جامعة الجزائر3

كلية العلوم الاقتصادية، العلوم التجارية وعلوم التسيير

قسم العلوم التجارية

مدخل الى الخوارزميات

بن يحيى ثاني نسرين

أستاذة محاضرة قسم ب

بن يحيى ثاني نسرين

الفهرس

تمهيد 3
الفصل الأول: مدخل إلى الخوارزميات 5
I. مقدمة 5
II. أصل الخوارزميات 5
III. تعاريف 6
IV. البنية العامة للخوارزميات 10
الفصل الثاني: أنواع المتغيرات و الثوابت 12
I. مقدمة 12
II. المتغيرات 12
1.تصريح المتغيرات 13
2.العمليات على المتغيرات 16
III. تعريف الثوابت 17
الفصل الثالث: التعليمات الأساسية 19
I. مقدمة 19
II. تعليمة التخصيص 19
1.العمليات الحسابية و أولوياتها 21
2.العمليات المنطقية و أولوياتها 23
III. تعليمات الإدخال و الإخراج 24
1.تعليمة الإدخال (القراءة) 24
2.تعليمات الإخراج (الكتابة) 25
IV. التعليقات 27
الفصل الرابع: تعليمات المراقبة 30
I. مقدمة 30
II. البنية الخطية 30
III. البنية الشرطية 31
1.تعليمة الشرط البسيطة 31
2.تعليمة الشرط المركبة 32
IV. البنية التكرارية 35
1.تعليمة تكرارية بعداد ثابت : الحلقة Pour 37
2.تعليمة تكرارية بدون عداد : الحلقة Tant que 39

ابن يحي ثاني نسرين

3.تعليمة تكرارية بدون عداد : الحلقة Répéter 40

الفصل الخامس: المتغيرات المرقمة 42

I. مقدمة 42

II. الشعاش أو الجدول ذو بعد واحد 42

1. تعريف الجدول 43

2.تصريح الجداول 43

3.العمليات على الجداول 44

4.بعض الخوارزميات الأساسية حول الجداول 46

III. المصفوفة أو الجدول ذو بعدين 56

1.تعريف المصفوفة 56

2.تصريح المصفوفات 57

3.العمليات على المصفوفات 58

الخاتمة 61

المراجع 62

جدول أسكي 64

أبن يحي ثاني نسرين

تمهيد

تهدف هذه المطبوعة طلبة السنة الثانية علوم التسيير علوم تجارية و علوم اقتصادية، لا تتطلب معرفة أي مبادئ أولية للبرمجة، لأنها تحتوي على مجموعة المفاهيم الأولية للخوارزميات مع بعض التمارين المحلولة.

الهدف من دراسة مقياس الخوارزميات هو جعل الطالب يفهم آليات البرمجة و منه إمكانية كتابة برامج لحل مشاكل معينة و إمكانية التعامل مع البرمجيات الأخرى التي سيستعملها في باقي مشواره التكويني أو المهني.

كتابة الخوارزميات تكمننا من جعل الآلة تقوم بعمل ما (حل المشاكل) في مكاننا، مثلا: حساب معدلات طلبة لسداسي معين أو اعداد الفواتير ...، والمشكل المطروح هو طريقة الشرح للآلة كيف يجب أن تقوم بالعمل و التعامل مع الالة التي لا تفهم سوى النظام الثنائي (مجموعة من '1' و '0' متتالية تمثل مرور وانقطاع التيار في الالة)، فكيف نقول لها ذلك؟ كيف نعلمها ذلك؟ و كيف نتيقن من أن الآلة تقوم بالعمل كما لو قمنا به نحن؟ بل أحسن منا؟

تعلم الخوارزميات يسمح بالإجابة عن هذه الأسئلة حيث يمكننا من :

- حل المشاكل كالآلة.
- تعلم شرح طريقة التفكير.
- تعلم كتابة طريقة التفكير بشكل رسمي.
- تصميم وكتابة خوارزميات وبالتالي البرامج.
- قراءة برامج جاهزة.
- التعامل مع البرمجيات التي نستخدمها.

أبن يحي ثاني نسرين

الفصل الأول: مدخل إلى الخوارزميات

I. مقدمة

لما نسأل عن كيفية كتابة برنامج ما، غالبا يكون الجواب: C ، pascal ، java ، delphi ...مختلف لغات البرمجة و يعرف أنها مجموعة من الأوامر توصلنا إلى غاية ما.

لكن القليل فقط يعرف أن كل هذه اللغات أصلها الخوارزميات، حيث أن الخوارزمية هي أيضا عبارة عن مجموعة من الأوامر مثل كل لغات البرمجة الأخرى والفرق بينها هو أن الخوارزمية تكتب على الورق وليس على الحاسوب.

فهي أول مرحلة لكل مبرمج يريد صنع برنامج، حيث يكتب أولا خوارزميته ثم يستبدل التعليمات باللغة التي يريد البرمجة بها.

II. أصل الخوارزميات

تعتبر الخوارزميات من أهم العلوم في فرع الإعلام الآلي وقد أسسها سنة 825م العالم المسلم أبوعبد الله محمد بن موسى الخوارزمي يكنى أبو جعفر (781م / 850م).

الخوارزمي عالم في الرياضيات والفلك (ترك بصمته على نحو مميز):

- وضع مبادئ علم الجبر في كتاب "الجبر والمقابلة" ومنه دخلت كلمة "Algebra" معظم لغات العالم.

- أما كلمة خوارزمية "Algorithme" فجاءت بعد ترجمة كتاب هام له في الحسابات لجداول الضرب والقسمة وعدد من عمليات الحساب العشرية" كتاب الجمع والتفريق بحساب الهند".

فاتخذت كلمة خوارزمية في عصر البرمجة لتعبر عن خطوات وآليات حل مسألة تمهيدا لبرمجتها حاسوبيا.

III. تعاريف

قبل الدخول في التعقيدات يجب علينا معرفة مراحل عمل برنامج ما.

في عالم الإعلام الآلي يوجد ثلاث أطراف للحوار:

- المستخدم : هو الشخص الذي يواجه مشكل ما و يريد حله بالإعلام الآلي، حيث أن المستعمل يرى في الكمبيوتر المزود بالبرنامج المناسب أداة لتيسير مهامه فهدفه إذا هو تعلم الاستخدام الأفضل لهذه الأداة.

- المبرمج : هو الشخص القادر على توجيه الحاسوب في حل المشاكل التي يطرحها المستخدم، فالحاسوب بالنسبة للمبرمج هو منفذ للأوامر و يجب توجيهه.

- الحاسوب : هو الذي لا يعرف سوى تنفيذ الأوامر، فيكفي أن يكون البرنامج سليم من حيث الكتابة ليقوم بكل ما هو مطلوب منه.

قبل كتابة أي برنامج لحل مسألة ما، يجب أن يتوفر لدينا فهم شامل للمسألة المطروحة. وهذا يتضمن تحديد وتوصيف المعطيات التي نعتمد عليها أو ننطلق منها، والنتائج التي نريد الوصول إليها.

فلكتابة برنامج نتبع أسلوبا منهجيا للحل، ونعبر عن هذا الحل بطريقة مهيكلة كافية لتنقلنا فيما بعد إلى لغة برمجة معينة دون عناء كبير.

الخوارزمية: هي متتالية منتهية من التعليمات البسيطة المعرفة بشكل دقيق وترتيب محدد للحصول على حل لمشكل ما، دون التقيد بلغة برمجة محددة.

تتميز الخوارزمية بالصفات التالية:

- يجب أن تكون مجموعة التعليمات منتهية،
- يجب أن تكون محددة ودقيقة،
- يجب تحديد مجال تعريف معطيات الدخل إن وجدت (أعداد صحيحة، حقيقية، أحرف،...)،
- يجب أن تكون هناك نتيجة واحدة على الأقل.

- يجب أن تكون التعليمات كلها قابلة للتنفيذ.

ملاحظة :

قد نجد للمسألة الواحدة عدة خوارزميات، لكن الخوارزمية الأفضل هي الأقل استهلاكا لموارد الحاسوب أي التي تمتاز بتنفيذ زمني منخفض وبأقل قدر ممكن من استهلاك الذاكرة.

كانت الخوارزميات تابعة لعلم الرياضيات إلا أنها أصبحت طريقة عامة تستعمل بالخصوص في البرمجة المعلوماتية التي يطبق أوامرها معالج الحاسوب (Microprocesseur).

مثال1: لو قدمنا لطفل صغير المسألة الرياضية التالية:

اشترى تاجر 20 صندوق تفاح وكل صندوق يحتوي على 100 حبة، سعر كل حبة 30 دج، فكم سعر كل الصناديق؟

هذا الطفل لا يدرك مباشرة الإجابة، بل يقوم بالتحليل أولا ويجعل كل جزأ من المسألة وحده، فتكون الإجابة بالطرق المنطقية التالية:

- كل صندوق فيه 100 حبة، وعدد الصناديق 20 إذا عدد التفاح 2000 حبة،
- سعر كل حبة 30 دج، إذا سعر كل الحبات 60000 دج.

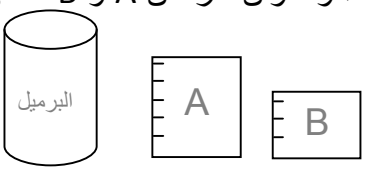
نلاحظ أن الطفل قد استوعب المعطيات ولكن لم يجاوب مباشرة إلا بعد مروره بخطوتين منطقيتين مرتبتين ولولاهما لما أدرك الإجابة.

إذا المعطيات هي المدخلات التي نقدمها للحاسوب، أما الجواب الذي قدمه لنا الطفل فهو المخرجات، في الحاسوب تكون إما عبر الشاشة أو عن طريق الطابعة، لكن الحاسوب لا يستطيع أن يمر بتلك الخطوات مثل الإنسان و لكنه يستوعبها بعد أن نخبره نحن بها، مثلا لو كانت هذه المسألة موجهة للحاسوب فلن يجبنا لأنه آلة و لكن علينا أن نخبره بما يجب فعله:

- أضرب 100 في 20،
- أضرب 30 في نتيجة المرحلة1.

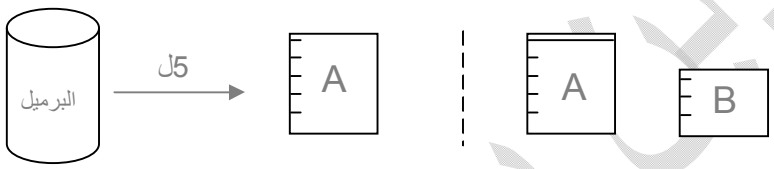
سيفهم الحاسوب عن طريق عملية الترجمة وسيجدها مرتبة وما عليه إلا التطبيق.

مثال 2 : لدينا برميل مملوء بالماء و دلوان فارغان A و B سعتهما على الترتيب 5ل و 3ل.

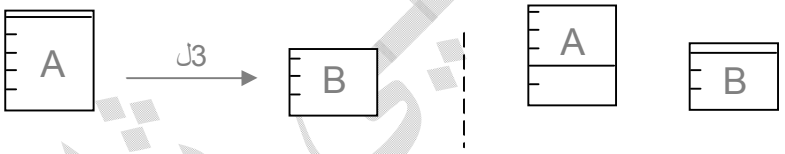


باستعمال البرميل والدلوان فقط، نريد التحصل على كمية 4ل من الماء، ما هي المراحل الأساسية لحل هذا المشكل؟

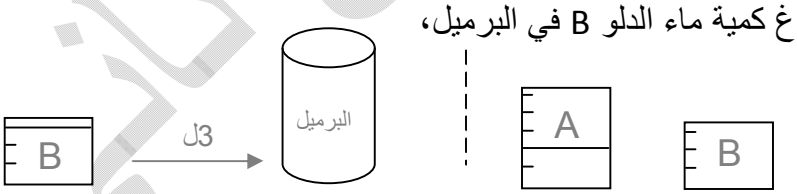
1- نملأ الدلو A ب 5ل من البرميل،



2- نملأ الدلو B ب 3ل من الدلو A ، فيبقى في الدلو A 2ل،

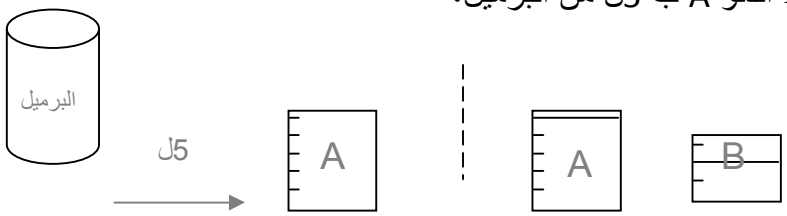


3- نفرغ كمية ماء الدلو B في البرميل،



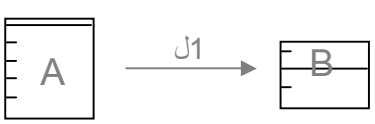
4- نضع محتوى الدلو A في الدلو B يصبح لنا 2ل في الدلو B، 0ل في A،

5- نملأ الدلو A ب 5ل من البرميل،



6- نملأ الدلو B إلى أن تصبح الكمية فيه 3ل (فيجب أخذ 1ل من الدلو A) فيبقى في الدلو A 4ل و هو المطلوب.

ملاحظة



دور المثالان السابقان هو تفكيك المشكل الى تعليمات بسيطة متتالية قابلة للتنفيذ.

تبحث الخوارزميات عن الحلول لمسائل معينة في المعلوماتية بغية تحويلها إلى برامج، كما تبحث أيضا عن الحل الأفضل (الأمثل) في حال تعدد الحلول.

الخوارزمية Algorithme هي طريقة لحل مشكلة.

البرنامج Programme هو ترميز سهل القراءة من طرف الحاسوب لهذه الطريقة، عبر ترجمة الخوارزمية الى لغة برمجة معينة (مرحلة البرمجة étape de programmation). البرنامج عبارة عن مجموعة من التعليمات المتسلسلة التي تسمح بأداء مهمة معينة.

لغة البرمجة langage de programmation هي وسيط بين الإنسان و الآلة، يُمَكِنْ من كتابة الخوارزمية بلغة قريبة للحاسوب من طرف البشر.

ملاحظة :

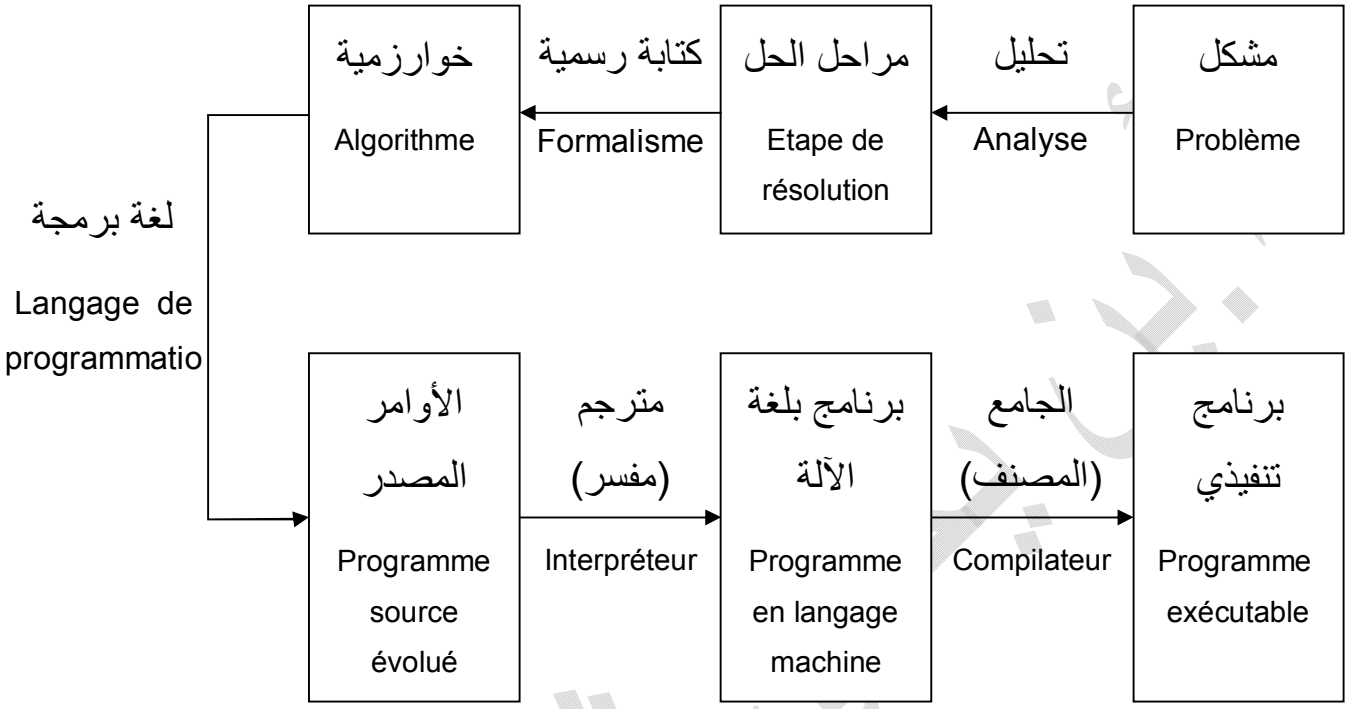
هناك عدة لغات برمجة منها: c، cobol ، fortron ، basic، pascal، java، c++ ، c# ، Delphi...

لغة البرمجة الأكثر تجاوبا و استعمالا للجامعيين فهي مهيكلة بامتياز للتعليم، و تعتبر آداة تعليمية ممتازة للمبتدئين في البرمجة.

علما أن الحاسوب يمكنه فقط معالجة معطيات في النظام الثنائي، أي سلسلة من 0 و 1، فمن الضروري استعمال لغة برمجة التي بدورها يتم تحويلها إلى لغة آلة (langage machine) و ذلك بطريقتين:

1- يقوم المفسر (المترجم l'interpréteur) بقراءة الأوامر المصدر ثم يترجم البرنامج إلى إجراءات مباشرة (وجود المترجم ضروري لتشغيل البرنامج من طرف شخص ما).

2- يقوم الجامع (المصنف compilateur) بترجمة الأوامر المصدر إلى برنامج تنفيذي exécutable يمكن تشغيله في وقت لاحق مع إمكانية توزيع البرنامج التنفيذي على أشخاص لا يملكون المترجم.



IV. البنية العامة للخوارزميات

هناك ثلاث أقسام رئيسية ضرورية لأي خوارزمية و هي:

رأس الخوارزمية أو قسم الديباجة Partie Entête: يتكون من الكلمة المخصصة Algorithme يليها الاسم الذي يعرف الخوارزمية.

قسم الإعلانات Partie Déclaration: يتكون من المتغيرات و/أو الثوابت التي نستخدمها في الخوارزمية.

جسم الخوارزمية corps de l'algorithme أو جزأ التعليمات Partie Instruction: هي مجموعة من التعليمات التي تبدأ بتعليمة الفتح و هي الكلمة المخصصة Début، ثم كتلة من التعليمات المتعلقة بالمهام التي يجب القيام بها، ثم ينتهي بالكلمة المخصصة Fin.

ومنه البنية العامة للخوارزمية (الهيكل العام) تكون كالتالي:

الديباجة Entête

الإعلانات Declaration

جسم الخوارزمية Corps de l'algorithme

```
Algorithme <nom_de_l_algorithme>
    <Définitions des constantes>
    <Déclarations des variables>
Debut
    <Partie instructions>
Fin
```

ملاحظات هامة :

الكلمات المخصصة مثل Algorithme، Debut، Fin تستعمل في جميع الخوارزميات لها استخدام محدد و لا يمكن استخدامها خارج سياقها.

<nom_de_l_algorithme>، <Définitions des constantes> ، <Partie instructions> متعلقة بالخوارزمية، فتكون محددة لكل خوارزمية، نتطرق لها في الفصول القادمة.

الفصل الثاني: أنواع المتغيرات و الثوابت

I. مقدمة

بعدما رأينا في الفصل السابق المفاهيم الأساسية للخوارزميات و آليات التعامل مع الحاسوب عن طريق المفسر و المصنف، وأيضا البنية العامة للخوارزمية، نتطرق في هذا الفصل لدراسة قسم الإعلانات للخوارزمية حيث يكون تصريح المتغيرات و تعريف الثوابت.

مثال : إذا كان عندنا : c=a+b، a=5 و b=8 نفهم سريعا أن c=13 في الحالة العامة a، b، و c تنتمي لمجموعة الأعداد الصحيحة فنسمي هذه الحروف متغيرات.

يجب إعلام الحاسوب قبل البداية أن a، b، cموجودة في الخوارزمية و أنها أعداد صحيحة نسمي ذلك تصريح المتغيرات.

II. المتغيرات

المتغيرة variable هي عنصر مستعمل في الخوارزمية مع إمكانية تغيير قيمتها و قد تكون معطيات الدخل أو النتيجة الأخيرة لعملية حسابية أو حاصل عملية انتقالية (وسيطة intermediaire).

ملاحظة : المتغيرة في الإعلام الآلي مختلفة عن المتغيرة في الرياضيات، هي عبارة عن اسم يقدم لخانة (مكان) في الذاكرة RAM بحجم معين، يتم فيها تخزين البيانات ليتم معالجتها و الاستفادة منها لإنجاز عمل معين.

يمكننا أن نعتبر الذاكرة المركزية RAM كمجموعة من العلب، فالمتغيرة عبارة عن علبة يعرفها البرنامج بواسطة علامة توضع عليها، فلكي يدخل البرنامج إلى محتوى العلبة يكفي الإشارة إليها بعلامتها، قيمة المتغيرة هي محتوى

العلبة لذا يقوم الحاسوب بحجز مساحة (مكان) مناسب لها.

نعلم أن خانات الذاكرة لا تحتوي إلا على '0' و '1' لكنها قد تعبر عن عدد صحيح، حقيقي، حرف، صورة، صوت،...

ملاحظة :

استعمال المتغيرات في الخوارزميات هو ما يسمح بحل نفس المشكل بقيم مختلفة للمعطيات.

1. تصريح المتغيرات

في الخوارزميات يجب التصريح بكل المتغيرات التي تستعمل بأنواعها (صحيح، حقيقي، حرف،...) لكي يتمكن الحاسوب من توفير الخانة الملائمة لكل منها.

تُصَرَحْ المتغيرة باثنان الاسم التعريفي Identificateur و النوع (النمط) Type .

- التصريح في حالة متغيرة واحدة :

< nom_var > : < type >

النوع الاسم التعريفي

- التصريح في حالة عدة متغيرات من نفس النوع:

< nom_var1 >, < nom_var2 > , < nom_var3 > : < type >

- التصريح في حالة عدة متغيرات من أنواع مختلفة:

< nom_var1 1>, < nom_var12 > : < type1 >

< nom_var21 >, < nom_var22 > : < type2 >

< nom_var31 >, < nom_var32 > : < type 3 >

يتصرف الحاسوب كموزع البريد حيث أن الاسم التعريفي يعتبر كعنوان لخانة الذاكرة، فهو الذي يسمح بالوصول إلى المكان الصحيح في مرحلة الترجمة (compilation) والتوصل إلى البرنامج التنفيذي الذي يسمح بموافقة اسم المتغيرة مع المكان المطابق لها في الذاكرة. أما النوع فهو المساحة المخصصة لهذه المتغيرة.

أ- الأسماء التعريفية

يتكون الاسم التعريفي من حروف لاتينية a,...,z و A,...Z و الأرقام 0,...,9 و العلامات الخاصة _، $. (يستعمل للمتغيرات، الثوابت، الخوارزميات، الدوال،...)

عند اختيار اسم تعريفي يجب مراعاة القواعد التالية:

- يبدأ الاسم التعريف بحرف أو رمز $ و ليس برقم.
- لا يحتوي على مساحات فارغة.
- لا يكون من الكلمات المخصصة (المحجوزة).
- لا يحتوي على أي علامة خاصة غير مذكورة (مثلا : أس ^{2} puissance ، مؤشر أو هامش $_{1}$ indice، %، &، π...)
- أن لا يزيد الاسم عن 255 حرف.
- أن لا يكون الاسم مستعملا لمتغيرة أو ثابت أو دالة أخرى مسبقا.
- يفضل أن يكون الاسم معبرا عن ما يقوم به المبرمج (مثلا : Prix أحسن من P).

ملاحظة: اسم المتغيرة ليس محدد (لا يتجاوز 255 حرف) لكنه في لغة الباسكال مثلا الجامع (compilateur) يأخذ فقط بعين الاعتبار الثمانية حروف الأولى.

أمثلة:

المتغيرة ABDELRAHMEN و المتغيرة ABEDLRAHIM تعتبران كنفس المتغيرة من طرف مصنف (compilateur) الباسكال.

X1، X2، Maison، somme، delta، $b4 هي أسماء متغيرات مقبولة.

X1، 4Y، 'X، ''X، *، A%، #، ∑ ، Ω ليست أسماء متغيرات مقبولة.

ب- أنواع المتغيرات

يمكن للخوارزمية أن تستعمل مختلف أنواع المعطيات نذكر منها:

- الأعداد الصحيحة (Entier) : 125,-4,+12,...
- الأعداد الحقيقية (Réel) عبارة عن سلسلة أرقام عشرية مع النقطة التي تفرق بين الجزء العشري و الجزء الكسري: 8.0, -12.4, 0.0,...
- الأحرف (caractère) تعبر عن كل أحرف الحاسوب التي يحتاجها للاتصال مع العالم الخارجي. الحرف يجب أن يكون محصور بين علامتي حذف " " guillemets: "A"، "v"، "1"، "&"،...
- سلسلة حرفية (chaine de caracteres) عبارة عن مجموعة من الأحرف محصورة بين علامتي حذف : "Mehdi"، "2017"، "Bonjour"،...
- منطقي (Booleen) متغيرة يمكنها فقط أخذ قيمتين صحيح أو خطأ (Vrai ou Faux).

ملاحظة: كل هذه البيانات تُشَفَرْ (ترمز seront codée) على شكل octet في الذاكرة.

Byte (octet) : 0 à 255 i.e. 28

Entier (16 Bits) : -32768 à 32767

Reel (32 Bits) : -3.40 E 38 à -1.40 E -45 valeurs négatives

1.40 E -45 à 3.40 E 38 valeurs positives

Caractere (8 Bits) : de 0 à 255 caractères ASCII

Chaine caracteres (256 Bits)

Booleen (1 Bit) : vrai ou faux

أمثلة :

Rayon : reel

Compteur : entier

C, lettre : caractere

N1, N2, N3, Moy : reel

Adm : booleen

ملاحظة:

الكلمات Entier، Reel، caractere، chaine de caracteres و Booleen هي كلمات مخصصة.

2. العمليات على المتغيرات

أ- العمليات على الأعداد

العمليات الحسابية		العمليات المنطقية	
+	للجمع	=	للمساواة
-	للطرح	<>	للاختلاف (لا يساوي)
*	للضرب	>	أصغر
/	للقسمة	<	أكبر
^ أو **	للأس	=<	أصغر أو يساوي
		>=	أكبر أو يساوي

ب- العمليات على الأحرف و السلاسل الحرفية

عمليات المقارنة: يمكننا مقارنة الحروف باستعمال رموز عمليات المقارنة : =، <>، >، =>، <، =<.

كما نعلم أن للمتغيرات الحقيقية و الصحيحة ترتيب هو : ...>3>2>1 فمجموعة الأحرف مزودة بعلاقة الترتيب التالية : ...>z>...>a>b>...>Z>...>A>B>...>9>...>3>2>1 وهي حسب ترميز ASCII.

عملية التسلسل: عملية '+' تسلسل (concaténation) تسلسل بين حرفين أو سلسلتين حرفيتين مثال: ''A'+'B' يكون 'AB'.

"Dely "+ "Brahim" تصبح "Dely Brahim".

ت- <u>العمليات على المتغيرات المنطقية</u>

<u>عمليات المقارنة:</u> = المساواة ، <> اختلاف (غير مساواة).

<u>المعاملات المنطقية</u> هي : ET(و ∧)، OU (أو ∨)، NON (لا–) و تكون النتيجة العمليات من النوع المنطقي.

العمليات على المتغيرات المنطقية تتبع (تطيع) قانون الجبر البولي Algèbre de Bool. إذا كانت a و b متغيرتان منطقيتان، فيكون جدول الحقيقة المقابل للعمليات المنطقية التي تجرى على هتان المتغيرتان هو:

A	B	NON A	A ET B	A OU B
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

III. تعريف الثوابت

الثابت هو عبارة عن قيمة ثابتة في البرنامج و المقصود بثابتة أنها لا تتغير عند تشغيل البرنامج التنفيذي، لتغييرها يجب تعديل نص البرنامج CODE SOURCE قبل الترجمة ثم إعادة بناء البرنامج التنفيذي.

نعرف الثابت في قسم الإعلانات كما يلي:

< nom_constante > = < valeur >

القيمة الاسم التعريفي للثابت

حالة تعدد الثوابت :

< nom_const1 > = < val1 >, < nom_const2 > = < val2 >,< nom_const3 > = < val3 >

<u>أمثلة :</u>

Pi=3.14

Moy=10, Etat= ‘’ADM ‘’

أ. بن يحي ثاني نسرين

الفصل الثالث: التعليمات الأساسية

I. مقدمة

بعدما رأينا مفهوم المتغيرات والثوابت نتطرق في هذا الفصل إلى التعليمات الأساسية التي توجد في قسم جسم الخوارزمية وتسمح باستعمال المتغيرات والثوابت مشكلتا مجموعة المراحل لحل مشكل ما.

II. تعليمة التخصيص

التخصيص (Affectation ou mémorisation) هي تعليمة أساسية تسمح بنسب (attribuer) قيمة لمتغيرة (التي كنا قد خصصنا لها مكان في الذاكرة في قسم الاعلانات).

في علم الخوارزميات نرمز لهذه التعليمة ب " ←" كما يلي :

< Variable > ← < Expression >

متغيرة عبارة

تكون دائما المتغيرة على اليسار، أما على اليمين نجد دائما قيمة لعبارة.

العبارة قد تكون قيمة (مثال : 40← surface)، متغيرة أخرى (مثال : note2←note1)، نتيجة معادلة (مثال : resultat ←sqrt(x)) أو عملية حسابية تكون بمختلف هذه العناصر (مثال: Pi*rayon**2 ←surface).

<u>ملاحظة:</u> إذا كان عندنا note2←note1 فقيمة note1 هي التي تتغير تمحى في الذاكرة و تعوض بقيمة note2 أما قيمة note2 فلا تتغير.

ترتيب التعليمات المكتوبة يلعب دورا مهما في النتيجة.

لنلاحظ البرنامجين التاليين:

Algorithme Test1	Algorithme Test2
a :entier	a :entier
Debut	Debut
a←8	a←15
a←15	a←8
Fin	Fin

ما هي القيمة النهائية للمتغيرة a في كلتي الخوارزميتين؟

في الخوارزمية الأولى قيمة a هي 15.

في الخوارزمية الثانية قيمة a هي 8.

في كلتي الخوارزميتين تأخذ a القيمة المخصصة الأخيرة لأنها تلغي (écrase) القيمة السابقة لها.

ملاحظة:

- لكي تكون تعليمة التخصيص صحيحة يجب أن تكون المتغيرة و قيمة العبارة من نفس النوع.
  مثال: لا يمكننا تخصيص قيمة من نوع حرف إلى قيمة من نوع عدد حقيقي.
- نتيجة عملية حسابية يكون من النوع الحقيقي إذا كانت على الأقل واحدة من متغيرات العملية من النوع الحقيقي (لأن تمثيل الآلة بالنظام الثنائي لمتغير من النوع الحقيقي يكفي لمتغيرة من النوع الصحيح لكن العكس ليس صحيح).
- نتيجة عملية تسلسل تكون من النوع سلسلة حرفية.
- نتيجة عملية قسمة أو جذر تربيعي تكون من النوع الحقيقي.

أمثلة:

X←4 : يعني نضع القيمة 4 في خانة الذاكرة المصرحة للمتغيرة X.

X←Y : نضع القيمة الموجودة في خانة الذاكرة المصرحة للمتغيرة Y داخل خانة الذاكرة المصرحة للمتغيرة X.

i←i+1 : تعريف المتغيرة يسمح بالقيام بهذه العملية لأننا نحسب أولا قيمة العبارة الموجودة على اليمين ثم نضعها

في المكان المخصص للمتغيرة الموجودة على اليسار.

لتكن التصريحات التالية:

a,a1 :entier

b :reel

c :booleen

d :caractere

التخصيصات التالية صحيحة :	التخصيصات التالية خاطئة :
a←a1	d←c
b←a	c←d
a←a+2	c←b
	a←b

1. العمليات الحسابية و أولوياتها

العمليات الحسابية تقدر من اليسار إلى اليمين.

يجب الأخذ بعين الاعتبار الأولويات التالية : 1) ** 2) * و / 3) + و –

الأقواس تغير ترتيب الحساب (ما بين القوسين هو الذي يصبح أولوي).

أمثلة:

a-b-c نحسب أولا (a-b) ثم نطرح c.

a+b.c تكتب a-b*c نحسب أولا b*c ثم نضيف a.

$x^2.y^2$ تكتب x**2 * y**2 نحسب أولا x^2 و يخزن ثم يحسب y^2 و يخزن ثم يحسب الضرب بين النتيجتين.

$\frac{a+b}{c+d}$ تكتب (a+b)/(c+d) إذا كتبنا a+b/c+d تحسب أولا b/c ثم الجمعان.

$\frac{x}{y.z}$ تكتب x/y/z يمكننا كتابة x/(y*z) لكن تتطلب استعمال أقواس لا حاجة لها و هي عملية إضافية من طرف الحاسوب.

$a+b.c^d$ تكتب a+b*c**d يحسب أولا c**d و يخزن ثم الضرب ب b ثم الجمع ب a.

<u>ملاحظة:</u>

- على عكس الرياضيات لا يمكن كتابة التعليمات الحسابية إلا على سطر واحد فقط.
- يمكننا تركيب عدة أقواس فيما بينها لكن إن لم تتوازن سوف تنتج خطأ عند ترجمة البرنامج.
- يمكننا أيضا استعمال مجموعة من العوامل (عبارة عن دوال) مثلا:

- ABS(x): القيمة المطلقة ل x : $|x|$ La valeur absolue de x
- SQRT(x): الجذر التربيعي ل x: $\sqrt{x}$ La racine carrée de x
- FLOOR(x): القيمة الصحيحة الدنيا ل x: $\lfloor x \rfloor$ La partie entière inférieure de x
- CEIL(x): القيمة الصحيحة العليا ل x: $\lceil x \rceil$ La partie entière supèrieure de x
- ROUND(x): القيمة المقربة ل x: $\{x\}$ La valeur approchée de x
- EXP(x) : الدالة الأسية: e^x La fonction exponentielle
- LOG(x) : اللوغاريتم النيبيري ل x: $Ln(x)$ La fonction logarithme népérien

<u>ملاحظة:</u>

الحاسوب عبارة عن آلة غبية إذا جعلناه يحسب مرتان نفس العبارة لا يدرك ذلك، مثلا إذا كان لدينا مرتان أو أكثر x*y داخل تعليمة حسابية فانه من الأحسن حساب x*y و تخزين النتيجة في متغيرة z ثم استعمالها في وقت لاحق.

<u>مثال:</u>

أحسب قيمة الصيغة التالية محترما مبدأ الأولوية:

$$\frac{\sqrt{a+b}}{\sqrt{a+20}}$$

علما أن : d =2 c =10 b =4 a =5

لدينا :

$$SQRT(a+b)/SQRT(a+20)$$

$$SQRT(5+4)/SQRT(5+20)$$

$$\underset{(1)}{SQRT(5+4)}/\underset{(2)}{SQRT(5+20)}$$

$$\underset{(1)}{SQRT(\underset{(1.1)=9}{5+4})}/\underset{(2)}{SQRT(\underset{(2.1)=25}{5+20})}$$

$$\underset{(1)=3}{SQRT(9)}/\underset{(2)=5}{SQRT(25)}$$

$$\underset{(3)=0.6}{3.0/5.0}$$

0.6

2. العمليات المنطقية و أولوياتها

العمليات المنطقية تقدر من اليسار إلى اليمين.

يجب الأخذ بعين الاعتبار الأولويات التالية : 1) Non 2) Et 3) Ou

الأقواس تغير ترتيب الحساب (ما بين القوسين هو الذي يصبح أولوي).

مثال:

Non A Ou B Et C : تكون مكافئة ل (Non A) Ou (B Et C).

A+5>=B : هي عبارة منطقية التي تعني B=<(A+5) فإذا كانت قيمة A+5 أكبر أو تساوي B فقيمة العبارة هي Vrai و إلا فقيمة العبارة هي Faux.

$\overline{a \wedge b}$ تكتب NON(a ETb)

$\overline{a \wedge \bar{b}} \vee \overline{c \wedge b}$ تكتب NON(a ET NONb) OU NON(c ET b)

تمرين : ما هي نتائج العبارات التالية ؟

العبارات	النتائج

قيمة A هي : 20 قيمة B هي : 100 قيمة N هي : 120	A←20 B←A*5 N←A+B	قيم صحيحة
قيمة R هي : 25.0 قيمة P هي : 0.8	R←B/4 P←A/25	قيم حقيقية
قيمة E هي :Université Alger 3	C← " Université " D← " Alger 3" E← C+D	قيم سلاسل حرفية
قيمة F هي : Vrai قيمة G هي : Faux قيمة H هي :Vrai قيمة K هي : Vrai قيمة J هي : Faux قيمة M هي : Vrai	F← 6<10 G← 2>14 H← 6<9 K← F Et G Ou H J← G Et F Et H M← Non G	قيم منطقية

III. تعليمات الإدخال و الإخراج

هذه التعليمات تسمح بالاتصالات الأساسية بين الحاسوب و الإنسان و أيضا تسمح بتخزين بصفة دائمة و استعادة المعلومات على الوحدات الإدخال-الإخراج أو التخزين (les périphériques d'entrée-sortie ou de stockage) مثل (القرص الصلب disque dur، القرص المضغوط CD Rom قرص فيديو رقمي DVD Digital Video ou Versatile Disk) لأن الذاكرة المركزية لا تحتفظ بالمعلومات بعد تنفيذ البرنامج لأنها ذاكرة مؤقتة.

1. تعليمة الإدخال (القراءة)

تعليمة القراءة تسمح بأخذ المعلومات (القيم) من وحدات (محيطات) الإدخال ووضعها في خانات الذاكرة المخصصة

(للمتغيرة).

في حالة إدخال متغيرة واحدة : Lire (<variable>)

في حالة إدخال عدة متغيرات : Lire (<variable1>,<variable2>,…,<variablen>)

ملاحظة:

إذا كانت لوحة المفاتيح هي وحدة الإدخال للحاسوب و A هي متغيرة صحيحة، فان التعليمة Lire(A) تتوقع أننا ندخل عدد صحيح في لوحة المفاتيح ثم تخزن قيمته (مترجمة في النظام الثنائي) في خانات الذاكرة المخصصة ل (A).

مثال:

أكتب خوارزمية تسمح بقراءة الرقم الآلي و علامة طالب.

```
Algorithme lecture
Note :reel
Code :entier
Debut
Lire(code)
Lire (note)
```

} أو Lire(code, note)

```
Fin
```

2. تعليمات الإخراج (الكتابة)

تعليمة الكتابة تسمح بإرسال محتويات خانات الذاكرة على وحدات الإخراج و تسمح أيضا بكتابة رسالة أو جمل على وحدات الإخراج.

في حالة إخراج متغيرة واحدة : Ecrire (<variable>)

في حالة إخراج عدة متغيرات : Ecrire (<variable1>,<variable2>,…,<variablen>)

في حالة كتابة جملة : Ecrire (‘’ <Message>’’)

في الحلة الأخيرة كل ما هو داخل المزدوجتين يظهر على وحدة الإخراج كما هو.

ملاحظة:

إذا كانت الشاشة هي وحدة الإخراج فكتابة المتغيرة A ,Ecrire(A) تعرض محتويات خانات الذاكرة المخصصة للمتغيرة A على الشاشة (بعد تحويلها من النظام الثنائي إلى النظام العشري.)

كتابة Ecrire(''Erreur'') تعرض كلمة Erreur على الشاشة.

مثال:

أكتب خوارزمية تسمح بحساب مجموع سعري منتوجين.

```
Algorithme Prix
Som,prix1,prix2 :reel
Debut
Ecrire(''Donnez SVP les prix'')
Lire(prix1,prix)
Som← prix1+prix2
Ecrire(''la somme est : '')
Ecrire(Som)
Fin
```

أو Ecrire(''la somme est : '', Som)

ملاحظة :

الحاسوب يقرا على وحدة الإدخال و يكتب على وحدة الإخراج أما الإنسان فيفعل العكس.

تمرين : ما هي قيم المتغيرات a، b، c بعد تنفيذ التعليمات التالية:

الحل : التظاهر باليد

التعليمة	a	b	C
1	“20”	/	/
2	“20”	“17”	/
3	“20”	“17”	“2017”

Algorithme Exercice1

a,b,c :chaine de caractère

Debut

a← “20”

b← “17”

c← a+b

Ecrire(c)

Fin

الشاشة Ecran

2017

IV. التعليقات

يمكننا إضافة تعليقات في الخوارزميات تكون محصورة بين خطين مائلين (Slash) ونجمتين أو قوسين و نجمتين:

/*commentaires*/ أو (*commentaires*)

التعليقات ليست تعليمات لكنها عبارة عن شرح لتعليمات الخوارزمية لكي يسهل المبرمج قراءة و فهم خوارزميته من طرف مبرمج آخر، فالتعليقات لا تظهر في البرنامج التنفيذي.

أمثلة:

(*Initialisation des variables*)

/* Affichage des résultats */

/* Cas infinité de solutions */

مثال:

أكتب خوارزمية تسمح بحساب ضرب عددان مع استخدام التعليقات.

أ. بن يحي ثاني نسرين

```
Algorithme Produit
Nb1,nb2,prod :reel
Debut
/*Lecture des données*/
Ecrire(''veuillez saisir deux nombres'')
Lire(nb1,nb2)
(*Calcul du produit*)
prod← nb1*nb2
/*Resultat*/
Ecrire(''le produit est : '', prod)
Fin
```

تمرين : أكتب خوارزمية تسمح بحساب مساحة دائرة بعد طلب قيمة نصف قطرها مع استخدام التعليقات.

```
Algorithme Cercle
Pi=3.14
r,surf :reel
Debut
/*Lecture des données*/
Ecrire('Donnez le rayon du cercle'')
Lire(r)
/*Calcul de la surface*/
surf← r*Pi**2
/*Resultat*/
Ecrire(''la surface du cercle  est : '', surf)
Fin
```

أ. بن يحي ثاني نسرين

الفصل الرابع: تعليمات المراقبة

I. مقدمة

يتضمن البرنامج نوعين من التعليمات:

التعليمات الأساسية و هي التعليمات التي تسمح باستعمال المتغيرات كالتخصيص، القراءة، الكتابة،...

تعليمات المراقبة، وهي التعليمات التي توضح كيفية تسلسل التعليمات الأساسية.

في الفصل السابق لم نتطرق إلا للبرامج التي تنفذ فيها التعليمات حسب الترتيب الذي كتبت فيه، لكن قوة البرنامج تظهر في إمكانيته لاختيار تنفيذ مجموعة من التعليمات.

مثلا : لحساب معدل طالب، يمكننا اختيار المعدل حسب قيمته، اذا كان أكبر من 10 و اذا كان أقل من 10 في هذه الحالة نأخذ علامة الامتحان الاستدراكي.

تعليمات المراقبة تستعمل لمراقبة طريقة و ترتيب تنفيذ التعليمات.

تنقسم بنية البرنامج الى ثلاث فئات :

- بنية خطية من التعليمات،
- بنية شرطية (الاختيار) أو تناوبية : ينفذ فيها البرنامج مجموعة من التعليمات حسب نتيجة شرط معين.
- بنية تكرارية: تحت مراقبة شرط معين، تنفذ مجموعة من التعليمات بشكل متكرر.

II. البنية الخطية

هذا النوع من البرامج يتضمن مجموعة من التعليمات المتعاقبة تُنَفذ الواحدة تلوى الأخرى.

< partie actions >

```
Debut
  Action1
  Action2
  …
  Action n
Fin
```

مثال: خوارزمية حساب جمع عددين x و y.

III. البنية الشرطية

التعليمة الشرطية أو التناوبية تسمح بالاختيار بين معالجتين أي مجموعتين من التعليمات أو أكثر و ذلك تبعا لنتيجة شرط معين (أو تعبير شرطي Expression conditionnelle).

1. تعليمة الشرط البسيطة

مثال : إذا أمطر غدا، إذا سوف ألغي السفر.

تكتب تعليمة الشرط البسيطة كما يلي:

```
Si <condition> alors <traitement>
   الشرط          المعالجة
Fsi
```

- قد يكون الشرط :

عبارة منطقية تستعمل المعاملات المنطقية : =، <>، >، =>، <، =< و معاملات المقارنة Et، Ou، Non.

متغيرة منطقية (بما أن قيمتها صحيح أو خطأ، قد نخزن فيها عبارة منطقية ثم نمتحن قيمة المتغيرة).

أمثلة :

شرط بسيط : x=10 ، bool=faux.

قيمة منطقية : bool.

شرط معقد complexe: ((y>0) ou (z>0)) et (x>0).

- المعالجة تتضمن مجموعة من التعليمات.

المبدأ: إذا كان الشرط صحيح فننفذ مجموعة تعليمات المعالجة و إلاّ لا ننفذ شيء و نمر للتعليمة الموالية ل Fsi.

مثال:

أكتب خوارزمية تسمح بحساب x^y حيث

$$x^y = \begin{cases} x^y & \text{إذا كان y موجب} \\ x^{-y} & \text{إذا كان y سالب} \end{cases}$$

```
Algorithme puissance
 x,y,z :reel
Debut
 Ecrire(''veuillez saisir deux nombres'')
 Lire(x,y)
 Si y<0 alors y ←-y
 Fsi
 z←  x**y
 Ecrire(''le resultat est : '', z)
 Fin
```

2. تعليمة الشرط المركبة

مثال : إذا أمطر غدا، إذا أخرج على الساعة السادسة و النصف و إلا أخرج على الساعة السابعة صباحا.

تكتب تعليمة الشرط المركبة كما يلي :

```
المعالجة1                    الشرط
```



```
المعالجة2
```

```
Si <condition> alors <traitement1>
        Sinon <traitement2>
Fsi
```

المعالجة1 و المعالجة2 تتضمن كلا منهما مجموعة من التعليمات.

المبدأ: إذا كان الشرط صحيح فننفذ مجموعة تعليمات المعالجة1 ثم يتوجه المترجم مباشرة إلى التعليمة الموجودة بعد Fsi و إلاّ (خلاف ذلك) تنفذ مجموعة تعليمات المعالجة2 بدون تنفيذ مجموعة تعليمات المعالجة1 ثم ينفذ المترجم التعليمة الموجودة مباشرة بعد Fsi.

ملاحظة:

لا يمكن تنفيذ مجموعة تعليمات المعالجة1 و مجموعة تعليمات المعالجة2 معا (في نفس الوقت).

مثال:

أكتب خوارزمية تسمح بتحديد أكبر قيمة بين عددين.

```
Algorithme valeur
 x,y :reel
Debut
 Ecrire(''veuillez saisir deux nombres'')
 Lire(x,y)
 Si x>=y alors Ecrire('' la plus grande valeur est : '',x)
        Sinon Ecrire('' la plus grande valeur est : '',y)
 Fsi
Fin
```

ملاحظة :

يمكننا مداخلة عدة تعليمات شرط مركبة أو بسيطة فيما بينها في حالة وجود علاقة بين شروط كل منها (و ذلك عند وجود أكثر من اختيارين) أو استعمال عدة تعليمات شرط الواحدة تلوى الأخرى في حالة عدم وجود علاقة بين شروطها.

استعمال تعليمات الشرط المتداخلة فيما بينها تظهر المنافع التالية:

اقتصاد في كتابة البرنامج، عوضا لكتابة عدة تعليمات شرطية متتالية (برنامج سهل القراءة ومبسط).

اقتصاد في مدة التنفيذ، واحدة من الحالات فقط تنفذ، إذا كانت الحالة الأولى هي التي تنفذ فنجتنب دراسة الشروط الباقية (برنامج عالي الأداء performant).

مثال :

أكتب خوارزمية تقوم بحساب و طبع سعر المراسلات البريدية، مع العلم أنه في حالة:

- وزن (poids) الرسالة أقل من 20غ، السعر (prix) يساوي 2دج للغرام،
- وزن (poids) الرسالة ما بين 20غ و 50غ، السعر (prix) يساوي 5دج للغرام،
- وزن (poids) الرسالة أكثر من 50غ، السعر (prix) يساوي 7دج للغرام،

بالإضافة إلى أنه في حالة بعث الرسالة إلى دول الخارج (Internationale) يضاف للسعر، ضريبة (taxe) تقدر ب 10% من السعر الإجمالي.

```
Algorithme Lettre
   pd,pr :reel
   type : caractere
Debut
   Ecrire(''Donnez le poids'')
   Lire(pd)
   Ecrire(''Tappez I en cas d un envoie internationnal'')
   Lire(type)
   Si pd<20  alors pr←pd*2
            sinon si pd <=50 alors pr←pd*5
                        sinon pr←pd*7
                     Fsi
```

```
  Fsi
si type =''l'' alors pr←pr+pr*0.1
Fsi
 Ecrire(''Le prix est '',pr)
Fin
```

IV. البنية التكرارية

فائدة الحاسوب تكمن في جعله يكرر عدة مرات مجموعة من التعليمات و ذلك باستعمال عدد قليل من الأمور (ordre) إذا من الممكن تنفيذ تعليمة واحدة أو مجموعة من التعليمات بشكل متكرر.

على الرغم من أن التعليمات تبقى نفسها فان المعطيات هي التي تتغير مع كل عملية تكرارية.

تستعمل التعليمة التكرارية لتكرار تنفيذ مجموعة من التعليمات، يرتبط هذا التكرار بتحقق شرط، إذ يختبر الشرط، فإن كان الشرط محققا (صحيحا) تنفذ مجموعة التعليمات و يكرر تنفيذ هذه التعليمات مادام الشرط محقق تسمى هذه التعليمة بالحلقة التكرارية.

ملاحظة:

تبقى الحلقة أو التكرار مستمرا في تنفيذ مجموعة التعليمات واختبار الشرط حتى يصبح الشرط غير محقق، عندها ينتقل التنفيذ إلى التعليمة الموالية للتعليمة التكرارية.

مثال: إذا أردنا كتابة جدول الضرب للعدد 7.

فكرة الخوارزمية هي ما نقوم به فعلا : $7 = 1 \times 7$

$14 = 2 \times 7$

...

$70 = 10 \times 7$

أي يكرر كتابة $7 \times i$ حيث المتغيرة i تبدأ بالقيمة 1 وتتزايد للتوقف عند القيمة 10.

مثال: إذا أردنا حساب معدل أربع طلبة نكتب كما يلي:

```
Algorithme moyenne
 m,n1,n2 :reel
Debut
 /*1er etudiant*/
Lire(n1,n2)
m←(n1+n2)/2
ecrire(m)
/*2eme etudiant*/
Lire(n1,n2)
m←(n1+n2)/2
ecrire(m)
/*3eme etudiant*/
Lire(n1,n2)
m←(n1+n2)/2
ecrire(m)
/*2eme etudiant*/
Lire(n1,n2)
m←(n1+n2)/2
ecrire(m)
Fin
```

نلاحظ أنه إذا أردنا حساب معدل 100 طالب بهذه الطريقة سيكون من المتعب إعادة كتابها و هنا نلجأ إلى الحلقات.

هناك ثلاث طرق أساسية لإنشاء الحلقات، الحلقة Pour، الحلقة Tant que و الحلقة Repeter تحتوي كلها على المراحل التالية:

1) التهيئة Initialisation
2) اختبار Test
3) الإضافة Incrémentation
4) العودة Retour

1. تعليمة تكرارية بعداد ثابت : الحلقة **Pour**

العداد هو متغيرة تستخدم في الحلقة مع تنفيذ التكرار.

في هذه الحلقة عدد التكرار يكون مُعَرَفْ مسبقا.

طريقة كتابة الحلقة :

```
Pour <compteur>← <expression1> à <expression2>
        العداد          العبارة1          العبارة2
Faire
        <groupe d'actions>   جسم الحلقة
Finfaire
```

حيث أن:

- compteur : هو اسم متغيرة يعتبر العداد.
- Expression1 : هي أول قيمة يأخذها العداد (قيمة بدائية) (valeur initiale).
- Expression2 : هي أخر قيمة يأخذها العداد (قيمة نهائية) (valeur limite).
- التعليمات الموجودة بين faire و finfaire تنفذ n مرة. حيث أن :

n = expression2 - expression1 + 1

ملاحظة:

compteur، expression1 و expression2 يجب أن تكون أعداد صحيحة (طبيعية) Entier.

المبدأ: تنفذ التعليمة Pour كما يلي:

- توضع قيمة Expression1 في Compteur.
- تقارن بين compteur و expression2:
  - إذا كان compteur أكبر من قيمة expression2، تنتهي تعليمة Pour،
  - إذا كان compteur أصغر أو يساوي قيمة expression2، إذا :
    - تنفذ تعليمات جسم الحلقة الموجودة بين faire و finfaire.
    - إضافة 1 إلى compteur (incrémentation).
    - الرجوع إلى المرحلة 2.

مثال:

أكتب خوارزمية تسمح بحساب المجموع من 1 الى 20 (s=1+2+…+20).

```
Algorithme somme
 n, I, s : entier
Debut
 Ecrire(''Donnez le nombre n'')
 Lire(n)
 s←0
 Pour i←1 à n faire
    s←s+i
 Finfaire
 Ecrire(''la somme est :'', s)
 Fin
```

مثال: أكتب خوارزمية تسمح بحساب جمع n عدد طبيعي.

```
Algorithme Nsomme
 n, I, s, x : entier
Debut
 Ecrire(''Donnez le nombre n'')
 Lire(n)
```

```
s←0
Pour i←1 à n faire
  Ecrire(''Donnez un nombre '')
  Lire(x)
  s←s+x
Finfaire
Ecrire(''la somme est :'', s)
Fin
```

2. تعليمة تكرارية بدون عداد : الحلقة **Tant que**

في هذه الحلقة عدد التكرار غير معروف مسبقا، لكن يحدد داخل جسم الحلقة (المعالجة) حسب شرط معين:

طريقة كتابة الحلقة :

```
Initilisation                    قيمة مبدئية
Tant que  <condition >           شرط
faire
  <groupe d'actions>             جسم الحلقة
  <agir sur condition>           تغيير قيمة الشرط
Finfaire
```

حيث أن:

Initialisation : هو تعليمة تسمح بتقديم قيمة مبدئية (للشرط).

Agir sur condition : هي تعليمة تسمح بتغيير قيمة condition و ذلك لإمكان توقيف المعالجة، لكي نتفادى تكرار لا نهائي للحلقة.

ملاحظة :

- في هذه الحلقة عدد التكرار غير معروف على عكس الحلقة Pour.

- في هذه الحلقة ما دام الشرط محقق يستمر البرنامج في تكرار العملية الى أن ينعدم الشرط (يصبح غير محقق).

مثال:

أكتب خوارزمية تسمح بحساب جمع n عدد طبيعي.

```
Algorithme Nsomme1
  n, I, s, x : entier
 Debut
  Lire(n)
  s←0
  i←1
 Tant que i<=n faire
   Lire(x)
   s←s+x
   i←i+1
 Finfaire
 Ecrire(''la somme est :'', s)
 Fin
```

3. تعليمة تكرارية بدون عداد : الحلقة **Répéter**

هذه الحلقة لها نفس دور الحلقة Tant que لكن باتجاه معاكس.

طريقة كتابة الحلقة :

قيمة مبدئية < Initialisation >

Répéter

جسم الحلقة <groupe d'actions>

تغيير قيمة الشرط <agir sur condition>

شرط Jusqu'à <condition >

في هذه الحلقة نكرر تنفيذ التعليمات ما دام الشرط غير محقق (غير صحيح) و تنتهي الحلقة عندما يتحقق الشرط.

ملاحظة :

- في هذه الحلقة أيضا عدد التكرار غير معروف مسبقا.
- في الحلقة Répéter تنفذ مجموعة التعليمات على الأقل مرة واحدة على عكس الطريقة Tant que.

مثال:

أكتب خوارزمية تسمح بحساب جمع n عدد طبيعي.

```
Algorithme Nsomme2
  n, I, s, x : entier
Debut
 Lire(n)
 s←0
 i←1
Répéter
   Lire(x)
   s←s+x
   i←i+1
jusqu'à i>n
Ecrire(''la valeur de la somme est :'', s)
Fin
```

الفصل الخامس: المتغيرات المرقمة

I. مقدمة

أنواع المتغيرات التي درسناها حتى الآن فردية (عادية)، أي أنها تمثل متغير واحد فقط.

في المعلوماتية، يمكننا جمع مجموعة من متغيرات البرنامج في متغيرة واحدة، والتي تم تحديد كل قيمة فيها باستعمال الترقيم (مؤشر).

الجداول هي عبارة عن متغيرات، تستعمل عوض التصريح بمجموعة من المتغيرات لها نفس النوع، أما السلاسل الحرفية فهي في حد ذاتها جداول متكونة من عدة متغيرات من نفس النوع (حروف).

في هذا الفصل سوف نتطرق إلى المتغيرات الرقمية (المركبة) وهي جداول ذات بعد واحد وجداول ذات بعدين تسمى المصفوفات والسلاسل الحرفية.

II. الشعاش أو الجدول ذو بعد واحد

تخيل أنه في البرنامج، تحتاج إلى 12 قيمة (مثال علامات طالب للسداسي لحساب معدله) يمكن استعمال 12 متغيرة وحساب المعدل:

Moy← (N1+N2+N3+N4+N5+N6+N7+N8+N9+N10+N11+N12)/12

البرنامج صحيح لكن إذا احتجنا إلى عدد أكبر من المتغيرات مثلا إذا أردنا حساب متوسط العمر لسكان مدينة ...

من الشائع استخدام الجداول لتخزين البيانات مؤقتا. وهكذا، يمكن أن تكون البيانات بينها علاقة مع 1 أو 2 أو 3 (أو أكثر) المدخلات.

الفائدة من هذا الجدول هي تخزين البيانات في الذاكرة التي يمكن العثور عليها من خلال قيم أخرى باستخدام الحلقات والصيغ الرياضية في الخوارزميات.

يمكن إدخال متغيرات من كل الأنواع التي درسناها في الجداول: صحيح، حقيقي، حرف، سلسلة حرفية، منطقي، ...

الخ

1. تعريف الجدول

الجدول T هو بنية من البيانات (structure de donnée) التي تجمع مجموعة من القيم من نفس النوع تدعى عناصر الجدول T[i] والتي نحددها بالمؤشر i.

عموما لكل الجداول الخصائص التالية:

- جميع العناصر لديها نفس النوع.
- عدد العناصر المخزنة يكون ثابت.
- مؤشر عنصر هو المكان الذي يؤخذه العنصر بالنسبة لبداية الجدول.
- يتم الوصول إلى عنصر من الجدول باستخدام المؤشر.
- استعمال و تغيير العنصر رقم i لا يغير قيمة المؤشر i ولا عدد عناصر الجدول.
- يمكننا اعتبار الجدول كمتغيرة واحدة مخزنة في خانات الذاكرة واستعماله كوسيط لدالة أو نتيجة عملية حسابية.

مثال :

إذا مثلنا أشهر السنة في جدول T:

1	2	3	4	5	6	7	8	9	10	11	12
Janv	fev	mars	avril	mai	juin	juillet	aout	sept	Oct	Nov	Dec

فإن شهر جانفي يكون في المؤشر1 : T[1] و ماي في المؤشر 5 : T[5].

2. تصريح الجداول

يجب التصريح بالجدول ككل المتغيرات في قسم الإعلانات. فيجب تحديد نوع المتغيرات التي سيتم إدخالها في الجدول.

طريقة الكتابة:

<nom_du_tableau> : tableau[<MinDim>..<MaxDim>] de < type>

نوع القيم أكبر قيمة أدنى قيمة الاسم التعريفي للجدول

حيث أن:

nom_du_tableau: هو الاسم التعريفي للمتغيرة المرقمة جدول.

MinDim : أصغر قيمة المجال الذي تؤخذه المؤشرات.

MaxDim : أكبر قيمة المجال الذي تؤخذه المؤشرات.

Type : نوع عناصر الجدول.

ملاحظة :

MinDim و MaxDim يجب أن تكون من النوع الصحيح Entier أو حرف Caractère أو منطقي Booléen.

Tableau و de : هي كلمات مخصصة.

أمثلة:

Tab1 : Tableau[1..100] d'entier ;

Tab2: Tableau [-10..10] de reel ;

Tab3 : Tableau [50..60] de chaine de caractere ;

Tab4 : Tableau ['A'..'S'] d'entier ;

Tab5 : Tableau [vrai..Faux] de reel {ne contient que deux valeurs}

3. العمليات على الجداول

1) الوصول الى عنصر من الجدول :

T[2] يمكننا من الوصول الى العنصر 2 من الجدول.

2) القراءة والكتابة في الجدول:

- قراءة العنصر2 : Lire(T[2])
- قراءة العنصر i : Lire(T[i])
- كتابة العنصر2 : Ecrire(T[2])
- كتابة العنصر i : Lire(T[i])
- قراءة جميع عناصر الجدول : Pour i←1 à n faire lire(T[i]) finfaire
- كتابة جميع عناصر الجدول : Pour i←1 à n faire Ecrire (T[i]) finfaire

3) التخصيص

تخصيص قيمة للعنصر الثاني T[2]←valeur

4) العمليات الحسابية

T[2]←sqrt(valeur^3+2)

5) عمليات المقارنة

Si T[2]>=valeur

<u>مثال:</u>

أكتب خوارزمية تسمح بحساب معدلات 100 طالب في مجموعة، لكل طالب علامة التقييم المستمر و علامة امتحان السداسي ثم احسب معدل المجموعة.

```
Algorithme moyenne
  i : entier
  cc, s, moy : tableau [1..100] de reel
  secmoy :reel
Debut
```

```
Ecrire(''donner les notes cc et s pour chaque étudiant'')
secmoy ←0
Pour i←1 à n faire
  Lire(cc[i],s[i])
  Moy[i] ←( cc[i]+s[i])/2
  Ecrire ("la moyenne de l'etudiant", i, " est: ", moy[i])
  secmoy ← secmoy + moy[i]
Finfaire
secmoy ← secmoy/100
Ecrire(''la moyenne de la section est :'', secmoy)
Fin
```

4. بعض الخوارزميات الأساسية حول الجداول

سوف نتطرق لبعض الخوارزميات الكلاسيكية التي يجب معرفتها لتسهيل كتابة خوارزميات جديدة بالجداول.

أ- خوارزميات البحث

خوارزميات البحث تسمح بالبحث عن قيمة معينة في جدول مرتب أو غير مرتب.

1.1- البحث عن قيمة في جدول غير مرتب Recherche d'une valeur dans un tableau non trié

الخوارزمية الأولى بدون استعمال متغيرة منطقية و الخوارزمية الثانية باستعمال متغيرة منطقية.

```
Algorithme Recherche1
  n, i, val : entier
  tab: tableau [1..100] de entier
Debut
/* on suppose que n est inferieur a 100 */
 Ecrire(''donner la valeur de n '')
```

```
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire   Lire(tab[i])
Finfaire
Ecrire(''donner la valeur que vous cherchez '')
Lire(val)
i←1
Tant que (i<=n) ET (tab[i]<>val) faire
i ← i+1
Finfaire
Si i>n alors Ecrire(''la valeur n'existe pas dans le tableau'')
        Sinon Ecrire(''la valeur existe à la position'', i ,''  dans le tableau'')
Fsi
Fin

Algorithme Recherche2
  n, i, val : entier
  bool : booleen
  tab: tableau [1..100] de entier
Debut
/* on suppose que n est inferieur a 100 */
 Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire
Lire(tab[i])
```

```
Finfaire
Ecrire(''donner la valeur que vous cherchez '')
Lire(val)
i←1
bool←faux
Tant que (i<=n) ET (bool=faux) faire
  Si tab[i]=val alors bool←vrai
                Sinon i ← i+1
  Fsi
Finfaire
Si bool=vrai alors Ecrire(''la valeur'', val, '' existe à la position'', i ,''  dans le tableau'')
      Sinon Ecrire (''la valeur'', val, '' n'existe pas dans le tableau'')
Fsi
Fin
```

1.2- البحث عن قيمة في جدول مرتب Recherche d'une valeur dans un tableau trié

1.2.1- بحث متسلسل Recherche sequentielle

```
Algorithme Recherche3
  n, i, val : entier
  tab: tableau [1..100] de entier
Debut
/* on suppose que n est inferieur a 100 */
 Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire
Lire(tab[i])
```

```
Finfaire
Ecrire(''donner la valeur que vous cherchez '')
Lire(val)
Si val<tab[1] OU val>tab[n]
alors Ecrire(''la valeur n'existe pas dans le tableau'')
        Sinon
        i←1
        Tant que (tab[i]<val) faire
        i ← i+1
        Finfaire
        Si val=tab[i]
        alors  Ecrire(''la valeur existe à la position'', i ,''  dans le tableau'')
        Sinon Ecrire(''la valeur n'existe pas dans le tableau'')
        Fsi
Fsi
FIN
```

1.2.2- بحث التفرع الثنائي Recherche dichotomique

```
Algorithme Recherche4
  n, i, val, i1, i2, m : entier
  bool : booleen
  tab: tableau [1..100] de entier
Debut
Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
```

```
Pour i←1 à n faire   Lire(tab[i])    Finfaire
Ecrire(''donner la valeur que vous cherchez '')
Lire(val)
Si val<tab[1] OU val>tab[n]
alors Ecrire(''la valeur n'existe pas dans le tableau'')
      Sinon
       i1←1
       i2←n
       bool←faux
      Tant que (i1<=i2) ET (b=faux) faire
       m ←Floor((i1+i2)/2)
       si tab[m]=val alors bool←vrai
                    sinon si val <tab[m] alors   i2←m-1
                                                 sinon  i1←m+1
                          Fsi
       Fsi
      Finfaire
      Si bool=vrai
       alors  Ecrire(''la valeur existe à la position'', m ,''  dans le tableau'')
       Sinon Ecrire(''la valeur n'existe pas dans le tableau'')
      Fsi
Fsi
FIN
```

ب- خوارزميات تحديث جدول مرتب

خوارزميات التحديث تسمح بالتغيير الجدول سوآءا بإضافة خانة جديدة في الجدول أو حذف خانة منه.

3.1- ادراج قيمة Insertion d'une valeur

```
Algorithme insertion_valeur
  n, i, j, val: entier
  tab: tableau [1..100] de entier
Debut
Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire
Lire(tab[i])
Finfaire
Ecrire(''donnez la valeur que vous voulez insérer '')
Lire(val)
i←1
Tant que (i<=n) et  (val>tab[i]) faire i←i+1 finfaire
j←n+1
Tant que (j>i) faire
 tab[j] ← tab[j-1]
 j←j-1
finfaire
tab[i] ← val
n←n+1
Ecrire(''Le nouveau tableau est :'')
Pour i←1 à n faire   Ecrire(tab[i])   Finfaire
```

FIN

3.2- <u>Supression d'une valeur حذف قيمة</u>

```
Algorithme suppression_valeur
  n, i, j, pos: entier
  tab: tableau [1..100] de entier
Debut
Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire   Lire(tab[i])   Finfaire
Ecrire(''donnez la position de la valeur que vous voulez supprimer '')
Lire(pos)
i←pos+1
Tant que (i<=n) faire
tab[i-1] ← tab[i]
i←i+1
finfaire
n←n-1
Ecrire(''Le nouveau tableau est :'')
Pour i←1 à n faire   Ecrire(tab[i])   Finfaire
FIN
```

أ. بن يحي ثاني نسرين

ت- خوارزميات الترتيب

خوارزميات الترتيب تسمح بالترتيب جدول ترتيبا تصاعديا.

2.1- الترتيب عن طريق الاختيار Tri par selection

```
Algorithme tri_selection
  n, i, j, x, min : entier
  tab: tableau [1..100] de entier
Debut
Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire   Lire(tab[i])  Finfaire
Pour i←1 à n-1 faire
 Min← i
 Pour j←i+1 à n faire
 Si tab[j]<tab[min] alors min←j
 Fsi
 Finfaire
 Si i<>min alors x←tab[i]
                tab[i] ←tab[min]
                tab[min] ←x
 Fsi
Finfaire
Ecrire(''Le tableau trié est :'')
Pour i←1 à n faire    Ecrire(tab[i])   Finfaire
FIN
```

2.2- الترتيب عن طريق التبديل Tri par permutation Bulle

```
Algorithme tri_Bull
  n, i, j, x: entier
  tab: tableau [1..100] de entier
Debut
Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire
Lire(tab[i])
Finfaire
Pour i←1 à n-1 faire
 Pour j←1 à n-i faire
 Si tab[j]>tab[j+1] alors x←tab[j]
                        tab[j] ←tab[j+1]
                        tab[j+1] ←x
 Fsi
 Finfaire
Finfaire
Ecrire(''Le tableau trié est :'')
Pour i←1 à n faire
Ecrire(tab[i])
Finfaire
FIN
```

2.3- <u>الترتيب عن طريق الادراج Tri par insertion</u>

```
Algorithme tri_insertion
  n, i, j, x, pos: entier
  tab: tableau [1..100] de entier
Debut
Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire
Lire(tab[i])
Finfaire
Pour i←2 à n faire
pos←1
 Tant que (tab[pos] <tab[i]) faire pos←pos+1 finfaire
 Si (pos<>i) alors x←tab[i]
                    j ←i
                    Tant que (j>=pos+1) faire
                      tab[j] ← tab[j-1]
                      j ← j-1
                    finfaire
                   tab[pos] ←x
 Fsi
 Finfaire
Ecrire(''Le tableau trié est :'')
Pour i←1 à n faire  Ecrire(tab[i])  Finfaire
FIN
```

تمرين: أكتب خوارزمية تسمح بحساب عدد تكرار قيمة معينة في جدول.

```
Algorithme occurence
  n, i, occ : entier
  val : reel
  tab: tableau [1..100] de reel
Debut
Ecrire(''donner la valeur de n '')
Lire(n)
Ecrire(''donner les valeur du tableau '')
Pour i←1 à n faire    Lire(tab[i])   Finfaire
Ecrire(''donner la valeur que vous cherchez '')
Lire(val)
occ←0
pour i←1 à n faire
si tab[i]=val alors occ←occ+1   FSI
Finfaire
Si occ=0 alors Ecrire(''la valeur n'existe pas dans le tableau'')
        Sinon Ecrire(''le nombre d'occurrence de la valeur est :'', occ)
Fsi
Fin
```

III. المصفوفة أو الجدول ذو بعدين

نوع الجداول التي درسناها حتى الأن ليس لها سوى مؤشر واحد فقط، لذا نسميها أحادية البعد. لكن كل عنصر من عناصر هذه الجداول يمكنه أن يكون جدول بحد ذاته، فنسميه جدول ثنائي الأبعاد أو مصفوفة.

1. تعريف المصفوفة

الجدول الثنائي الأبعاد هو متغير يتكون من عدة عناصر مركبة لها نفس النوع.

العنصر في جدول ثنائي الأبعاد هو بحد ذاته يعتبر شعاع. منه تمثيل الجدول ثنائي الأبعاد يكون بمصفوفة أي مجموعة من الأسطر والأعمدة.

الإشارة الى عنصر تكون باستعمال اسم المصفوفة و مؤشرين : المؤشر الأول هو رقم السطر و المؤشر الثاني هو رقم العمود، أي مكانه بالنسبة للأسطر و الأعمدة.

العنصر M[i,j] يوجد في التقاطع بين السطر i و العمود j.

2. تصريح المصفوفات

يتم التصريح بالمصفوفة في قسم الإعلانات. فيجب تحديد نوع المتغيرات التي سيتم إدخالها فيها.

طريقة الكتابة:

<nom_de_la_matrice> : tableau[<MinDimL>..<MaxDimL> , <MinDimC>..<MaxDimC>] de < type>

الاسم التعريفي للمصفوفة أدنى قيمة للأسطر أكبر قيمة للأسطر أدنى قيمة للأعمدة أكبر قيمة للأعمدة نوع القيم

حيث أن:

nom_de_la_matrice: هو الاسم التعريفي للمتغيرة المرقمة مصفوفة.

MinDimL : أصغر قيمة المجال الذي تؤخذها مؤشرات الأسطر.

MaxDimL : أكبر قيمة المجال الذي تؤخذها المؤشرات الأسطر.

MinDimC : أصغر قيمة المجال الذي تؤخذها مؤشرات الأعمدة.

MaxDimC : أكبر قيمة المجال الذي تؤخذها المؤشرات الأعمدة.

Type : نوع عناصر الجدول.

<u>مثال:</u>

MAT1 : Tableau[1..10, 1..20] de entier

3. العمليات على المصفوفات

1) الوصول الى عنصر من المصفوفة :

MAT[1,2] يمكننا من الوصول الى العنصر الموجود في السطر الأول العمود الثاني من المصفوفة.

MAT[2,1] يمكننا من الوصول الى العنصر الموجود في السطر الثاني العمود الأول من المصفوفة.

2) القراءة والكتابة في المصفوفة:

- قراءة عنصر : Lire(MAT[1,2])
- قراءة جميع عناصر السطر الأول من المصفوفة تتكون من C عمود :

```
Pour i←1 à C faire lire(MAT[1,i]) finfaire
```

- قراءة جميع عناصر العمود الثاني من المصفوفة تتكون من L سطر :

```
Pour i←1 à L faire lire(MAT[i,2]) finfaire
```

- قراءة جميع عناصر المصفوفة:

```
Pour i←L à n faire
|Pour j←C à n faire
| lire(MAT[i,j])
|finfaire
finfaire
```

- كتابة عنصر : Ecrire(MAT[1,2])
- كتابة جميع عناصر السطر الأول من المصفوفة تتكون من C عمود :

```
Pour i←1 à C faire Ecrire (MAT[1,i]) finfaire
```

- كتابة جميع عناصر العمود الثاني من المصفوفة تتكون من L سطر :

```
Pour i←1 à L faire Ecrire (MAT[i,2]) finfaire
```

- كتابة جميع عناصر المصفوفة:

```
Pour i←L à n faire
|Pour j←C à n faire
| Ecrire (MAT[i,j])
|
```

finfaire
finfaire

3) التخصيص

تخصيص قيمة لعنصر MAT[1,2]←valeur

4) العمليات الحسابية

MAT[1,2]←sqrt(valeur^3+2)

5) عمليات المقارنة

Si MAT[1,2]>=valeur

تمرين: أكتب خوارزمية تسمح بترتيب عناصر مصفوفة ترتيبا تنازليا.

```
Algorithme Matri_Bull
  n, i, j, k: entier
  x : reel
  mat: tableau [1..50,1..50] de reel
Debut
Ecrire(''donner la valeur de n et m '')
Lire(n,m)
Ecrire(''donner les valeur de la matrice '')
Pour i←1 à n faire Pour j←1 à m faire Lire(mat[i,j]) Finfaire  Finfaire


Pour k←1 à n*m-1 faire
Pour i←1 à n faire
Pour j←1 à m-1 faire
Si mat[i,j]<mat[i,j+1] alors x←mat[i,j]
                        mat[i,j] ← mat[i,j+1]
```

```
                    mat[i,j+1] ←x
 Fsi
 Finfaire
Si i<n et mat[i,m]<mat[i+1,1] alors x←mat[i,m]
                                    mat[i,m] ← mat[i+1,1]
                                    mat[i+1,1] ←x
 Fsi
 Finfaire
 Finfaire
 Ecrire(''La matrice triée est :'')
Pour i←1 à n faire
 Pour j←1 à m faire
  Ecrire(mat[i,j])
 Finfaire
Finfaire
 FIN
```

أبن يحي ثاني نسرين

الخاتمة

تطرقنا من خلال ما قدمناه في الفصول السابقة الى أساسيات الخوارزميات المبرمجة لطلبة السنة الثانية علوم التسيير علوم تجارية و علوم اقتصادية لنظام ل.م.د، و الهدف منه جعل الطالب قادر على التعامل مع الخوارزميات أي كتابة و قراءة الخوارزميات.

ليس المطلوب من الطالب في هذا التخصص أن يكون مبرمجا لأنه سوف يكون في معظم الأحيان مستخدم، ولكن هذا التعليم سيمكنه من التعامل بسهولة أكبر مع البرمجيات التي سوف يستخدمها.

هذه المطبوعة مقسمة الى خمسة فصول وهي تعبر عن الثلاث المراحل الأساسية لكتابة أي خوارزمية في علم البرمجيات.

الفصل الأول يسهل الفهم الشامل للتمرين.

الفصل الثاني يسمح لنا بتوضيح المتغيرات والثوابت أي المدخلات والمخرجات لأي تمرين.

الفصول الموالية هي عبارة عن الأدوات التي نستعملها لحل التمرين، فقد تطرقنا من خلالها لآليات كل هذه الأدوات طريقة كتابتها ومبادئها.

كما أن كل مفهوم جديد مرفق بتمرين توضيحي.

أ. بن يحي ثاني نسرين

BIBLIOGRAPHIE **المراجع**

- Mc Belaid, Algorithmique en langage C++, Pages bleues, Algérie, 2003.
- Mc Belaid, Structures de données Algorithmes et Programmes en Pascal, Pages bleues, Algérie, 2004.
- Mc Belaid, Algorithmique et programmation en pascal, Pages bleues, Algérie, 2008.
- Mc Belaid, Initiation à l'Algorithmique, Pages bleues, Algérie, 2012.
- P. Berlioux, Ph. Bizard, Algorithmique, Dunod, Paris, 1983.
- T.H. Cormen, C.E.Leiserson, R.L.Rivest, C.Stein, Introduction to Algorithms, MIT Press, 2001.
- J. Courtin, L. Kowarski, Initiation à l'algorithmique et aux structures de données, Dunod, Paris, 1994.
- C. Delannoy, Programmer en langage C++, Chihab-Eyrolles, 1995.
- F.Daoudi, Introduction à l'algorithmique, l'abeille, Algérie, 2008.
- S.Graine, Pascal en 12 leçons faciles, l'abeille, Algérie, 2004.
- C.Ighilaza,H.Messaoud et S. Aouat, Initiation à l'Algorithmique, OPU, Algérie, 2009.
- C. Khichane, Le leader de l'Algorithmique, Edition El Maarifa, Algérie, 2004.
- M.Koudil, S.L.Khelifati, Structure des Ordinateurs, OPU, Algerie, 2005.
- J.M Léry, L'intro développement le langage C, Campus Press, 2002.
- P.Lignelet, Algorithmique Méthodes et Modèles – T1 notions de base, Masson Edition, Paris, 1985.
- J. Lonchamp, Les langages de programmation, Masson, Paris, 1989.
- B. Meyer, Introduction à la théorie des langages de programmation,

InterEdition, Paris, 1992.

- D.Mokhtari, Le Pascal facile, Edition Hamdi, Algérie, 1999.
- D.Rebaine, une introduction à l'analyse des algorithmes, ENAG Edition, Algérie, 2000.
- N.SALMI, Principes des Systèmes d'exploitation, Edition Pages Blues, Algérie, 2007.
- H.Thomas, E. Charles, Introduction to Algorithms, McGraw Hill, New York, 2001
- Wesley, The art of computer programming, volume 3 Sorting & Searching. 1998
- ي. بوكليخة، الخوارزمية و لغة باسكال، ديوان المطبوعات الجامعية، 2009.
- ك.كارلوف، البرمجة بلغة باسكال، دار هومة، الجزائر ،2000.
- ع. هامل، دروس للمبتدئين في الإعلام الألي، نوميديا للطباعة و النشر و التوزيع، 2007.

أ. بن يحي ثاني نسرين

جدول أسكي

ن. بن يحي ثاني نسرين

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL