الدرس الأول : أساسيات في الباسكال

قبل أن نتمكن من كتابة برامج بلغة الباسكال أو أي لغة أخرى لا بد لنا أولا من فهم منطقي لجميع الخطوات الواجب

إتباعها لحل المسألة بواسطة الحاسب
هذه الخطوات هي ما تعرف باللوغاريتم The Algorithm بعد ذلك يمكن تحويل هذه الخطوات إلى إيعازات باسكال
والتي ستتحول بدورها إلى لغة الآلة المستعملة بواسطة برنامج المصنفCompiler

وبرنامج الباسكال هو مجموعه من الإيعازات بعضها منفذ Executable وبعضها غير منفذ-Non Executable
وتتألف هذه الإيعازات من كلمات دليليه أو من كلمات تعريفية يجب على المبرمج تعريفها في بداية كل برنامج

ولتوضيح ذلك نرى مخطط لبرنامج باسكال

```
Program The Total (input, output);

{This program find sum of two integer number}
Var
A, B, Total: integer;
Begin

Write (‘Enter two integer numbers to be add: ‘);

Readln (a, b);
Total: = a+b;

Writeln (‘The sum of’, A,’and’, B,’is’, Total);

End.
```

حيث نلاحظ
-السطر الأول هو للتعريف باسم البرنامج ووسيله الإدخال والإخراج

-أن الجزء الملون بالأزرق هو عباره عن كلمات محجوزه سنتعرف عليها بالتفصيل بالدروس القادمه

-الجزء الملون بالأخضر هو عباره عن إيعاز غير منفذ وهي ملاحظات خاصه بقاريء البرنامج

-الكلمه الدليليه Var للتصريح عن المتغيرات وسيتم دراستها بالتفصيل

-الجزء الممتد من Begin إلى End هو عباره عن جسم البرنامج والأوامر بينها

-السطر الأخير يحتوي على End متبوعه بنقطه وذلك للدلاله على نهايه البرنامج وسنلاحظ أن

-1بعد كل سطر هنالك نقطه فاصلة) للدلالة على سطر آخر يلية

-2بعد كلمه End هنالك نقطة (.) للدلاله على أنه لا يوجد سطر بعد ذلك

الدرس الثاني : أنواع البيانات في الباسكال
في هذا الدرس سوف نتعرف على أنواع البيانات في لغة الباسكال والقواعد الخاصه لكل منها للتمكن من جمع هذه البيانات في تعابير تشكل إيعازات البرامج في هذه اللغة
يمكننا تمييز نوعين من أنواع البيانات :

الأول
وهو ما تشترك فيه كل لغات البرمجه لذلك سنطلق عليه إسم البيانات القياسيه وهي:
١- الكميات الصحيحه Integer
وهي الأعداد الصحيحه الكامله التي لا تحوي على كسور

مثل ١١

٢- الكميات الحقيقيه Real
وهي الأعداد الحقيقيه التي تحوي كسور فيها
مثل ٢.٣٤٣٤

٣- الكميات المميزه Char

وهي الحروف والأشكال المميزة
مثل &،^،A,@,H

٤- الكميات المنطقيه Boolean
ويمكن أن تأخذ قيمتين هما إما صحيح True أو خطأ False

الثاني

وهو البيانات التي يصرح بها عن طريق المبرمج وذلك بتعريف أنواع جديده من البيانات ليست من الأنواع الأربعة التي تكلمنا عنها

الدرس الثالث : المتغيرات في برنامج الباسكال
التصريح Var

ويتم في هذا القسم من البرنامج التصريح عن نوع البيانات بالبرنامج قبل إستخدامها
ويتم التصريح حسب نوع البيانات إما الأنواع الأربعه Integer , Real , Boolean , Chr أو نوع قام المستخدم بتعريفه وسيتم دراسته بفصل آخر
ويخضع إسم المتغير إلى شروط وهي
١- أن يبدأ بحرف (ولا تقبل الأسماء التي تبدأ بأرقام)

٢- يحوي الإسم حروفا وأرقاما فقط

٣- لا يكون الإسم من ضمن الكلمات المحجوزه بالذاكرة

٤- يفضل دائما أن يكون إسم له معنى للتسهيل

أمـــثـلة

A: integer؛ تم تعريف مكان بالذاكره إسمه A وهو يقبل البيانات فقط من نوع الأعداد الصحيحه

B: Real؛ تم تعريف مكان بالذاكره إسمه B وهو يقبل البيانات من النوع الصحيح وأيضا النوع الذي الحقيقي الذي يحوي كسور

Grade: Chr؛ تم تعريف مكان بالذاكره إسمه Grade وهو يقبل البيانات من النوع Chr

F: Boolean؛ تم تعريف مكان بالذاكره إسمه F وهو يقبل بيانات من النوع Boolean أي قيم منطقيه (يقبل إما قيمه True – False)

والآن لنأخذ مثال تطبيقي على إستخدام التصريح Var

A:=5(صحيح) ؛

A:=3.5صحيح على متغير إلى كسري حقيقي عدد إسناد تم لإنه – (خطأ) ؛

B:=5(صحيح) ؛

B:=3.5(صحيح) ؛

Grade:=`r(صحيح) ؛`

Grade:=d الأقواس وضع لعدم – (خطأ) ؛

F:=True(صحيح) ؛

F:=False(صحيح) ؛

F:=10 المتغير لإن – (خطأ)؛ F إما منطقيه قيم إلا يقبل لا True أو False

ونلاحظ مما سبق
القيمه Real اعم من Integer فهي تقبل القيم الصحيحه والحقيقيه ولكنها تأخذ حيز من الذاكرة أكبر

الدرس الرابع : الثوابت في البرنامج

لقد تعرفنا على طريقة تعريف متغيرات بالبرنامج لحفظ البيانات فيها حسب نوعها وذلك داخل التعريف Var

ولكن ماذا لو أردنا تعريف بيانات تكون قيمتها ثابته طوال عمل البرنامج فإننا نستخدم لتعريفها

نوع جديد داخل الأمر Const وتكون على الشكل التالي

```
Const

A = 30;
B = 9.6;
C = "Name:"
```

- طوال عمل البرنامج سيسند للثابت A القيمة ٣٠ ولا يمكن تغيير هذه القيمة خلال البرنامج أبدا
- نلاحظ أن هناك علامة مساواة = بين إسم الثابت وقيمته بينما في تعريف Var المتغيرات هناك علامة : بين إسم المتغير ونوعه

- الثابت المعرف يعامل في البرنامج على أنه كلمة محجوزة Reserved Word

```
Program Circle(input,output:(
Const

Pi=3.14;
Var

Radius : integer;
Circum , area : real;
Begin
Writeln ( 'Enter Radius:('
Read ( radius:(
Circum := 2*pi* radius;
Area := pi * radius * radius;
Writeln ('The value of circum is',circum:(
Writeln ('The value of area is',area:(

End.
```

في البرنامج السابق قمنا بتعريف ثابت في البرنامج بالإسم pi وهو يتخذ دائما القيمة ٣.١٤

إستخدام الثوابت في البرنامج يسهل عمل البرنامج ويسهل تعديل القيم بسهولة بخاصة في البرامج الكبيرة

لو أردنا تنفيذ البرنامج السابق ولكن بدون الإعتماد على الثوابت فسيكون البرنامج على الشكل

```
Program Circle(input,output:(

Var

Pi : real:
Radius : integer:
Circum , area : real:
Begin
Write ('Enter Value of pi:(':
Readln (pi:(
Writeln ( 'Enter Radius:('
Read ( radius:(
Circum := 2*pi* radius:
Area := pi * radius * radius:
Writeln ('The value of circum is',circum:(
Writeln ('The value of area is',area:(
End:
```

الدرس الخامس : أمر Write والأمرWriteln

سنتعرف في هذا القسم على نوع جديد من الأوامر في الباسكال وهذا النوع من الأنواع الشائعة الإستخدام في البرنامج وهو الأمر Writeln ويستخدم هذا الأمر للطباعة وإخراج البيانات على الشاشة وله نوعان

الأول : Writeln

وهو يقوم بطباعة الجملة والإنتقال بعدها إلى السطر الذي يليه مثل

```
Writeln ('This is First Line:('
Writeln ('This is the second Line:('
Writeln ('This is the first line','This is complete of the first line:('
```

نلاحظ في هذا المثال أنه سيقوم بطباعة الجملة الأولى وفي نفس السطر سيقوم بطباعة الجملة الثانية وسينتقل بعدها للسطر الذي بعده

الثاني : Write

وهو مشابه للأمر Writeln ولكن الفرق الوحيد أنه لا ينتقل سطر للأسفل بعد طباعة الجملة بل يكمل بنفس السطر

```
Write ('This is the first line:('
Writeln ('This is the complete of the first line:('
Write ('This is the Second line:('
Write ('This is complete of line 2:('
Writeln ('This is the End:('
```

في هذا المثال توضيح كامل لعمل دالة الإخراج بحالتيها

يقرأ البرنامج السطر الأول ويطبعه وبما أنه Write فإن المؤشر لن ينتق للسطر الذي يليه بل سيقرأ السطر الآخر ويطبعه بنفس السطر

وبما أن السطر الثاني من النوع Writeln فإن المؤشر بعد الإنتهاء من طباعة السطر ينتقل للسطر التالي لطباعة البيانات الأخرى
لذلك سيكون ناتج السطور السابقة بعد التنفيذ

This is the first line This is the complete of the first line
This is the Second line This is complete of line 2 This is the End

بعد الإنتهاء من نتفيذ الأوامر ينتقل المؤشر I إلى السطر الذي يلي آخر جمله لإنها من النوع Writeln

تدريب :
لو أراد منك طباعة الأسطر التالية

My Name is Rayan
o
I live in Riyadh

حيث يقصد دائما بالعلامة o فراغ
الجواب

Writeln (‘My Name is Rayan:(‘
Writeln:
Writeln (‘I Live in Riyadh:(’

ويمكن أيضا حلها بالشكل التالي

Writeln (‘My Name is Rayan:(‘
Writeln:(’)
Write (‘I Live in Riyadh:(’

والفرق بينهما أن المؤشر في الحالة الأولى سيكون في النهاية في السطر الذي يلي آخر عبارة
وفي المثال الثاني سيكون في نفس السطر الأخير
لو أراد طباعة قيمة في متغير فنستخدم الأمر كما يلي في هذا المثال
فلو فرضنا أن S=10 وأراد طباعة قيمتها فنكتب

Writeln (‘The Value of S =’,s:(

ويمكن أيضا طباعة أكثر من قيمة أكانت جملة أو متغير وذلك بإستخدام الفاصلة [،] بينها
مثل لو عرفنا المتغيرات التالية

A = ‘Rayan :’

B: ‘=’ =

C= ‘20:’

Writeln (‘My Name is’,B,A,’and my age is’,c:(

فسيكون الناتج على الشكل

My Name is =Rayan and my age is 20

الدرس السادس : أمر Read والأمرReadln

سنتعرف في هذا القسم على نوع جديد من الأوامر في الباسكال وهذا النوع من الأنواع الشائعة الإستخدام في البرنامج وهو الأمر Readln ويستخدم هذا الأمر لإدخال البيانات من المستخدم وحفظها في متغيرات تناسب نوع البيانات المدخلة وله نوعان

الأول : Readln
وهو يقوم بقراءة البيانات من المستخدم وبعدها ينتقل المؤشر للسطر التالي
الثاني : Read
وهو مشابه للأمر Readln ولكن الفرق الوحيد أنه لا ينتقل سطر للأسفل بعد قراءة البيانات بل يكمل بنفس السطر

```
Write ('Enter Your Name:');
Readln (Name:);
```

في المثال السابق يقوم البرنامج في السطر الأول بطباعة السؤال (أدخل إسمك) ولإننا إستخدمنا الأمر Write فإن المؤشر لا ينتقل للسطر التالي بل يقرأ السطر التالي والمؤشر في نفس السطر
في السطر الثاني ينتقل البرنامج لوضع إستقبال البيانات من المستخدم ومن ثم حفظها في المتغير المختار Name و يجب أن تكون البيانات المدخلة من نفس نوع المتغير وإلا سوف يولد البرنامج رسالة خطأ

Enter your name : I

لو أضفنا السطور التالية للمثال

```
Writeln ('Enter your Age:');
Read (Age:);
Write ('Enter your Salary:');
Readln (Salary:);
```

في السطر الثالث يطبع البرنامج الرسالة وينتقل بعدها للسطر التالي لإستقبال البيانات من المستخدم ، ونلا حظ هنا اننا إستخدمنا الأمر Read
بعدها سيقوم البرنامج بطباعة السطر الخامس ولن ينتقل للسطر التالي بل سيستقبل البيانات في نفس السطر لإننا إستخدمنا الأمر Write

الدرس السابع : قاعدة الأولويات
يجب علينا قبل الدخول في البرمجة فهم الطريقة الرياضية للتعامل مع البيانات وطريقة تعامل المصنف مع الدوال الرياضية والأقواس

وبرنامج الباسكال يتعامل مع العمليات الرياضية حسب القاعدة التالية
والتي تعتمد على الأسبقية بحيث
أولا : تنفيذ الدالة Not
ثانيا : تنفيذ الدوال التالية
AND
MOD
DIV
/
*
بحيث أن الدالتين Div و Mod دوال رياضية سيتم التعرف عليها لاحقا
ثالثا : تنفيذ العمليات
OR

-

+
رابعا : تنفيذ المعاملات المنطقية

=

<>

=>

=<

>

<
عندما يكون هناك دالتين من نفس المستوى ننفذ الدالة بدءا من اليسار إلى اليمين
لتتضح الرؤيا حول إستخدام قاعدة الأولويات نرى الأمثلة التالية

١- Write a PASCAL program to find the roots of a quadratic equation (assume that b2 – 4ac >= 0)

يريد في هذا المثال إيجاد جذور لمتتابعة ثنائية

الحل

```
Program Root(input,output;(
Var
A,b,c :integer;
X1,x2 : real;
```

الناتج عرفناه على أنه عدد حقيقي لإنه سيكون ناتج من عمليات منها عمليات قسمة

وفي حالة وجود عمليات قسمة فناتجها دائما عدد حقيقي وليس صحيح

```
Begin
Writeln ('Enter the Numbers A , B ,C:('
Readln (a,b,c:(
X1:= (-b+sqrt(b*b –4*a*c))/(2*a:(
X2:= (-b-sqrt(b*b –4*a*c))/(2*a:(
Write ('The Nunmber X1 =',x1 ,'And Number X2 =',x2:(

End.
```

٢- Find (7 Div 2/3)

في هذا المثال نلاحظ أن الدالتين Div و / يأتيان في مرتبة واحده في أولوية التنفيذ
ولكن لإن Div أتت قبل من جهة اليسار فنقوم بتنفيذها أولا لذلك

```
Vdiv 2/3=
= ٣ / ٣
١.٠
```

(لاحظ أن العدد جوابه ١.٠ وليس ١ لإنه ناتج من قسمة فلذلك يعد عدد حقيقي Real ولا يعتبر Integer)

الدرس الثامن : التعابير الرياضية

في هذا القسم سوف نتعرف على العمليات الرياضية المستخدمة في الباسكال وطريقة إستخدامها

أولا : العمليات - و + و *

التعامل مع هذه العمليات متشابة وهي تقبل الأعداد الحقيقة والصحيحة

٥ + ٧ = ١٢
٥.٠ + ٧ = ١٢.٠
٥– ٧ =٢
٥*٧ = ٣٠

نلاحظ من هذا المثال أن ناتج

عدد صحيح Integer + عدد صحيح Integer = Integer
عدد حقيقي Real + عدد حقيقي Real = Real
عدد صحيح Integer + عدد حقيقي Real = Real

ثانيا : العملية /

عملية القسمة دائما مهما كان نوع المدخلات أكانت أعداد حقيقة أو صحيح
فجوابها دائما عدد حقيقي Real
طبعا القسمة على الصفر لا تصح وتولد خطأ بالبرنامج وهو خطأ من النوع الذي يظهر عند تشغيل البرنامج فيجب الإنتباه

ثالثا : العملية Div

وهي عملية قسمة ولكن فقط تعطي الناتج من القسمة وتهمل الباقي من القسمة

٢٠Div 3 = 6

١٨Div 3 = 6

(١٧-)Div 3 = -5

١٩Div 3 = 6

٣Div 10 = 0

١٦Div (-3) = -5

الدالة Div تقبل دائما مدخلات من النوع الصحيح Integer وتعطي جواب دائما Integer

ولا تقبل أبدا القيم الحقيقية Real

١.٤Div 4 = X

ثالثا : العملية Mod

وتعطي هذه العملية الباقي من قسمة عددين ولاحظ دائما عند قسمة عدد على أكبر منه فناتجها العدد نفسه

٢٠Mod 3 = 2
١٨Mod 3 = 0
١٩Mod 3 =1
٣Mod 10 = 3

الدالة Mod تقبل دائما مدخلات من النوع الصحيح Integer وتعطي جواب دائما Integer
ولا تقبل أبدا القيم الحقيقية Real

١.٤Mod 4 = X

هذه الدورة من موقع الموسوعة العربية للكمبيوتر
http://www.c4arab.com