Chapter One
Introduction To programming In FORTRAN 90

مقدمة إلى البرمجة بلغة FORTRAN

وهي اختصار عبارة **FOR**mula **TRAN**slation ، وتعني ترجمة الصيغ أو المعادلات حيث تعد لغة **FORTRAN** من أقدم لغات البرمجة ذات المستوى العالي شيوعاً واستعمالاً لمختلف الأغراض العلمية والهندسية ، وقد ظهرت هذه اللغة في منتصف الخمسينات، حيث بدأ عام **1954** من قبل فريق من الباحثين برئاسة المهندس جون باكس (**JOHN BACKUS**) بالعمل من قبل شركة **IBM** الأمريكية على تطوير لغة برمجة تقبل برنامجاً مكتوباً بلغة قريبة من لغة الإنسان، ويتم تحويله إلى شفرة قابلة للتنفيذ على الحاسب الآلي، وبعد **3** سنوات أي في عام **1957** ظهر أول مفسر فورتران. وفي تلك الفترة كان استخدام الحاسب الآلي يكاد يكون حكراً على العلماء والمهندسين والرياضيين، ومن الطبيعي أن تكون هذه اللغة المطورة حديثاً قد جاءت لتواكب احتياجاتهم إذ تتميز لغة **FORTRAN** بقدراتها على إجراء العمليات الحسابية المعقدة وحل المعادلات الرياضية.

وقد شهدت هذه اللغة العديد من التطورات والإضافات التي وسعت إمكانياتها وجعلتها من أهم اللغات التي تستخدم في مجال الرياضيات . الإصدار الأول **FORTRAN I** صدر في شهر أبريل من عام **1957**م وفي العام التالي **1958**م صدر الإصدار الثاني **FORTRAN II** الذي وفر إمكانية ترجمة الروتينات الفرعية بشكل مستقل. وفي عام **1966** صدرت **FORTRAN 66** من قبل **ANSI** (**America National Standards Institute**) التي كانت الأكثر استخداماً في الأوساط العلمية لفترة طويلة. وفي عام**1978** أصبح الإصدار **FORTRAN 77** من قبل **ANSI** هو اللغة القياسية وتضمن جملة **IF- Else** ومعاملة السلاسل الحرفية .

والإصدار **FORTRAN 90** ظهر عام **1991** وتضمن المؤشرات والمصفوفات الديناميكية. وهناك إصدارات أخرى مستمرة للغة فورتران هي (**FORTRAN 95** ، **FORTRAN 2003** ، ..) ولغة فورتران كأي لغة أخرى تتألف من مجموعة من الكلمات والعبارات والتركيبات المؤلفة طبقاً لقواعد معينة ينبغي التقيد بها دوماً .

مما يحد من انتشار هذه اللغة أنها تحتاج إلى مترجم خاص بها **FORTRAN Compiler** ومن أشهر تلك المترجمات **Power Station, Microsoft Fortran Compiler , Lahey Fortran** و...

ولكتابة وتشغيل برنامج **Fortran** : تحدد طبيعة المسألة وتهيئ طرق حساباتها وتنظم خطوات الحل المطلوبة بدقة ويفضل تهيئة مخطط أنسيابي (**Flow Chart**) يصف بدقة طريقة حل المسألة ، ويحول المخطط الانسيابي إلى برنامج مكتوب بلغة فورتران ويكون هذا البرنامج هو البرنامج المصدر (**Source Program**) ، وتقوم الحاسبة بترجمة البرنامج المصدر ويتم ذلك بواسطة برنامج خاص يعرف بمترجم فورتران (**Fortran Compiler**) ، ويطبع بعدها

البرنامج وتوضح الأخطاء أن وجدت فيه ، ويعاد البرنامج دون أن يشغل في حالة وجود أخطاء فيه ليتم تصحيحها من قبل المبرمج . وأن أفضل وسيلة لتعلم أية لغة في البرمجة هو أن تكتب برنامجاً بسيطاً يحتوي جملة أخراج ثم تراقب كيف يعمل هذا البرنامج ، أن هذه الخطوة من شأنها التأكد من صحة عمل المحرر **Editor** ومن ثم المترجم **Compiler** وكذلك الرابط **Linker** وهذه كما ذكرنا مراحل أعداد وتنفيذ البرنامج ٠ وكذلك فإن كتابة برنامج بسيط تؤدي إلى الارتقاء بهذا البرنامج إلى برامج أكثر تعقيداً وبالتدريج ٠

العناصر الأساسية في لغة FORTRAN 90

1- الأحرف والرموز المستخدمة في لغة FORTRAN 90 : أن الأحرف والرموز المسموح استخدامها تشمل ما يأتي :

١ - الحروف الأبجدية الإنكليزية: تتكون من الحروف الكبيرة والصغيرة وهي ستة وعشرون حرفاً هي :

A , B , C , D , ... , Z وتسمح فورتران 90 بأستعمال الحروف الأبجدية الصغيرة مثل **a , b , c ,...z** .

٢ - الأرقام الحسابية عربية الأصل ، وهي : **0 , 1 , 2 , 3 , 4 , 5 , 6 ,..... , 9**

٣ - الرموز الخاصة وتشمل ما يأتي :

Name	Character	Name	Character
Colon	:	Blank (space)	
Exclamation mark	!	Equal sign	=
Quotation mark	"	Plus sign	+
Percent sign	%	Minus sign	-
Ampersand	&	Asterisk	*
Semicolon	;	Slash	/
Less than	<	Left parenthesis	(
Greater than	>	Right parenthesis	)
Question mark	?	Comma	,
Dollar sign	$	Period (decimal point)	.
underscore	_	Apostrophe	'

2- الثوابت Constants :

الثوابت هي الكميات التي تبقى ثابتة أثناء تنفيذ البرنامج وتنقسم إلى قسمين :

الثوابت العددية Numerical Constants : وهي عبارة عن أعداد رقمية وكل عدد يمكن أن يحوي علـــى فاصلة عشرية وكما يمكن أن يحمل الإشارة السالبة أو الموجبة وهناك ثلاث أنواع من الثوابت العددية :

1- الثوابت الصحيحة Fixed Point Constant or Integer Constants

تعُرف الثوابت الصحيحة بأنها أعداد صحيحة أو صفرية مسبقة أو غير مسبقة بإشارة وتظهر هذه الأعداد بدون الفاصلة العشرية ((.)) ، أمثلة للثوابت الصحيحة (.... , 314- , 300+ , 5 , 0)

يتراوح الحد الأدنى والحد الأقصى لهذا النوع من الثوابت العددية حسب نوع وطراز الحاسب المستعمل.

قواعد الثوابت العددية الصحيحة Rules for Integer Number:

١- لا يجوز استخدام الفاصلة العشرية ((.)) في كتابتها.

٢- لا يجوز استخدام الفارزة ((,)) في كتابتها.

٣- يمكن وضع إشارة ((+)) أمام العدد للدلالة على أنه موجب وكذلك الحال بالنسبة للإشارة ((-)) عندما يكون العدد سالباً وعند عدم وجود الإشارة يعتبر العدد موجباً .

2- الثوابت الحقيقية Real Constants or Floating Point Constants

وتشمل هذه المجموعة من الثوابت الأعداد الحقيقية الكسرية وتكتب بأستخدام الفاصلة العشرية والصورة العامة لها في لغة فورتران تكون بالشكل التالي :

n.m	n.	n.mE±s	n.E±s	.mE±s	nE±s

حيث : n : تمثل العدد الصحيح

m : تمثل الجزء الكسري أي الأرقام الموجودة على يمين الفاصلة .

E : تمثل علامة الضرب ((×)) والأساس ١٠ .

s : تمثل الأس وهو عدد صحيح مكون من رقمين فقط.

أمثلة على الثوابت الحقيقية (.... , 5.00 , +5. , 14.1- , 13.25)

ويتغير الحد الأدنى والأقصى لهذه الثوابت حسب نوع وطراز الحاسب المستخدم.

قواعد الثوابت العددية الحقيقية Rules for Real Number :

١- العدد الحقيقي الكسري يحتوي على فاصلة عشرية .

٢- الفارزة غير مسموح بها في أي عدد .

٣- يمكن إضافة أي عدد من الاصفار في مقدمة العدد أو مؤخرته دون التأثير في قيمته.

٤- يمكن للعدد أن تكون له إشارة تسبقه .

ملاحظـــــــــــة : يمكن تمثيل الثوابت العددية بواسطة طريقة التدوين اليائي (E - Notation) وبخاصة أذا كانت هذه الثوابت كبيرة جداً أو صغيرة جداً ، فمثلاً الثابت العددي الحقيقي الكسري 0.00135 يمكن تمثيله بـــ0.135E-2 والتي تعني أن العدد الموجود إلى يسار E مضروب في ١٠^-٢ أو العدد الموجود إلى يمين E يمثل القوة (الأس) التي يرفع اليها العدد ١٠ ،

$$\pm YE\pm n = \pm Y \times 10^{\pm n}$$

والشكل العام الذي يمثل هذا النوع من التدوين هو :

حيث Y عدد حقيقي أو صحيح في لغة فورتران و n عدد صحيح بالإضافة إلى الإشارة ،

وهنالك أشكال أخرى للمعطيات الحقيقية في لغة Fortran الغرض منها أن تحدد مجالاً أو دقة للأرقام مثل 0.3_DOUBLE الذي يعطي الثابت 0.3 دقة مضاعفة أو الشكل 0.1234567_QUAD الذي يعطي الثابت المذكور أربعة أمثال الدقة المعهودة ،

3- الثوابت المركبة Complex Constants

الثابت المركب : هو ثابت عددي ذو فاصلة عشرية وهو مكون من جزئين يسمى أحدهما الجزء الحقيقي (Real Part) ويسمى الآخر بالجزء الخيالي (Imaginary Part) ويكتب عادة بين قوسين ويفصل بين جزئيه هذين بفارزة،

فيما يأتي بعض الأمثلة على هذا النوع من الثوابت :

صورة الأعداد المركبة الرياضية	صورة الأعداد بالفورتران
3.2-1.86i	(3.2,-1.86)
4.0+5.0i	(4.0,5.0)
-21.+3.24i	(-21.,3.24)

الثوابت غير العددية : وأنواع هذه الثوابت هي:

1- الثوابت المنطقية Logical Constants : هذا النوع من الثوابت يختلف عن الأنواع سابقة الذكر في أنه ليست له قيمة عددية وإنما يكون على أحد حالتين إما أن تكون TRUE صواب أو FALSE خطأ، وحسابياً TRUE تأخذ قيمة الرقم ١ أو أي رقم آخر عدا صفر و False تأخذ الرقم صفر ،

2- الثوابت الرمزية String Constants : وهذه الثوابت تتكون من سلسلة أو عدد من الرموز وتستخدم عادة في كتابة العناوين أو تمييز بعض نتائج البرامج وتكتب بين حاصرتين علويتين (من النوع المفرد أو المزدوج) وهذا النوع من الثوابت غير عددي أي لا يخضع للعمليات الحسابية ومثاله : "Amount" , 'Quality' , "10+20" , 'Fortran 90 solved all of Julie"s problems' , " program's logic " , '2008' ...

ملاحظـــــــــــة : هنالك تعبير واحد يتعامل مع هذا النوع من الثوابت وهو تعبير الوصل (Concatenation) ورمزه (//) ، وتكون عملية الوصل نتيجة لوصل نصين ينتج عنهما نص ثالث مثل عملية الوصل :

"NO"//"SMOKING" تكون نتيجتها "NO SMOKING" ،

3- المتغيـــــــرات Variables

المتغير هو كمية تعطي أسماً معيناً مثل X أو Y ويسمح لها بالتغير أي بأخذ قيم مختلفة خلال تنفيذ البرنامج ٠

وشروط كتابة أسم المتغير في لغة FORTRAN 90 هي :

١ - يتكون من رمز أو مجموعة رموز وهذه الرموز هي أما حروف أو أرقام والرمز (_) UNDER SQURE فقط

٢ - يسمى المتغير بحيث لا يتجاوز عدد رموزه عن 31 رمزاً (حروف وأرقام والرمز _ فقط) ٠

٣ - يجب أن يبدأ أسم المتغير بحرف ولا يجوز أن يبدأ برقم أو (_) ٠

٤ - يستحسن استعمال أسم للمتغير يدل على نوع استخدامه ويفضل تجنب الحرف الواحد ٠

وهناك نوعان من المتغيرات في لغة FORTRAN 90 هما المتغيرات العددية والمتغيرات غير العددية ٠

المتغيرات العددية Numerical Variables

وهي المتغيرات التي تستخدم لتخزين الثوابت العددية في وحدة الذاكرة وهي على أنواع أهمها :

1 - المتغيرات الصحيحة Integer Variables

هي المتغيرات التي يسمح لها بأخذ قيمة ثابت صحيح فقط وتخضع أسماء المتغيرات الصحيحة لقاعدة الحرف الأول أذا لم تعرف في بداية البرنامج في عبارة نوع وهي أن كل متغير يبدأ بأحد الأحرف الستة الآتية :

I , J , K , L , M , N فهو متغير صحيح

فمثلاً الأسماء التالية : MASS_1 , K5 , I , JM , KK , NUM_2 هي متغيرات صحيحة ٠

2- المتغيرات الحقيقية Real Variables

وهي المتغيرات التي تستخدم لخزن الأعداد الحقيقية المحتوية على كسور عشرية في الذاكرة، ويخضع أسم المتغير الحقيقي لقاعدة الحرف الأول أذا لم يعرف في بداية البرنامج في عبارة نوع وهي أن كل متغير يبدأ بأي حرف عدا : الحروف الستة الآتية : I , J , K , L , M , N فهو متغير حقيقي

فمثلاً الأسماء التالية : Value_1 , X5 , Y , ABC , Calculate_of_X هي متغيرات حقيقية ٠

3- المتغيرات المركبة Complex Variables

وهي المتغيرات التي تستخدم لخزن الثوابت المركبة (تم تعريفها سابقاً) في الذاكرة ، ويجب تعريف هذه المتغيرات في بداية البرنامج بعبارة Complex ٠

المتغيرات غير العددية : وأنواع هذه المتغيرات هي:

1- المتغيرات المنطقية Logical Variables :

وهي المتغيرات التي تستخدم لخزن الثوابت المنطقية (تم تعريفها سابقاً) في الذاكرة ، ويجب تعريف هذه المـــتغيرات في بداية البرنامج بعبارة Logical ٠

2- المتغيرات الرمزية CHARACTER TYPE OR STRING VARIABLES

هذا النوع من المتغيرات ليس له قيم عددية ، كالمتغيرات العددية ، وإنما له قيم رمزية ويتكون المتغير الرمزي من مجموعة من الرموز Strings والتي يمكن أن تكون من الحروف والأرقام والرموز الخاصة التي سبقت الإشارة اليها ٠ويستخدم هذا النوع من المتغيرات كعناوين أو رموز توضيحية يحتاج أليها المبرمج لتبيان أسماء المتغيرات أو معانيها أو للتعبير عن قيم ثوابت رمزية ٠

ويتم استخدام عبارة النوع **CHARACTER** في بداية البرنامج للتعبير عن المتغيرات الرمزية ٠

أمثلة على هذا النوع من المتغيرات **CHARACTER (Len=25) :: NAME**

حيث : **NAME** : أسم المتغير الرمزي

(Len=25) : طول المتغير الرمزي أي عدد رموزه فيحجز له في الذاكرة 25 موقع ٠

ويمكن كتابة العبارة السابقة بالشكل التالي: **CHARACTER (25) :: NAME**

لأن كلمة **Len** هي اختيارية **Optional** ٠

وقد تكتب عبارة نوع المتغير الرمزي بالشكل التالي : **CHARACTER (*) :: ANSWER**

وهذا يعني استخدام الصيغة الحرة (*) لمقدار الحجز لقيمة المتغير الرمزي **ANSWER**

(أي مهما كانت قيمة الثابت الرمزي المراد الحجز له) ٠

ملاحظـــــــــــة: إذا لم يذكر أي رقم بعد عبارة **CHARACTER** فأنه يفهم من ذلك أن طول المتغير الرمزي المعلن عنه لا يتعدى رمزاً واحداً فقط ٠ والمثال التالي يوضح هذه الملاحظة

```
CHARACTER :: RLGION
RLGION = 'I'
```

حيث **I** : هو رمز واحد فقط يمثل القيمة الرمزية للمتغير الرمزي **RLGION** والحرف **I** هنا هو اختصار للثابت الرمزي **ISLAM** وقد أستعمل الحرف **I** وحده لأن عبارة **CHARACTER**لم تحجز غير مكان واحد فقط لقيمة المتغير الرمزي٠

وقد تكتب عبارة **CHARACTER** كما في الإصدار السابق **FORTRAN 77** بالشكل التالي :

مثال : **CHARACTER *10,X,Y**

وتعني أن المتغيرين **X,Y** متغيرات رمزية وتم الحجز لقيم كل منهما بمقدار **10** ٠

مثال آخر : **CHARACTER *6,L*4,M,N,S*8**

ومعناه : المتغيرات الرمزية **M , N** يحجز لهما **6** رموز والمتغير الرمزي **L** يحجز له **4** رموز والمتغير الرمزي **S** يحجز له **8** رموز ٠

قواعد استعمال المتغيرات الرمزية Rules for string variables

عند استعمال المتغيرات الرمزية فأنه ينصح بأتباع القواعد التالية :

١ - يجب الإعلان عن أي متغير رمزي في البرنامج وذلك باستخدام عبارة CHARACTER في بداية البرنامج وقبل أسم المتغير الرمزي.

٢ - لا تستخدم الحاصرات العلوية مع المتغيرات الرمزية وإنما مع الثوابت الرمزية .

٣ - ينبغي أن يكون الرقم المستعمل بعد عبارة CHARACTER صحيح العدد.

٤ - يجوز استعمال المتغيرات المؤشرة والمصفوفات في جملة الحجز CHARACTER

مثل : CHARACTER (2):: MONTH(12)

٥ - لا يجوز أن يستعمل أسم متغير رمزي في أكثر من جملة CHARACTER واحدة .

أمثلة :

```
CHARACTER (LEN=8) :: DateINFO
CHARACTER (LEN=4) :: Year, Month*2, Day*2
CHARACTER (LEN=2) :: Hour, Minute, Second*6
```

ملاحظـــــــــــة: لم يكن من الضروري الإعلان عن المتغيرات في إصدارات لغة FORTRAN الأولى (FORTRAN 4 , FORTRAN 77) وكانت أنواع المتغيرات تستنبط من الحرف الأول لأسم المتغير . فالمتغيرات التي كانت تبدأ بالأحرف (I , J , K , L , M , N) كانت تعتبر من النوع الصحيح (INTEGER) تلقائياً ، وعدا ذلك كانت المتغيرات تعتبر حقيقية ولكن في لغة FORTRAN 90 ولمجاراة اللغات الأحدث، يستحسن أن يعلن المبرمج دائماً أسماء وأنواع المتغيرات لما ينطوي ذلك عن فائدة في استقصاء الأخطاء وسهولة قراءة البرنامج .

عبـــارات النوع TYPE STATEMENTS

هي عبارات التصريح (declaration) بنوع المتغيرات التي تستخدم في البرنامج وتوضع هذه العبارات عادة في بداية البرنامج وهي عبارات اختيارية Optional ولكنها مهمة جداً كما تم ذكره في لغة FORTRAN 90 لأنه في كثير من الأحوال يحتاج المبرمج إلى استخدام بعض أسماء المتغيرات التي لا تخضع لقاعدة الحرف الأول السابقة وفي هذه الحالة نستطيع استخدام عبارات تحديد النوع للتغلب على المشكلة، وكذلك لتجنب الوقوع في خطأ استخدام متغيرات من نفس النوع في العمليات الحسابية وأيضاً سهولة قراءة البرنامج وهناك خمسة أنواع من عبارات النوع المستخدمة في لغة FORTRAN 90 هي:

INTEGER, REAL, LOGICAL, COMPLEX and CHARACTER

والصيغة العامة لكتابة عبارات النوع هي : **Type , specifier :: list**

حيث :

Type : عبارة النوع المراد تعريف المتغيرات بها (INTEGER, REAL, LOGICAL, ...)

Specifier : وهي بعض العبارات التي تحدد المتغير إضافة إلى عبارة النوع وسنأتي على ذكرها ..

list : هي أسماء المتغيرات المراد تعريفها مفصولة عن بعضها بفارزة (,) .

أمثلة على عبارات النوع (Specification statements)

1) **INTEGER :: A2, B_1, BETA, QUEEN** حيث تحدد هذه العبارة

أن كل من **A2,B_1,BETA,QUEEN** هي متغيرات صحيحة (دون التقيد بقاعدة الحرف الأول)

2) **REAL :: I_10,JAK2,MAX,MIN** حيث تحدد هذه العبارة

أن كل من **I_10,JAK2,MAX,MIN** هي متغيرات حقيقية (دون التقيد بقاعدة الحرف الأول)

3) **LOGICAL :: FLAG,MAR** حيث تحدد هذه العبارة

أن كل من **FLAG,MAR** هي متغيرات منطقية

4) **COMPLEX :: Total, A_1** حيث تحدد هذه العبارة

أن كل من **Total,A_1** هي متغيرات مركبة

- هناك نوع آخر من عبارات النوع هي عبارات النوع الضمنية مثل :

IMPLICIT INTEGER :: (P-T)
IMPLICIT REAL :: (J-N)

حيث تعني العبارة الأولى أن أي متغير يبدأ بالحروف من P إلى T هو متغير صحيح ، مثل **PETA , SQ , RN ...,**

والعبارة الثانية تعني أن أي متغير يبدأ بالحروف من J إلى N هو متغير حقيقي ، مثل **KAT , NOON , L1 , ...**

عبارة **IMPLICIT NONE** : هي عبارة مهمة جداً في لغة **FORTRAN 90** ويكون موقعها في بداية البرنامج بعد أسم البرنامج وقبل عبارات التصريح بالنوع وتعني إلغاء قاعدة الحرف الأول في تعريف نوع المتغيرات وأن كل المتغيرات يجب أن يعرف نوعها بأحد عبارات النوع التي تم ذكرها سابقاً ورغم أنها عبارة اختيارية ألا أنه يجب استخدامها في كل برامج لغة **FORTRAN 90** لأنها تساعد أن يكون البرنامج أمين من الأخطاء .

هيكلية برنامج FORTRAN 90 : F90 Program Structure

يتكون برنامج لغة FORTRAN 90 من البرنامج الرئيسي ووحدات أخرى من البرامج ومنها البرامج الفرعية :

ويكون الشكل العام للبرنامج الرئيسي هو:

```
PROGRAM program-name
IMPLICIT NONE
Specification statements
...
Executable statements
...
END PROGRAM program-name
```

عبارة PROGRAM

وهي عبارة اختيارية ، يصح البرنامج بدونها ولكن من الجيد التعود على كتابتها في كل برنامج مع أنها لا تـــأثير لهـــا على البرنامج سوى أعطاء أسم يستفيد منه مستخدم البرنامج في المستقبل .

وأسم البرنامج (***program-name***) هو أسم يعطى إلى البرنامج ويفضل أن يكون أسم البرنامج يدل على عمل البرنامج وهذا الأسم تنطبق عليه شروط كتابة أسم المتغير أي هو كلمة أو مجموعة كلمات متكونة من (حروف وأرقام والرمز _) بشرط أن لا يتعدى طوله (31) رمزاً وأن لا يبدأ برقم .

وأمثلة على أسم البرنامج هي: My_Program , Test , Calculate_Value_Of_Z , ...

Executable statements : تعني عبارات تنفيذية مثل عبارات الإدخال والإخراج وغيرها من العبارات التنفيذية الأخرى وسوف نأتي على ذكرها لاحقاً .

عبارة END

وهي عبارة غير تنفيذية ينتهي بها كل برنامج والتي تشير الى نهاية نص البرنامج وفي لغة FORTRAN 90 يفضل أن تكتب عبارة End ويكتب بعدها نفس العبارة الأولى أي (عبارة PROGRAM) كما موضح بالمثال التالي:

```
PROGRAM Calculate
Print *, 10 + 20
END PROGRAM Calculate
```

ملاحظة : لا يجوز أن يحتوي البرنامج على أكثر من جملة END

عبارة الإيضاح والتعليق Comments

وهي عبارة غير تنفيذية تستخدم لتوضيح وشرح بعض خطوات البرنامج أو وظيفة البرنامج وبعض المعلومات عنه ، ولذلك فمن الممكن أن يوجد أكثر من عبارة تعليق واحدة في بداية البرنامج وفي ثناياه ..

وصيغة هذه العبارة هي ظهور رمز علامة التعجب (!) Exclamation mark في بداية السطر أو بعد نهاية عبارة معينة وكل شيء يكتب بعد الرمز (!) يعتبر غير تنفيذي وهو توضيح أو شرح أو تعليق ما .

مثال :

```
PROGRAM Who
!  This program reads a name
!  Then prints it
READ *, NAME    ! Input Statement
PRINT *, NAME   ! Output Statement
END PROGRAM Who
```

عبارة تحديد الثوابت PARAMETERS

أذا كان لدينا أسماء لثوابت معينة وأردنا تحديد هذه الثوابت خلال البرنامج فيتم ذلك من خلال المحدد Parameter في أحد عبارات النوع كما ذكر سابقاً ويكتب بالصيغة العامة التالية :

```
TYPE , PARAMETER :: CONS_1 = 100 , CONS_2 = 200 , ...
```

أمثلة :

```
REAL,  PARAMETER :: pi = 3.1415926 , e =2.17828
INTEGER, PARAMETER  :: MAXIMUM = 100
LOGICAL, PARAMETER :: TRUE = .true. , FALSE = .false.
CHARACTER(LEN=3), PARAMETER :: YES  = "yes" , NO= "no"
```

الغرض من استخدام أسماء للثوابت هو زيادة الوضوح ، وكذلك أعطاء فرصة للكمبيوتر من أي خطأ غير مقصود عند استعمال المتغيرات . وعل سبيل المثال لاحظ البرنامج الآتي :

```
PROGRAM METER_TO_INCHES
!  This program Convert Meter To Inches
IMPLICIT NONE
REAL :: METERS
REAL, PARAMETER :: INCHES_PER_METER=39.37
READ *, METERS
PRINT *, METERS * INCHES_PER_METER
END PROGRAM METER_TO_INCHES
```

ملاحظة : يجب الانتباه إلى أن أسم الثابت لا يمكن تغيير قيمته خلال البرنامج فلا يمكن أن تظهر العبارة التالية في البرنامج مثلاً :

INCHES_PER_METER=39.3701

ولو ظهرت هذه الجملة فإن البرنامج سيظهر خطأ ينبه المبرمج لذلك .

القواعد الشكلية للبرامج المكتوبة بلغة FORTRAN

لقد تغيرت القواعد الشكلية التي كانت تستخدم في الإصدارات السابقة للغة FORTRAN ولأسباب تاريخية كانت لغة FORTRAN تكتب في بطاقات ، فعليه أن صيغة عبارات FORTRAN يجب أن تكون متوافقة مع شكل هذه البطاقات وكانت كل بطاقة مقسمة إلى أربعة أقسام متميزة هي:

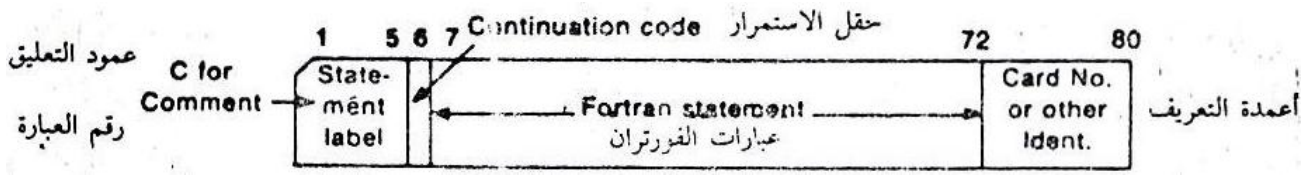
١ - الأعمدة من (١) إلى (٥) تعرف بأعمدة ترقيم العبارة (Statement Label) .

٢ - العمود السادس يعرف بعمود الاستمرارية (Continuation Code) وهو الذي يستعمل للدلالة على استمرارية العبارة أي أن ما موجود في البطاقة هو تكملة لما هو موجود في البطاقة السابقة لها .

٣ - الأعمدة من (٧) إلى (٧٢) تستعمل لكتابة عبارة فورتران وتعرف بمجال العبارة (Statement Field)

٤ - الأعمدة من (٧٣) إلى (٨٠) تستعمل لترقيم البطاقات الأختياري وتعرف بمجال ترقيم البطاقة أو حقل التعريف (Identification Field) وللمبرمج وضع أي حروف أو أرقام أو كلاهما معاً لتشير الى البرنامج والسطر حيث يفيد هذا الحقل في الرجوع لأي سطر في البرنامج دون الرجوع الى محتوياته ، وكذلك يمكن ترتيب البرنامج إذا حدث أي خلل أو تغيير في هذا الترتيب .

أن كافة عبارات فورتران تخضع لهذا التصنيف عدا عبارات التعليق والشرح (Comment Statement) والتي تبدأ عادة بطبع حرف (C) في العمود الأول . كما في الشكل التالي :



أما في الأصدار FORTRAN 90 فقد تحررت هذه القيود وأصبح شكل البرنامج كما موضح ويمكن أن نذكر أن القواعد الشكلية السابقة لا تزال مقبولة في FORTRAN 90 غير أنه لا يجوز الخلط بين القواعد الشكلية السابقة والحالية في نفس الملف .

```
PROGRAM program-name
IMPLICIT NONE
Specification statements
...
Executable statements
...
END PROGRAM program-name
```

قواعد السطر بلغة FORTRAN 90: F90 LINE RULES

البرنامج الواحد يتكون من مجموعة من العبارات وكل عبارة تبدأ بسطر منفرد والسطر الواحد في لغة FORTRAN 90 يحتوي على (132) رمزاً . لذا لا يمكن كتابة سطر أكثر من (132) رمزاً ولذلك يجب أن تكتب العبارة بأكثر من سطر بإضافة الرمز & (Ampersand) إلى نهاية السطر للإشارة إلى استمرارية السطر كما في المثال التالي :

ويمكن كتابة أسطر استمرار إلى حد 39 سطر .

```
PRINT *, "LINE ONE" , &
         "LINE ONE" &
       , "LINE ONE"
```

كما أنه يمكن كتابة أكثر من جملة في لغة FORTRAN في سطر واحد وذلك بفصل الجملتين عن بعضهما بواسطة الفاصلة المنقوطة (;) مثل : SUM = O ; PRODUCT = 1

ملاحظـــــة : بإمكان المبرمج أن يحدد قيماً ابتدائية للمتغيرات عند الإعلان عنها بعبارات النوع مثل :

```
INTEGER :: SUM = O , PRODUCT = 1
REAL :: Offset = 0.1, Length = 10.0 , Tolerance  = 1.E-7
```

العمليات الحسابية ARITHMETIC OPERATIONS

توجد خمس عمليات حسابية أساسية ، وهذه العمليات مع الرموز المقابلة لها في لغة FORTRAN هي:

الجمع : + مثال B + A

الطرح : - مثال A – B

الضرب : × مثال A * B تعني A.B

القسمة : / مثال A/B تعني A÷B

الرفع إلى الأس : ** مثال A**B تعني A^B

التعبير الحسابي ARITHMETIC EXPRESSION

كان الغرض الأساسي للبرمجة بلغة FORTRAN هو أجراء العمليات الحسابية، ثم تطورت فيما بعد لتشمل أجراء عمليات على رموز متنوعة . فالتعبير الحسابي هو أجراء العمليات الحسابية على بعض الثوابت والمتغيرات والدوال وذلك لإيجاد قيمة عددية، والتعبير الحسابي في صورته البسيطة يتكون من ثابت واحد أو متغير واحد أو دالة واحدة ومثال على ذلك: T-R3 , (X+Y)**3 , A/B*(X+12.5)

وفي لغة FORTRAN 90 يمكن ربط المتغيرات والثوابت من النوع الرمزي CHARACTER بواسطة وسيلة الوصل (//). ومثال ذلك : X // Y // "abcde" .

قواعد كتابة وحساب قيمة التعبير الحسابي :

1- يجب أن لا يظهر رمزان من رموز العمليات الحسابية بجانب بعضها البعض مباشرةً فمثلاً من الخطأ أن يترجم التعبير الجبري الى فورتران بالتعبير الحسابي A/-B حيث ظهر الرمزان - ، / بجانب بعضهما دون أي فاصل بينهما ولكن يمكن أن يترجم الى A/(-B) حيث أستخدمنا القوس للفصل أو أن يترجم الى A/B- وهذا يعطي نفس القيمة المطلوبة .

2- قاعدة الأسبقية (RULE OF PRCEDENCE): ويتم تنفيذ التعبير الحسابي على أساس قاعدة الأسبقية الآتية :

١ - تنفيذ ما بداخل الأقواس ( ) ٠

٢ - الدوال إن وجدت ٠

٣ - تنفيذ عمليات الرفع للأس **٠

٤ - تنفيذ عمليات الضرب * أو القسمة / وحسب أسبقيتها من اليسار الى اليمين٠

٥ - تنفيذ عمليات الجمع + أو الطرح – وحسب أسبقيتها من اليسار الى اليمين٠

ملاحظـــــــــات مهمة :

١ - في حالة وجود عدة أقواس داخلية وخارجية ((( ))) فتعطي الأسبقية لما هو داخل الأقواس الداخلية أولاً ثم الخارجية التي تليها ثم التي تليها وهكذا ٠٠٠

٢ - يجوز أن يكون الأس (القوة) تعبيراً وليس مجرد ثابت أو متغير وكذلك يجوز أن تكون الكمية التي ترفع إلى هذا الأس تعبيراً٠

٣ - لا يسمح برفع قيمة سالبة إلى قوة حقيقية ولا برفع الصفر إلى قوة صفرية٠

٤ - عند أجراء عملية حسابية من نفس النوع ينتج ذلك النوع ٠

مثال ذلك **3./2. = 1.5 REAL**

3/2 = 1 INTEGER

٥ - عند أجراء عملية حسابية مختلفة يحول الصحيح إلى حقيقي (كسر) وينتج حقيقي٠

مثال ذلك: **3./2 = 1.5**

3/2. = 1.5

مثال : حول كل علاقة رياضية مما يلي إلى عبارة واحدة مقابلة لها بلغة فورتران ؟

1) $A= \frac{4}{3}p\ R^3$

2) $K= (\frac{\lambda}{\mu})^n$

3) $X = \frac{-B+\sqrt{B^2-4AC}}{2A}$

4) $Y = (ax^4 + 10^{-5} + a\ \cos^2 x)^{3/4}$

الحل :

```
1)  A= (4./3.)*3.141597*R**3
2)  K= (LMBDA/MU)**N
3)  X= (-B+SQRT(B**2  - 4.*A*C)) / (2*A)
4)  Y= (A*X**4+1.0E-5+ALFA*COS(X)**2)**(3./4.)
```

عمليات المقارنة RELATIONAL OPERATORS

يمكن أجراء عمليات مقارنة بين القيم العددية وتكون نتيجة هذه التعابير منطقية (TRUE صواب أو FALSE خطأ)

الجدول التالي يمثل تعابير المقارنة المستخدمة في لغة FORTRAN:

الرمز الرياضي	المعنى	الرمز بلغة Fortran 77	الرمز بلغة Fortran 90	مثال
=	يساوي	.EQ.	==	C = B
1	لا يساوي	.NE.	/=	C /= R
<	أصغر من	.LT.	<	I < J
≤	أصغر من أو يساوي	.LE.	< =	I < = K
>	أكبر من	.GT.	>	Z > B
3	أكبر من أو يساوي	.GE.	> =	Y > = X

العوامل المنطقية :

تستعمل العوامل المنطقية للربط بين التعابير المنطقية ومن أهمها :

١ - .AND. : ومعناها (و) وتكون صواب إذا المقارنة على الجانبين صواب ٠

٢ - .OR ٠ : ومعناها (أو) وتكون صواب إذا المقارنة على الجانبين صواب أو أحدهما صواب ٠

٣ - .NOT. : ومعناها (لا) وتكون صواب في حالة المقارنة على اليمين خطأ ٠

٤ - .EQV. : ومعناها (مكافئ) وتكون صواب إذا المقارنة على الجانبين صواب أو المقارنة على الجانبين خطأ

٥ - .NEQV٠ : ومعناها (لا يكافئ) وتكون صواب إذا المقارنة على الجانبين أحدهما صواب والأخرى خطأ

ملاحـــــــــظات :

١ - العامل (.NOT.) يجب أن لا يسبق بأي عبارة أو متغير أما العاملين (.AND.) و (.OR.) فيجب أن يكونا مسبوقين ومتبوعين بتعبير أو متغيرات ٠

٢ - قاعدة الأسبقية للعوامل المنطقية تعطي أولوية التنفيذ لما هو داخل الأقواس الداخلية أولاً ثم الخارجية التي تليها ثم التي تليها وهكذا ٠٠ ويعطي العامل (.NOT.) الأولوية بين العوامل المنطقية ويليه العامل (.AND.) ويليه العامل(.OR.) ثم (.EQV٠) أو (.NEQV٠)

مثال: أفرض X = 30.0 و Y = 40.0 و A = 10 جد قيمة التعبير المنطقي التالي:

الحل:

```
IF (X+2.0 < Y .OR. Y /= 40.0 .AND. A > -3.1)
    T          F          T
               F
               T
```

نبدأ بالعامل AND حسب قاعدة الأسبقية ثم العامل OR ٠

الدوال المكتبية LIBRARY FUNCTIONS

لكل حاسب الكتروني مكتبة للدوال الرياضية وهي على شكل برامج معدة لبعض الصيغ الرياضية ومخزونة بصورة دائمة داخل الحاسب لأستخدامها عند الطلب من قبل المستفيدين ومخططي البرامج على أن تذكر فقط ضمن برامجهم ولها أولوية التنفيذ. وتحوي لغة FORTRAN العديد من الدوال المكتبية المعرفة لدى هذه اللغة ونذكر فيما يلي بعضاً من هذه الدوال :

١- دوال تحويل نوع المتغير Numeric Type Conversion Function :

قد يستخدم بعض المبرمجين الأنواع المختلفة (الصحيح والحقيقي) من المتغيرات في نفس العبارة الحسابية وأن عملية المزج بالأنواع هذه قد تؤدي في بعض الأحيان الى نتيجة غير متوقعة .

وعلى سبيل المثال عند أجراء عملية القسمة التالية : PI/2

حيث PI = 3.1415 كما هي معرفة لذا تكون نتيجة القسمة تساوي (1) والسبب في ظهور مثل هذه النتيجة أن المقسوم عليه من نوع صحيح INTEGER ولذلك فإن الكمبيوتر يقوم بإسقاط الكسر في عملية القسمة أي أن الكمبيوتر يقوم بإتباع قاعدة القسمة وكأنها أعداد صحيحة ،ولإلزام الكمبيوتر بإجراء القسمة على أنها قسمة أعداد حقيقية REAL وهناك طريقتان :

أحداهما أن يكتب التعبير على الشكل الآتي : PI/2.0 بإضافة كسر عشري للمقسوم عليه

أو باستعمال الدالة المكتبية REAL لتغيير النوع الى الحقيقي كما يلي : PI/REAL(2)

أما أذا كان المقسوم عليه متغيراً من نوع صحيح INTEGER فإن الطريقة الثانية هي الوحيدة ، فإذا أردنا أن نقسم المتغير PI على المتغير I وكان المتغير I صحيحاً فإننا نكتب التعبير الحسابي كما يلي : PI/REAL(I)

وهنالك دوال مكتبية أخرى لتغيير النوع مثل INT التي تقوم بإسقاط الجزء الكسري من قيمة المتغير .

فلو كان عندنا متغير X من نوع REAL وكانت قيمة X هي 1.5 فإن نتيجة الدالة INT(X) ستكون 1 أي بإسقاط الكسر عنه .

هناك عمليات تغيير من نوع آخر تتم تلقائياً عند أجراء المساواة فبعد أجراء العملية التالية : I = X

على فرض أن قيمة X هي 1.5 وأن المتغير I من نوع صحيح فإن قيمة I تكون 1 أي أن قيمة X حولت الى نوع صحيح أولاً ثم أجريت عملية المساواة .

وعلى الرغم من التحويلات التلقائية الجارية عند المساواة فإنه من المستحسن التعود على أجراء عمليات تحويل النوع بشكل ظاهر كما في : I = INT(X)

وذلك لتجنب الأخطاء غير المقصودة التي يصعب تعليلها عند تنفيذ البرنامج .

الجدول التالي يحتوي على بعض دوال التحويل المكتبية الشائعة في لغة FORTRAN 90 :

Function	Meaning	Arg. Type	Return Type
INT(x)	truncate to integer part x	REAL	INTEGER
NINT(x)	round nearest integer to x	REAL	INTEGER
FLOOR(x)	greatest integer less than or equal to x	REAL	INTEGER
FRACTION(x)	the fractional part of x	REAL	REAL
REAL(x)	convert x to REAL	INTEGER	REAL

Examples:

```
INT(-3.5) Æ   -3
NINT(3.5) Æ    4
NINT(-3.4) Æ  -3
FLOOR(3.6) Æ   3
FLOOR(-3.5) Æ  -4
FRACTION(12.3) Æ  0.3
REAL(-10) Æ  -10.0
```

٢ - الدوال الحسابية Mathematical Functions :

تحوي مكتبة لغة FORTRAN 90 العديد من الدوال الحسابية المعروفة والجدول التالي يحتوي على بعض الدوال الشائعة الاستعمال في هذه اللغة .

Return Type	Arg. Type	الصيغة الرياضية	Meaning	Function (الصيغة بالفورتران)
INTEGER REAL	INTEGER REAL	$\|x\|$	القيمة المطلقة	ABS(X)
REAL	REAL	e^x	الدالة الأسية	EXP(X)
REAL	REAL	$\ln x$	اللوغاريتم الطبيعي	ALOG(X) Or LOG(X)
REAL	REAL	$\log x$	اللوغاريتم الأعتيادي	ALOG10(X) Or LOG10(X)
REAL	REAL		الجذر التربيعي	SQRT(X)
REAL	REAL	Sin(x)	جيب الزاوية	SIN(X)
REAL	REAL	Cos(x)	جيب تمام الزاوية	COS(X)
REAL	REAL	Tan(x)	ظل الزاوية	Tan(x)
REAL	REAL	$Tan^{-1}(x)$	الظل المعكوس للزاوية	ATAN(X)
REAL	REAL	Tanh(x)	الظل الزائدي للمقطع	TANH(X)
REAL	REAL	$Sin^{-1}(x)$	جيب الزاوية المعكوس	ASIN(X)
REAL	REAL	$Cos^{-1}(x)$	جيب تمام المعكوس	ACOS(X)

Return Type	Arg. Type	الصيغة الرياضية	Meaning	Function (الصيغة بالفورتران)
REAL	REAL	SINH(X)	جيب زائدي	SINH(X)
REAL	REAL	COSH(X)	جيب تمام زائدي	COSH(X)
INTEGER	INTEGER		maximum of x1, x2, ... xn	MAX(x1,x2, ..., xn)
REAL	REAL			
INTEGER	INTEGER		minimum of x1, x2, ... xn	MIN(x1, x2, ..., xn)
REAL	REAL			

ملاحظـــــــــات:

١ - يجب أن تكون الزوايا في الدوال المكتبية بالزوايا النصف قطرية.

٢ - لكي نتجنب الخلط بين الكميات الكسرية والكميات الصحيحة في التعبير الحسابي نستخدم دالة التحويل REAL

العبارة الحسابية ARITHMETIC STATEMENT

هي عبارة تستخدم لحساب قيمة تعبير حسابي ثم إعطاء هذه القيمة إلى متغير (فإذا كان لهذا المتغير قيمة سابقة فأنها تمحى وتحل محلها هذه القيمة الجديدة أي قيمة التعبير الحسابي) .. والصورة العامة للعبارة الحسابية هي :

تعبير حسابي = أسم متغير

Variable Name = Arithmetic Expression

وعلامة (=) هي ليست المساواة الموجودة في الرياضيات وأنما هي عملية أحلال بالبرمجة أي حل الطرف الأيمن (التعبير الحسابي) محل الطرف الأيسر (المتغير) فمثلا العبارة الحسابية A = B+C تفقد A قيمتها السابقة لتحل محلها قيمة B + C أما قيمة كل من B,C فأنها لا تتغير .

وفي حالة ما إذا أختلف نوع المتغير عن نوع التعبير الحسابي فأن قيمة التعبير الحسابي تحسب أولاً حسب نوعه ثم تعدل هذه القيمة إلى صورة النوع الآخر قبل إعطائها للمتغير. فمثلاً إذا كان المتغير حقيقياً والتعبير صحيحاً فأن كل العمليات الحسابية في التعبير تجري بالأعداد الصحيحة ثم تحول النتيجة وهي عدد صحيح إلى الصورة الحقيقية قبل أعطائها للمتغير الموجود على يسار العبارة الحسابية .. وكذلك العكس فإذا كان المتغير صحيحاً والتعبير حقيقياً فإن قيمة التعبير الحقيقية المحسوبة تحول أولاً إلى قيمة صحيحة بحذف الجزء الكسري قبل إعطاء هذه القيمة للمتغير الصحيح على يسار العبارة.

نؤكد هنا على الصورة العامة للعبارة الحسابية حيث الطرف الأيمن عبارة عن تعبير حسابي أو متغير معرف سابقاً أو ثابت والطرف الأيسر عبارة عن اسم متغير واحد فقط. فمثلاً المعادلة الرياضية X – Y = Z + T

لا يمكن أن تكون عبارة حسابية وذلك لأن الطرف الأيسر فيها ليس متغيراً واحداً وإنما هو تعبير حسابي.

عبارات الإدخال والإخراج Input–Output Statements

طرق أدخال البيانات INPUT DATA METHODS

هناك طريقتان لإدخال البيانات هما :

1- الطريقة المباشرة **Direct Method** : وتتم بإدخال البيانات مباشرة إلى الحاسبة من خلال العبارات الحسابية حيث يكون الطرف الأيمن في العبارة الحسابية ثابتاً . مثل : **A = 2.4**

M = 4

Z = 825.75/40.0

ويستخدم هذا النوع من الإدخال عندما تكون البيانات قليلة ولا يحبذ استخدامها حين الحاجة إلى عدد كبير من البيانات أو في حالة تبديل البيانات بسبب استعمال البرنامج لأغراض أخرى .

2- الإدخال باستخدام عبارة القراءة **Read Statement** :

والصورة العامة لعبارة القراءة هي: **Read (m,n) List**

حيث :**m** : يمثل رقم وحدة الإدخال المستخدمة ويعبر عنه بالرمز (*) ويعتمد على نوع الحاسبة .

n : رقم صحيح يدل على رقم عبارة الصيغة (Format) أو توضع * (نجمة) تعني (Free Format) أي صيغة حرة.

List : مجموعة المتغيرات المراد أدخالها مفصولة عن بعضها بفارزة (,) .

مثال : **Read (*,*) A, B, M**

وتعني أدخال ثلاث متغيرات هما **M, B, A** بأستخدام الصيغة الحرة (*) .

ويمكن كتابة عبارة القراءة بالصورة الآتية : **Read *, A , B , M**

مثال :

```
Read (*,10) A,B,M
10   Format ( 2F10.3, I6 )
```

وتعني أدخال ثلاث متغيرات هما **M, B, A** بأستخدام صيغة **Format** المعبّر عنها في عبارة رقم **10** .

ويمكن كتابة عبارة القراءة بالصورة الآتية : **Read (*, FMT="(2F10.3, I6)") A , B , M**

طرق أخراج النتائج OUTPUT RESULTS METHODS

هناك عبارتان تستخدمان لإخراج النتائج هي:

١ - باستخدام عبارة الكتابة **Write Statement** :

والصورة العامة لعبارة الكتابة هي : **Write (m , n) List**

حيث:

m : يمثل رقم وحدة إخراج النتائج المستخدمة ويعبر عنه بالرمز (*) ويعتمد على نوع الحاسبة .

n : الصيغة (Format) المستخدمة أو توضع *(نجمة) تعني (Free Format) أي صيغة حرة .

List : مجموعة أسماء المتغيرات المراد أخراجها وهي مفصولة عن بعضها بفارزة (,) .

مثال : **Write (*,*) I, C**

وتعني أخراج نتائج قيم المتغيرات **I,C** بأستخدام الصيغة الحرة (*) التي تطبع النتائج بطريقة التدوين اليائي .

مثال : **Write (*, "(I6, F10.3)") I, C**

وتعني أخراج نتائج قيم المتغيرات **I,C** بأستخدام صيغة **Format** .

٢ - باستخدام عبارة الطباعة **Print Statement** :

والصورة العامة لعبارة الطباعة هي : **PRINT * , List**

حيث :

List : مجموعة أسماء المتغيرات المراد أخراجها وهي مفصولة عن بعضها بفارزة (,) .

مثال : **PRINT *, A, B, C**

وتعني أخراج نتائج قيم المتغيرات **A ,B,C** باستخدام الصيغة الحرة

مثال : **PRINT "(3F8.3)", A, B, C**

وتعني أخراج نتائج قيم المتغيرات **A ,B,C** باستخدام صيغة **FORMAT** ٠

ملاحظات مهمة :

١ - في عبارة القراءة **Read Statement** يفضل أستخدام الصيغة الحرة **Free Format** حتى يقوم المستخدم بأدخال البيانات بحرية تامة دون التقييد بأي صيغة **Format** .

ويمكن كتابة عبارة القراءة بالصورة الآتية : **Read *, A , B , M**

٢ - في عبارة الكتابة **Write Statement** يفضل أستخدام رقم عبارة الصيغة **Format** حتى يسهل على المستخدم أخراج النتائج بالشكل المطلوب (تستخدم في نسخ لغة **FORTRAN** القديمة).

٣ - في لغة **FORTRAN 90** يفضل استخدام عبارة **Print** في إخراج النتائج بشكليها (أستخدام صيغة حرة أو صيغة **FORMAT**)٠ مثال : **PRINT *, AREA** (صيغة حرة)

PRINT "(A,F9.3)", "AREA=", AREA (صيغة **FORMAT**)

٤ - يتضح مما سبق أن البرنامج هو عبارة عن إدخال بيانات ومعالجة البيانات وإخراج النتائج .

عبارات الصيغة Format Statements

هناك عدة أنواع من الصيغ هي :

1- صيغة الثوابت العددية الصحيحة I FORMAT

تستعمل هذه الصيغة مع الأعداد الصحيحة (أي تستعمل مع متغيرات الفاصلة الثابتة) سواء في إدخال البيانات أو أخراج النتائج، والصورة العامة لها هي : **nIw**

حيث : **n** : عدد الحقول في سجل الإدخال والأخراج،

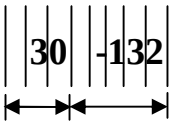
I : رمز صيغة الأعداد الصحيحة،

W : عدد الأعمدة المحجوزة لكل متغير (حقل) متضمن الأشارة،

أمثـــــــلة :

مثال (1) : **PRINT "(I3,I5)", M,N**

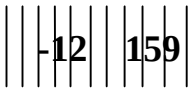
فإذا كانت قيم كل من N,M هي **M = 30** و **N = -132** فسوف يكون الإدخال كالآتي (تثقيب بطاقة البيانات)،



I3 I5

مثال (2) : **PRINT "(2I5)", J,K**

IF J= -12 , K= 159



2 – صيغة الثوابت العددية الحقيقية F FORMAT

تستعمل هذه الصيغة مع الأعداد الحقيقية (الكسرية) (أي تستعمل مع متغيرات الفاصلة السائبة) سواء في إدخال البيانات أو أخراج النتائج، والصورة العامة لها هي : **nFw.d**

حيث: **n** : عدد الحقول في سجل الإدخال أو الأخراج،

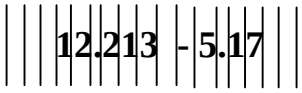
F : رمز صيغة الأعداد الكسرية،

W : عدد الأعمدة الكلي في كل حقل،

d : عدد الأعمدة إلى يمين الفاصلة العشرية،

مثال (3) : **PRINT "(F9.3,F8.4)", X,Y**

IF X=12.213 , Y=-5.17



مثال (4) :

```
PRINT "(F10.5,2F7.4)", A,B,C
```

IF A=296.28 , B=27.18 , C=-3.54

```
| 296.28 | 27.18 |-3.54 |
```

مثال (5) :

```
PRINT "(F10.3)", TOTAL
```

IF TOTAL =2802.4

```
| 2802.4 |
```

3 - صيغة الأعداد الكسرية الأسية E FORMAT

تستعمل هذه الصيغة للأعداد الكسرية الأسية والتي تكون قيمها كبيرة جداً أو صغيرة جداً.

والصورة العامة لها هي: nEw.d

حيث:

n : عدد الحقول في سجل الادخال او الأخراج.

E : رمز صيغة الأعداد الكسرية والأسية وتطبع في الأدخال والاخراج.

w : عدد الأعمدة الكلي في كل حقل.

d : ويمثل عدد الأعمدة المخصصة لأرقام العدد فقط والتي محصورة بين الفاصلة العشرية والرمز E .

بشرط ((w-d ³ 7)) .

وتظهر صيغة E عند طبعها على جهاز الأخراج كما يلي : ±0.← d →E±ee

مثال (6) :

```
PRINT "(E12.3)", AREA
```

IF AREA =918000

```
| 0.918E+06 |
```

مثال (7) :

```
PRINT "( E12.4,E10.3)", A,B
```

IF A=251.832 , B=0.0082468

```
| 0.2518E+03 0.825E-02 |
```

ملاحظة :

إذا كان الجزء الكسري في عبارة الصيغة Ew.d أقل من الجزء الكسري الممثل للعدد المطلوب طبعه فإن الحاسب يقوم بتقريب الكسر إلى أقرب رقم.

مثال(8):

```
PRINT "( E13.6,E14.6)", H,S
```

IF H=0.0000001357 , S=-235.7978

```
|0.135700E-06|-0.235798E+03|
```

ملاحظة :

إذا كان الجزء الكسري في عبارة الصيغة Ew.d أكثر من الجزء الكسري الممثل للعدد المطلوب طبعه فإن الحاسب يقوم بوضع أصفار على يمين العدد.

4 – صيغة A A FORMAT

تستعمل هذه الصيغة لطباعة الثوابت الرمزية والصورة العامة لها هي:

حيث : Axx

A : رمز الصيغة .

xx : عدد المواقع الموجودة في حقل هذه الصيغة وهي اختيارية (OPTIONAL) .

مثال (9) :

```
PRINT "( A4,I3)", Y,IN
```

IF Y="ZAID" , IN=96

```
|ZAID|96|
```

مثال (10) :

```
PRINT "(A)", "FORTRAN 90"
```

تكون نتيجة الطباعة

```
|FORTRAN|90|
```

5- صيغة فسحة أو أهمال الحقول X FORMAT

تستعمل هذه الصيغة لترك فراغات بين الحقول والصورة العامة لها هي : nX

حيث:

n : عدد الأعمدة التي تترك فارغة.

X : صيغة فسحة الحقول.

أمثلة :

مثال (11) :

```
PRINT "( I4,5X,I6)", I,J
```

IF I=115 , J=-25

```
|115|     |-25|
```

مثال (12):

```
PRINT "( F12.6,4X, F6.2)", C,P
```

IF C=927.123456 , P=191.24

```
927.123456    191.24
```

6- صيغة تحديد الحقل T FORMAT T :

تستخدم الصيغة T لإلزام الكمبيوتر بالطباعة في عمود يحدد رقمه والصورة العامة له هو: Tn

حيث:

n : رقم العمود المراد الطباعة فيه.

T : صيغة تحديد الحقل.

مثال (13) :

```
PRINT "(A,T10,I2)", " FORTRAN ", 90
```

تكون نتيجة الطباعة كما يلي:

```
FORTRAN    90
```

7- صيغة القيود المتعددة Multiply Record Format:

تستعمل هذه الصيغة عند إدخال أو أخراج عدد من السطور حيث يستعمل الخط المائل (Slash) ((/)) لإنهاء السطر.

مثال (14):

```
PRINT "(2F7.3/2I5)", C,D,I,J
```

IF C=22.35 , D=91.62 , I=753 , J=8024

```
22.35   91.62
 753  8024
```

ملاحـــظات مهمة :

١- يمكن استعمال عبارة صيغة واحدة تستخدم للقراءة والطباعة مثل:

```
READ (*,10) I,J
WRITE (*,10) I,J
10  FORMAT (2I5)
```

٢- توضع فارزة لفرز صيغة عن صيغة أخرى في عبارة الصيغة.

٣- إذا كان الحجز في صيغة FORMAT أقل من عدد الأعمدة أو المواقع فأن الطبع يظهر**** بقدر الحجز .

أســـــــــــــئلة محلولة :

س1/ حول كل علاقة رياضية مما يلي إلى عبارة واحدة مقابلة لها بلغة فورتران ؟

1) $\varphi = \sqrt{|m-n|}$

2) $X = \frac{\alpha^3 .10^{22}}{\ln\left|cos\frac{\phi}{3}\right|}\left|e^{sin\phi}\right|$

3) $Y = \ln(|sec\, x + \sin x|)$

4) $\theta = \sqrt[3]{\frac{e^{2x}}{\sec^2 x .\csc 2x}}$

الحل:

```
1) FI=SQRT(ABS(REAL(M-N)))
2) X=ALFA**3 *1.0E22*ABS(EXP(SIN(FI))) / LOG(ABS(COS(FI/3.)))
3) Y=LOG(ABS(1.0/COS(X)+SIN(X)))
4) THETA= (EXP(2*X) / ((1./COS(X))**2*1/SIN(2*X)))**(1./3.)
```

س2/ فيما يلي مجموعة من العبارات FORTRAN 90 المطلوب بيان أي من هذه العبارات صحيحة وأيها غير صحيحة وبالنسبة للعبارات غير الصحيحة وضح الخطأ ؟

```
1) B=SQRT(X+3E3)
2) Y=3.2X+A**3/-4.0
3) X-Y=ALOG(T+Q)
4) J=J+1
5) X  Y=12.5
6) XI=JI+K-4
7) GAMA=A=(B+C)**2
8) PI=3.14159
9) 12.5=X-A
10) J=SIN(a+cos(x))/4.0
```

الحل:

١ - عبارة صحيحة.

٢ - عبارة غير صحيحة لا يوجد رمز لعملية حسابية بين 3.2 والمتغير X وكذلك الرمزان / و – متجاوران مباشرة.

٣ - عبارة غير صحيحة – الطرف الأيسر تعبير حسابي وليس متغيراً واحداً.

٤ - عبارة صحيحة .

٥ - عبارة غير صحيحة – لاحتواء الطرف الأيسر على فراغ (b) فصل بين مكونات اسم المتغير.

٦ - عبارة صحيحة. ٧ - عبارة غير صحيحة – لاحتواء العبارة على علامتين يساوي (أحلال).

٨ - عبارة صحيحة . ٩ - عبارة غير صحيحة – الطرف الأيسر ثابت وليس اسم متغير.

١٠ - عبارة صحيحة.

HOME WORK :

س1/ حول كل علاقة رياضية مما يلي إلى عبارة واحدة مقابلة لها بلغة FORTRAN ؟

1) $T=\ln\left|\sin\frac{X}{3}\right| + \frac{1}{e^{2X}+4}$

2) $C=\sqrt{|COS(a-n.b)|} + \ln|m-n|$

3) $H=1+X+X^2/2!+X^3/3! +X^4/4!$

4) $Y=\frac{1}{2} r^2(q-\sin q)+\frac{4}{3} p\, r^3$

5) $\sigma = \sqrt{\frac{y(x-\bar{x})^2}{n} - \frac{yx}{n^2}}$

6) $\beta = \frac{x^{\frac{3}{2}}(1+\tan x^2)}{10^{-5}+\alpha\, x^4}$

س2/ فيما يلي مجموعة من العبارات بلغة فورتران والمطلوب بيان أي من هذه العبارات صحيحة وأيها غير صحيحة وبالنسبة للعبارات غير الصحيحة وضح الخطأ ؟

1) Z2=A*-B+C**4
2) X=SQRT(SQRT(X))
3) I+1=I
4) R=16.9X+AB
5) SQRT(49.0)=7.0
6) V-3.96=X**1.67
7) K12345=I**J
8) W=SQRT(YZ+4E3+12.5)
9) –A=5.0*D-F
10) R2=EXP(SIN(X/2.0)

س3/ ما هي القيمة المتوقعة لكل من SQ,IA,RN,I,X,Y بعد تنفيذ البرنامج التالي:

```
PROGRAM HOME_WORK
IMPLICIT NONE
 REAL :: I , X , Y
INTEGER :: N= 6 ,  J=4 , SQ , RN , IA
SQ= 6.8+N/4.
 IA= 4.2+N/8
 RN= SQ+IA/6.
 I= N/IA
X= N/REAL(J)
 Y=1/3*X-1.2
 PRINT *,SQ,IA,RN,I,X,Y
 END PROGRAM HOME_WORK
```