solve a problem using FORTRAN language

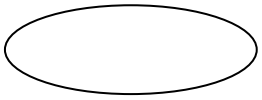
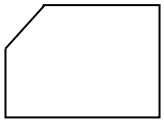
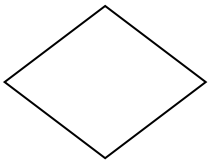
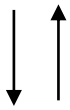
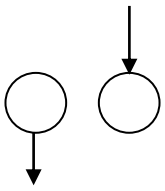
كيف يتم حل المسألة باستخدام لغة فورتران ؟

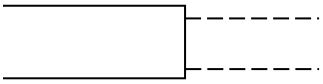
الشكل التالي يوضح ذلك :-



المخطط الانسيابي ***Flow chart***

لكتابة أي برنامج تكون مسبوقة بعملية رسم مخطط انسيابي يمثل تصور لخطوات أو مفردات عمل البرنامج وتوضيح المراحل المنطقية له.

بداية أو نهاية البرنامج	Start / stop	
إدخال وإخراج البيانات	Input / Output	
	or Input	
العمليات الحسابية والخزن	Calculation and store	
تقرير	Decision	
تكرار أو دوران	Looping	
استدعاء برنامج فرعي	Call subroutine	
اتجاه سير البرنامج	Flow line	
نقطة توصيل وربط	connecter	



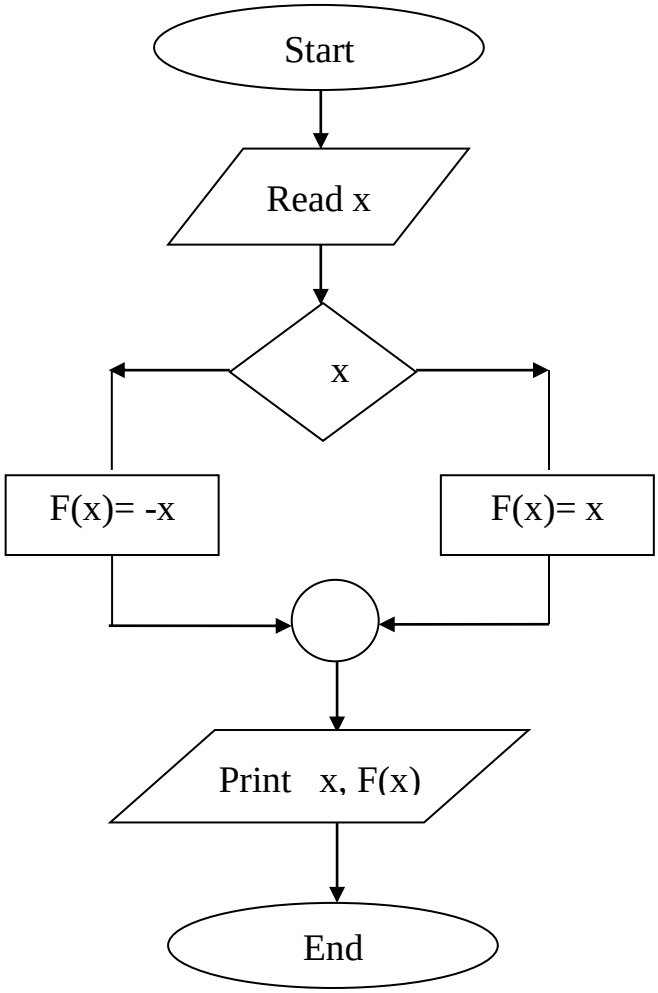
تعليق Comment

نهاية حلقة التكرار End of loop structure

مثال :-

ارسم خريطة سير العمليات لإيجاد قيمة الدالة F(x) المعرفة كما يلي :-

$$F(x)=\begin{cases} x & x\geq 0 \\ -x & x<0 \end{cases}$$



البايت (Byte) :- يعرف البايت الخلية التي تتكون منها وحدة الذاكرة ذات ثمانية أرقام ثنائية (بت)

1 Byte =8 bit

قوانين الكتابة بلغة فورتران — ***Rules For Layout Format***

1- Fortran 77 Fixed_ Format :

- الأعمدة من (1-5) تستخدم لرقم الجملة , أو تترك فارغة عند عدم وجود رقم للجملة ويكون العدد بلا أشارة وبلا فارزة , ويجوز أن يظهر في أي عمود أو مجموعة من الأعمدة المتجاورة.
- العمود رقم (6) يسمى بحقل الاستمرارية وهو يساعد على كتابة العبارات الطويلة والتي تكون أطول من 66 عمود أو حقل, فيوضع أي رمز عدا الصفر والفراغ في حقل رقم 6 , ليدل أن ما مكتوب في هذا السطر هو تكملة للسطر السابق.
- عبارة فورتران الحقيقية يمكن كتابتها من الأعمدة (7-72), وليس من الضروري البدء بالعمود السابع بل يجوز البدء بأي عمود من هذه الأعمدة.
- محتويات الحقول من (72-80) تهمل من قبل الحاسبة وتستعمل لكتابة بعض الدلالات.
- الملاحظات والعبارات التوضيحية في البرامج يجب أن تكون مسبوقة بحرف (c) أو (*) في الحقل الأول من السطر.

2- Fortran 90 Free_ Format Layout :

- الأسماء الرمزية (symbolic_ names) يمكن أن يصل طولها إلى 31 رمز وتكتب بالحروف الصغيرة أو الكبيرة , ويجوز أن تبدأ الكتابة من الحقل الأول. بينما في For 77 يصل طول الرمز 7 أو8 حروف.
- يمكن استخدام الفارزة المنقوطة (semi-colon) لفصل عبارتين او اكثر

```
Sumx=0.0 ; Sumy=0.0 ; Sumz=0.0
```

- العبارات يمكن أن تظهر في أي مكان من السطر ويمكن أن يصل طول السطر إلى 132 رمز أو أكثر.
- التعليقات تتم بوضع علامة التعجب **(!)** (exclamation mark) قبل السطر.
- أذا كانت الجمل طويلة أو لإكمال العبارات توضع علامة **(&)** (Ampersand) في نهاية كل سطر غير مكتمل.

```
Print*, ' line one '&
      , ' line two '&
      , ' line three '
```

❖ تستخدم الفراغات عادة لسهولة القراءة ولكن هنالك فراغات غير مسموح بها مثلا

```
Int ger      or  < =
```

بينما يجوز كتابة التعبير التالي باستخدام الفراغات A*B or A * B

لا يجوز Realx , يجب وضع الفراغات مثل Real x

What is the program?

Example:-

Program p1

A=10.5;B=7.3

Print, 'A+B=', A+B*

End (or End Program p1)

* ملاحظة: في جملة End يجوز كتابة اسم البرنامج أو حذفه.

It consist of

1- orders : program, print, end أوامر و ايعازات

2- characters : A,B حروف أو رموز

3- special characters : =, *, ' رموز خاصة

4- Digits : 10.5 , 7.3 أرقام

أذن البرنامج : هو مجموعة من الايعازات أو الأوامر لتنفيذ مهمة معينة والحصول على نتيجة.

لو نفذ البرنامج p1 لحصلنا على الناتج التالي A+B=17.8

* ملاحظة : يجب تسلسل الايعازات تسلسلا منطقيا , فلا يجوز وضع جملة print في البداية قبل تعريف قيمتي A, B

جملة *program* :

جملة program اختيارية , يصح البرنامج بدونها مع أنها لا تأثير لها على البرنامج سوى أعطاء اسم يستفيد منه المبرمج في المستقبل. ويتكون اسم البرنامج من أي كلمة تحتوي حروفا أو أرقاما ورموز خاصة ويجب أن تبدأ هذه الكلمة بحرف ولا يتعدى طولها 31 رمزا مثلا.

My_program

Ali12

Flow

الأسماء التالية غير جائزة

يحتوي فراغا my program , يبدأ برقم 11_ method

أنواع المعطيات *The intrinsic data types*

1 - الثوابت Constants

الثوابت : هي الكميات التي لا تتغير قيمتها خلال تنفيذ البرنامج, وتقسم إلى 5 أقسام :

❖ **الثابت الصحيح Integer constant**

هو العدد الذي يتكون من رقم أو مجموعة من الأرقام ويخلو من الفارزة العشرية.

-6 , 25, 0

❖ **الثابت الحقيقي Real constant**

هو العدد الذي يتكون من كسر عشري ويأخذ احد الشكلين في لغة فورتران , الشكل الأول هو الذي يحتوي الكسر العشري المعتاد مثل

1.001 , 100.0 , 0.5 , 3.14

الشكل الأخر هو الشكل الآسي (exponential) . وهذا الشكل يستخدم لتمثيل الأرقام الصغيرة أو الكبيرة جدا مثل $0.1*10^{-13}$ ويكتب بهذا الشكل 0.1E-13.

Or $503.8D2=503.8*10^{2}=50380$

يستخدم الحرف D للدلالة على الدقة المضاعفة.

❖ **الثابت المركب Complex constant**

يتكون الثابت المركب من جزئين , ويكتب على النحو التالي

A-B.i

حيث ان :-

A : يمثل الجزء الحقيقي real part ,

B.i : يمثل الجزء ألتربيعي لـ (-1) او الجزء الخيالي imaginary part

ولكتابة الثابت المركب في لغة فورتران تستخدم الأقواس لذلك

(7.5, 2.2)=7.5+2.2i

(180E-3, - 4E-1)=0.180-0.4i

❖ **الثابت المنطقي Logical constant**

هو الثابت الذي يكون على إحدى حالتين الخطأ أو الصواب , أو الحقيقي true أو الزائف false وتكتب قيمتا الثابت المنطقي كما يلي :

.True. .False.

النقطتان المحيطتان بالقيمة ضروريتان , يحجز الثابت المنطقي بايتا واحدا في الذاكرة.

❖ **الثابت الحرفي (الرمزي) character constant**

لا يحوي الثابت الحرفي على قيمة عددية بل يحوي حرفا أو مجموعة من الحروف, تستخدم علامة المتن (الحاصرات العلوية) لإحاطة حروف الثابت , مثل :

' Ali ', "computer center "

2 – المتغيرات Variables

يعرف المتغير بأنه اسم يرتبط بقيمة قد تبقى ثابتة أو تتغير أثناء تنفيذ البرنامج. فان المتغير في فورتران يجب أن يبدأ بأحد الحروف (A-Z) أو (a-z) وقد يتكون من حرف أو أكثر من حرف بحيث لا يزيد عدد الحروف أو الأرقام لاسم المتغير على 7 أو 8 في لغة Fortan 77 بينما في Fortran 90 يمكن أن يحتوي المتغير على 31 رمز أو حرف.

والمتغيرات لها نفس أنواع الثوابت وهذه الأنواع هي :

❖ **المتغير الصحيح Integer variable**

هو المتغير الذي لا يقبل إلا قيمة صحيحة , ويكون صحيحا إذا ابتدأ اسمه بأحد الحروف (I,J,K,L,M,N) مثل : Moon , KK20 . وعند استعمال أسماء أخرى لا تبدأ بأحد الحروف من (I-N) فعندئذ نحتاج إلى جملة نوعية من نوع Integer , مثل :

Integer x,y,z

Integer :: sum

❖ **المتغير الحقيقي Real variable**

هي المتغيرات التي تبدأ بأي حرف من الحروف الصغيرة أو الكبيرة عدا الحروف من (I-N), مثل :

Small zed x1

وعند الحاجة لزيادة قيمة المتغير أو تغيير نوعيته , تستخدم الجملة النوعية Real, مثل :

Real K , M

Real :: K

Real*8 M1 , M2

وللحصول على الدقة المضاعفة تستخدم العبارة التالية :

Double precision N

N : متغير ذا دقة مضاعفة ويحجز 8 بايتات في الذاكرة.

❖ المتغير المركب Complex variable

يحوي المتغير المركب ثابتا مركبا يتكون من جزئين , الجزء الحقيقي و الجزء الخيالي. ويعلن عن المتغير المركب باستعمال الجملة Complex , ويستفاد من هذه الجملة أيضا في تحديد عدد البايتات المحجوزة لكل متغير مركب, مثل :

Complex V , I

Complex *16 X,Y

Complex :: S,N

* **ملاحظة** : المتغير المركب لا يعرف في البرنامج إلا بعد الإعلان عنه في الجملة النوعية Complex وعند عدم الإعلان فان المتغيرات تأخذ الصفة الضمنية التي حددتها لغة فورتران حسب الحرف الأول من اسم المتغير.

❖ المتغير المنطقي Logical variable

المتغير المنطقي هو المتغير الذي لا يحجز إلا بايتا واحدا لان القيمة التي ستخزن في هذا البايت أما أن تكون حقيقية .True. أو زائفة .False.

تستخدم الجملة النوعية Logical لتعريف هذا المتغير , مثل :

Logical I , A

Logical :: I , A

❖ **المتغير الحرفي Character variable**

يختلف هذا المتغير عن المتغيرات السابقة لكونه :

- يتعامل مع الحروف بدل الأعداد.
- لا يوجد عدد محدد للبايتات يحجزها المتغير الحرفي, إذ يعتمد الحجز على عدد الحروف التي يتكون منها المتغير الحرفي.

* تستخدم الجملة النوعية Character لتعريف المتغيرات الحرفية وتحديد حجمها, مثل :

```
Character  A, B
Character *6  V*3 , I , N*20
```

يحجز V : 3 بايتات , يحجز I :6 بايتات , يحجز N : 20 بايتات

- جملة Character لها القدرة على أعطاء حجز مختلف لكل متغير حرفي على حده , وهي ميزة لا توجد مع الجمل النوعية السابقة.

كذلك يمكن الإعلان عن المتغيرات الحرفية باستخدام هذا النموذج.

```
Character (len=11) :: name
```

أي أن المتغير name يمكنه أن يأخذ أي قيمة نصية مكونة من 11رمز مثل "Ali-Mohammad".

* **ملاحظة**: بإمكان المبرمج أن يحدد قيما ابتدائية للمتغيرات عند الإعلان عنها مثل :

```
Integer :: Sum=0 , p=1
```

فوائد عبارات النوع ***Type statements***

يستفاد من عبارات النوع :

1- للإعلان أو التصريح Declaration.

2- لحجز الموقع.

3- التأكيد على النوع.

4- قلب النوع.

5- استقصاء الأخطاء.

ملاحظات:-

- قسمة عدد صحيح على عدد صحيح , الناتج يكون عدد صحيح.

10/3 → 3

4/5 → 0

10/3*3 → 9

- قسمة عدد صحيح على عدد حقيقي أو بالعكس , الناتج يكون عدد حقيقي.

10/3.0 → 3.3333

4./5 → 0.8

2**(-2) → 0

- أذا كان المتغير صحيح , فان قسمة العدد الحقيقي على العدد الصحيح يكون الناتج صحيح.

N=10./3=3

X=10/3=3.0

Y=10/3.0=3.333

جمل الإحلال *Assignment statements*

Variable = expression

تتكون جمل الإحلال من طرفين تفصل بينهما علامة المساواة.

الطرف الأيمن: يتكون من تعبير (ثابت أو متغير أو مجموعة من الثوابت أو المتغيرات تفصل العمليات الحسابية فيما بينهما).

الطرف الأيسر: يكون متغير والذي يمكن أن يكون من أي نوع حسب تعريفه.

تبدأ الحاسبة بحساب الطرف الأيمن وتحول النتيجة النهائية للطرف الأيسر.

A=5

N=(I-L)**K-3

أولويات العمليات ***Operations Precedence***

أولويات العمليات أو ما يسمى بتسلسل العمليات الحسابية . لإيجاد قيمة أي تعبير حسابي في لغة فورتران, ينبغي أن نراعي تسلسل تنفيذ العمليات الحسابية, وفقا لما يسمى بقاعدة الأسبقية. ويكون التسلسل على النحو التالي:-

1- الدوال المكتبية.

2- الأقواس.

3- عملية الرفع (**) أيا كان موقعها.

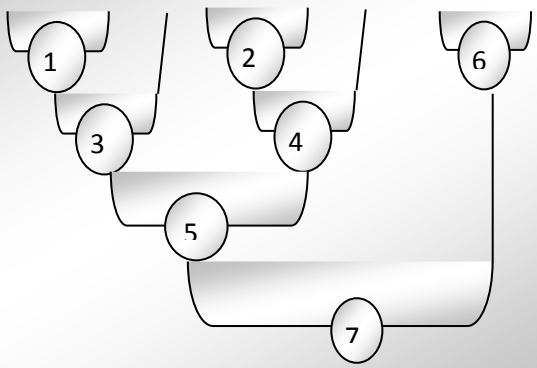
4- الضرب (*) أو القسمة (/) من اليسار إلى اليمين.

5- الإضافة (+) وعملية الطرح (-)من اليسار إلى اليمين.

مثال :- اكتب التعبير التالي بلغة فورتران وبين تسلسل العمليات

$$x= [(a+b)^2+(3c)^8]^{a/b}$$

X=((A+B)**2+(3*C)**8)**(A/B)



ملاحظات:-

- عند تداخل الأقواس , تبدأ عملية الحساب بمحتويات الأقواس الداخلية أولا وبالتسلسل نفسه.
- لا يجوز استعمال علامتي عمليتين حسابيتين متتاليتين من دون أن يفصل بينهما متغير أو ثابت, باستثناء عملية الرفع (**) التي تعد عملية واحدة.

P*-Q 9 ☒

P*(-Q) ✓

- يجب وضع علامة العملية الحسابية بين متغيرين أو ثابتين أو ثابت ومتغير متتاليين

AB ☒
A*B ✓
6A ☒
6*A ✓

التعبير الحرفي ***Character expression***

يتعامل التعبير الحرفي مع البيانات الحرفية, والعملية الوحيدة التي تتعامل مع المتغيرات والثوابت الحرفية هي عملية الربط (الوصل) (concatenation) ورمزه (//).

تكون عملية الوصل نتيجة لوصل نصين ينتج عنهما نص ثالث, مثل:

' No '//' smoking ' →' No smoking'

" For "//" tran " → Fortran

تمارين :-

1- جد ناتج العبارات التالية مبينا خطوات التنفيذ :-

- 4**(1/2)
- ((4+1)**1/2.0)**2

2- ما ذا يطبع البرنامج التالي ؟

```
Character x*4,y*2
Y='no' ; 'flow'
Print*,y//x
End
```

- ثم ارفع علامة الاقتباس من no, flow , ثم نفذ البرنامج.
- ضع علامة الاقتباس على x,y في جملة الطباعة , ثم نفذ البرنامج.

مثال:- ما قيم المتغيرات A,B,C في المقطع البرمجي التالي :-

```
Integer A,B,C
A=10
C=20
B=A
A=C
```

Solution

A=10		A=10		A=20		A=20
B=?	$\xrightarrow{B=A}$	B=10	$\xrightarrow{A=C}$	B=10	$\xrightarrow{C=B}$	B=10
C=20		C=20		C=20		C=10

الدوال المكتبية ***Library Function***

يوجد في لغة فورتران مكتبة من الدوال الرياضية إذ يمكن للمبرمج استخدامها في البرامج إذ تكون جاهزة عند الطلب , وتسمى أيضا بالدوال الضمنية intrinsic functions , انظر الجدول التالي :-

كتابة الدوال في لغة فورتران	معنى الدوال رياضيا
Sin(x) Cos(x) Tan(x) Cot(x)	Sin(x) Cos(x) Tan(x) Cot(x)
Asin(x) Acos(x) Atan(x)	$Sin^{-1}x$=arcsine $Cos^{-1}x$=arccosine $Tan^{-1}x$=arctan
Sinh(x) Cosh(x) Tanh(x)	Sinh(x)=hyperbolic sine of x Cosh(x) Tanh(x)
Exp(x) Alog10(x) Alog(x)=log(x)	e^x log(x) ln(x)
Abx(x) Sqrt(x)	$\lvert x \rvert$ $\sqrt{x}$

Mod(k,l)=modulo(k,l)	Remainder(k,l)

* **ملاحظة**:- ان الدليل الوسيط للدوال المثلثية يجب أن يكون بالزاوية نصف قطرية (radian)

X=sin(90.0*3.1415/180) or X=sind(90.0)

جمل الإدخال والإخراج *Simple Input and Output*

أن عملية نقل المعلومات من والى الحاسبة هي نوع من أنواع ما يعرف بنقل البيانات Data transfer ومن ابسط أشكالها Read*, Print*,

▶ *Input*

Read*,list

List : a list of variables separated by commas

Read*, A,B,C

ملاحظات:-

1- السطر الواحد من المدخلات أو المخرجات يعرف بالسجل record.

2- كل عبارة read تحتاج إلى سجل إدخال جديد, مثلا :

```
Read *,A,B,C
input→3 4 5
```

whereas the statements:

```
read*,A
read*,B
read*,C
input→3
       4
       5
```

require 3 input record

3– إذا كان عدد البيانات في السطر الواحد اقل مما هو مطوب قرأته في جملة read, فان البرنامج ينتقل إلى السطور التالية لتغطية بقية المتغيرات

```
Read*,A
Read*,B,C
Read*,D
input→ 1 2 3
       4
       7 8
       9 10
A=1    B=4    C=7    D=9
```

4- تستعمل النجمة (*) بعد الإيعاز read للدلالة على القراءة المباشرة التي لا تحتاج فيها إلى تعريف حجم القيمة المقروءة وموقعها.

5- يجب مراعاة تسلسل المتغيرات في جملة القراءة ومطابقته مع تسلسل البيانات وأي اختلاف يؤدي إلى توقف البرنامج أو نتائج خاطئة.

```
Read*, number, name, m1, m2, m3
input→ 10, 'Mohammad Ali', 75, 90, 80
```

لاحظ المتغير الثاني هو متغير حرفي ويوضع بين علامتي الاقتباس.

6- إذا كان عدد البيانات في السطر الواحد أكثر مما هو مطلوب قراءته في جملة read, فان البرنامج سيأخذ كفايته من البيانات و يهمل الباقي.

```
Read*, A
input→ 5, 6,7
 A=5
```

7– يجوز استخدام الفراغات للفصل بين قيم الثوابت في البيانات بدلا من استخدام الفوارز.

▶ *Output*

Print*, list

List: constant, variable, expressions, and character strings, separated by commas.

جملة print هي جملة لإخراج النتائج من داخل الحاسبة إلى الوسط الخارجي.

ملاحظات:-

1- كل عبارة print تولد سجل إخراج جديد.

2- لمنع انقطاع سطر الطباعة تستخدم العلامة (&) في الطرفين او في الطرف العلوي.

```
Print*,' This program sol &
    & ve all equation  in the problem'
```

3- عند استخدام علامتي الاقتباس في جملة الطباعة لعبارة حرفية أو ثابت حرفي يظهر كما هو

Print*, ' Ali ' → Ali

Print*, " 1+2 " → 1+2

4- استخدام النجمة (*) التي تلي الإيعاز print تعني إن الطباعة تتم مباشرة من دون صيغة معينة وإنما

تتولى الحاسبة ترتيب النتائج بنفسها.

مثال :- اكتب برنامجا يقرا اسم الطالب ورقمه ودرجاته وإيجاد مجموع ومعدل الدرجات :

```
Integer *2  sum, m1,m2,m3,Number
Character (len=12) Name
Name=' Mohammaed Ali'
Number 10
M1=75;M2=90;M3=80
Sum=M1+M2+M3
Avg=float(sum)/3.0
Print*,'Name:',name,'--------No.:',number
Print*,"marks:",M1,M2,M3
Print*,'sum=',sum,'Average=',Avg
End
```

ملاحظة:-

- تكتب جمل النوع في بداية البرنامج.
- تستخدم float لتحويل المتغير الصحيح إلى حقيقي.

تمـــرين

(1)

A=2, B=3 ,C=5 real

I=2, J=3 integer

Evaluate:

- A**B**I
- J/(I/J)
- A/I/J

(2) جد الأخطاء في هذا البرنامج ثم صحح الخطأ بإعادة كتابة البرنامج .

```
Program   Error-1
Real: A,B,X
X:=6.67
B=.X\Y
Print*,'The answer is",B
End
```

الملفات الخارجية *Text files*

(1) Reading Data from Text Files

يحدث دائما أننا نحتاج أن نختبر برنامج بقراءة مجموعة من البيانات , نفترض أننا نحتاج إلى كتابة برنامج لإيجاد معدل 10 أرقام وبالتالي سيكون صعب جدا أن نعيد كتابة الأرقام مع كل تنفيذ (حيث من

الصعوبة أن يعمل البرنامج من المرة الأولى) لذلك نستخدم ملف خارجي (text file) لخزن البيانات على شكل نص (text) مثلا Data .

صيغة كتابة الملف the formula of the text file

ملاحظة:-

- جملة open و read كلاهما يرتبطان بنفس الرقم (1).
- تستخدم جملة open في بداية البرنامج بعد جمل النوع.
- يمكن أن تستخدم الأرقام من (1-99).
- يقوم المبرمج بتوليد ملف القراءة.

(2) Sending Output to the Printer

صيغة كتابة الملف the formula of the text file

Ex.

```
Open(2,File='D1')
Write (2,*)'This is on the printer'
Print*,'This is on the screen'
```

ملاحظة:-

- لاحظ جملة Write لها نفس الرقم في جملة Open وهو (2).
- تستخدم كلمة Write للكتابة في ملف خارجي , وكلمة print للكتابة على الشاشة.
- تقوم الحاسبة بتوليد ملف الطباعة.

العوامل العلاقية ***Relational Operators***

تعرف العوامل العلاقية بأنها العوامل التي تستخدم لبيان العلاقة بين قيمتين. وتسمى أحيانا بعمليات المقارنة.

Old form	New form	Name
.LT.	<	Less than
.LE.	<=	Less than or equal
.EQ.	==	Equal
.NE.	/=	Not equal
.GT.	>	Greater than
.GE.	>=	Greater than or equal

هذه العوامل الستة يمكن استخدامها بين التعابير الحسابية وكذلك بين المتغيرات الحرفية أيضا.

ملاحظة:-

النقطتان المحيطتان بكل رمز من رموز العوامل العلاقية ضروريتان وتعدان جزءا من الرمز نفسه.

العوامل المنطقية *Logical Operators*

هذا النوع من المعطيات أو العوامل يأخذ إحدى قيمتين True أو False ومن هذه العوامل:-

.Not. ***.And.*** ***.OR.***

.AND.

T .and. T ⟶ T

T .and. F ⟶ F

F .and. T ⟶ F

F .and. F ⟶ F

.OR.

T .OR. T ⟶ T

T .OR. F ⟶ T

F .OR. T ⟶ T

F .OR. F ⟶ F

.NOT.

.NOT. T ⟶ F

.NOT. F ⟶ T

ملاحظة:-

- تكون الأولوية في حالة وجود أكثر من عامل منطقي بالشكل التالي :

- تطبق القاعدة من اليسار الى اليمين في حالة تشابه العوامل .

مثال:- جد الناتج النهائي للتعبير المنطقي التالي :

G*(P/K).GT.3.OR..NOT.A.GT.B

OR G*(P/K) > 3.OR..NOT.A > B

G=5 , P=3 , K=4 Integer variables

A=15.5 , B=20.5 Real variables

Solution

5(3/4) > 3.OR..Not.15.5 > 20.5*

*5*0 > 3.OR..Not.15.5 > 20.5*

0 >3.OR..Not.15.5 > 20.5

F .OR..Not. F

F.OR.T = T