TP 1 : Premier Pas en Pascal

Semaine du 01 Novembre 2008

1- Syntaxe d'un programme Pascal

Progam *Nom du Programme ;*
Déclaration des constantes, des variables, des fonctions ...
Begin
Instructions **;**
...
Instructions **;**
End.

Remarque. Il faut mettre **un point** à chaque fin de programme et **un point virgule** à chaque fin d' instruction.

Exercice 1 : Saisir le programme suivant :

```
Begin
Write('Bonjour Msila') ;
End.
```

Sauvegarder ce programme (cliquer sur le menu File puis le sous-menu Save). Sous le nom tp1.pas

2-Déclaration de variables

Il faut pour pouvoir utiliser des variables dans un programme déclarer son type en début de programme.

Syntaxe.

Var *nom de la variable : type ;*

Pour déclarer qu'une variable i *est de type entière, on ´écrit* **Var** i : integer ;
Pour déclarer qu'une variable nom *est de type chaîne de caractères, on écrit* **Var** nom : string
Pour déclarer qu'une variable x est de type réel, on écrit **Var** x : real

3-Entrées et sorties à l'écran (read / write)

Write :

La commande write permet d'afficher du texte à l'écran et laisse le curseur à la fin du texte affiché.
Syntaxe.
write*('Texte à afficher', variable1, variable2, 'texte2')*1 *;*
Pour cela, utiliser la commande writeln qui est semblable à write mais qui renvoie le curseur à la ligne après l'affichage.

read :

La commande read permet de donner une valeur (par le clavier) à une variable.

Syntaxe.

read*(variable) ;*

La commande readln est semblable à read mais permet de renvoyer le curseur à la ligne.

Exercice 2. *Ecrire le programme suivant :*

```
Program Name ;
Var nom : String ;
begin
write('Entrer ton nom') ;
readln(nom) ;
writeln('Ton nom est ', nom);
readln;
end.
```

Changer ce programme afin de demander `a l'utilisateur son nom et son prénom et de les afficher.
Pour pouvoir arrêter le défilement, ajoutez l'instruction readln ; *avant la fin du programme.*

Exercice 3 : Ecrire un programme qui donne le code ASCII d'un caractère.

TP 2 : les instructions Conditionnelles

Semaine du 08 Novembre 2008

INSTRUCTION IF - THEN - ELSE (si - alors - sinon)

Structure: **IF** condition **THEN** instruction1 (CAS 1)

IF condition **THEN** instruction1 **ELSE** instruction2 (CAS 2)

Si la condition est vraie, alors on exécute l'instruction1 (simple ou composée). Sinon, on passe à la suite (cas 1), ou on exécute l'instruction2 (cas 2).

Remarque :

- il n'y a pas de ; devant le ELSE.
- L'instruction2 peut être composée ou entre autres être une instruction IF :

```
IF condition1 THEN instruction1
 ELSE IF condition2 THEN instruction2
            ELSE IF condition3 THEN instruction3
.....
             ELSE instructionN
```

- Un ELSE correspond toujours au dernier IF rencontré.

```
ex: IF cond1 then if cond2 then inst1 {cond1 et cond2}
        else inst2 {cond1 et pas cond2}
else if cond3 then inst3 {pas cond1 mais cond3}
        else inst4 {ni cond1 ni cond3}
```

- Si on désire autre chose, utiliser BEGIN et END :

```
IF cond1 THEN    begin
                 if cond2 then inst1
            end   (* le prochain ELSE se rapporte à COND1 *)
ELSE inst2
```

- En cas de plusieurs instructions dans le bloc if ou else utiliser le Begin et le End afin de délimiter le bloc.

Exercice 1: exécuter le programme suivant :

```
PROGRAM valeur_absolue ;
VAR x : real ;
BEGIN
WriteLn('Entrez la valeur de x.') ;
ReadLn(x) ;
IF x < 0 THEN WriteLn(-x) ELSE WriteLn(x) ;
END.
```

- modifier ce programme pour afficher la nature de X(positive ou négative) dans une ligne et sa valeur absolue dans une autre ligne.

Exercice2: Écrire un programme qui demande à l'utilisateur une valeur réelle, et affiche la valeur de f(x), où $f(x) = 2x$ si $x \leq 3$ et $f(x) = x - 3$ si $x > 3$.

TP 3 : Structure de sélection

Semaine du 15-21 Novembre 200

Syntaxe générale de l'instruction case of :

Elle peut prendre une de ces deux formes :

```
case variable of
            domaine_1: instruction_1;
            domaine_2: instruction_2;
            ...
            domaine_n: instruction_n
end

case variable of
            domaine_1: instruction_1;
            domaine_2: instruction_2;
            ...
            domaine_n: instruction_n;
      else instruction_bis
end
```

Exercice :

1- Exécuter manuellement les programmes suivants, puis donner les résultats de chaque programme pour les valeurs : 1,2,3,4,5,8,9 .

2- compléter et exécuter automatiquement chaque programme avec les valeurs précédentes, puis comparer votre réponse avec les résultats affichés par chaque programme .

	Programme A		Programme B		Programme C		Programme D	
	Votre réponse	Résultat affiché	Votre réponse	Résultat affiché	Votre réponse	Résultat affiché	Votre réponse	Résultat affiché
1								
2								
3								
4								
5								
8								
9								

a)
```
Case a of
    1 : writeln (' *') ;
    2 :  writeln (' +') ;
3,4,5 :  writeln (' -') ;
    8 :  writeln (' /')
End ;
Writeln ( 'OK') ;
```

b)
```
Case a of
    1 : writeln (' *') ;
    2 :  writeln (' +') ;
 3..5 :  writeln (' -') ;
    8 :  writeln (' /')
End ;
Writeln ( 'OK') ;
```

C)
```
Case a of
    1,4 : begin
     writeln (' (') ;
    writeln (' )')
    end ;
    6 :  writeln (' +') ;
    9 :  begin
     writeln (' (') ;
    writeln (' .') ;
    writeln (' )')
    end ;
    End ;
    Writeln ( 'OK') ;
```

d)
```
Case a of
      1 : writeln (' *') ;
    2,7 :  writeln (' -') ;
      5 :  writeln (' +')
Else
        writeln (' /') ;
        writeln (' $') ;
end;

Writeln ( 'OK') ;
```

TP 4 : la structure répétitive
Semaine du 22-26 Novembre 2008

Les boucles :

Rappel de syntaxe :
En Pascal les boucles tant que , pour , répéter ont la forme suivante :

While *condition* **do**
Begin
Instruction 1 ;
Instruction 2 ;
Instruction n ;
End ;
	For i:= val min **to** val max **do**
	Begin
	Instruction 1 ;
	Instruction 2 ;
	Instruction n ;
	End ;
		Repeat
		Instruction 1 ;
		Instruction 2 ;
		Instruction n ;
		Until *condition*

Remarque :

1- On peut utiliser un pas dégressif en remplaçant TO par DOWNTO.

ex: for lettre:='Z' downto 'A' do writeln(lettre).

2- On peut supprimer le Begin et le End d'un bloc en cas d'une seule instruction dans ce dernier.

Exercice 1 :

```
PROGRAM debut ;
VAR
x :integer;
BEGIN
x :=0 ;
WHILE x <= 5 DO
BEGIN
x :=x+1 ;
WRITELN(x) ;
END;
END.
```

1. Que fait le programme ? donner le résultat d'affichage manuellement.
2. Déplacer writeln (x) à l'extérieur de la boucle while . donner le résultat d'affichage manuellement.
3. Remplacer la boucle WHILE par la boucle REPEAT. et for

Exercice 2 :

- **Quel** type de boucles pouvez-vous utiliser **dans** un programme nommé 'Encore' qui demande à l'utilisateur « encore ? » , et qui continue de lui poser la question tant que celui-ci lui répond « oui ».
- Ecrivez les programmes correspondants à chaque type de boucle puis exécutez.

Exercice 3 :
Ecrivez un programme qui demande un entiers N , puis affiche la valeur : N^N en utilisant les trois types de boucles.

TP 5 : Types Structurés définis par l'utilisateur **Informatique Fondamentale 1**
Semaine du 29Novembre - 6 décembre 2008

Les tableaux :

Syntaxe de la déclaration d'un type tableau en Pascal:

```
type nom_du_tableau = array[valmin ..valmax] of type_éléments ;
exemple : tableau= array[1 ..10] of integer ;
```

Syntaxe de la déclaration d'une variable de type tableau :
Vous avez deux choix :

```
Var
x : tableau ;
```

Ou bien

```
var
X : array[1 ..10] of integer ;
```

Exercice 1 :
Soit le programme suivant :

```
Program tableau ;
Const n= 4
Type   t1= array [ 1..n] of integer ;
Var    tab:t1;
       I: integer;
Begin
For i:=1 to N do
Readln (tab[i]);
For i:= 1 to N do
writeln (tab[i]);
end.
```

1- Quelle est la fonction de ce programme?
2- Remplacez l' intervalle de i par :[-3..0] ,['a'..'d'], puis ré exécutez le programme.

Exercice 2 :
Écrire un programme qui affiche la plus grande et la plus petite valeur stockée dans un tableau de 10 entiers non ordonnés.

TP 6 : **Types Structurés définis par l'utilisateur** **Informatique Fondamentale 1**
Semaine du 17 au 21 Janvier 2009

Les enregistrements, les chaînes de caractères :

Syntaxe de la déclaration d'un type chaine de caractères en Pascal:

Une chaîne de caractères se déclare au moyen du mot réservé **string**.

Exemple : type t_nom = **string** [30];

Syntaxe de la déclaration d'un type enregistrement en Pascal:

Un enregistrement se déclare au moyen du mot réservé **record**.

Exemple : type etudiant = **record**
Num : **integer ;**
Nom **: t_nom**;
Prenom : **t_nom**

End ;

Exercice :

Soit un tableau annuaire téléphonique de 10 entrées, chaque entrée possède deux champs : Nom et Num-portable.

- Quel est le type du champ ; Num-portable ? pourquoi

Ecrire un programme permettant la saisie des entrées de l'annuaire, puis l'affichage de l'annuaire et enfin la recherche de numéro de portable d'un nom donné.

TP 7 : **Informatique Fondamentale 1**
Semaine du 17 au 21 Janvier 2009

Les procédures et les fonctions :

On appelle "palindrome" un mot ou une phrase qui se lit de la même façon à l'endroit comme à l'envers, sans tenir compte des espaces.

Exemple : le mot "ABCBA" est un palindrome.

La phrase "ESOPE RESTE ET SE REPOSE" (sans tenir compte des espaces on obtient le mot "ESOPERESTEETSEREPOSE") se lit de façon identique de la gauche vers la droite ou de la droite vers la gauche.

Ecrire une fonction récursive qui teste si une chaine donnée est palindrome ou non.
Ecrire un programme appelant de cette fonction.

TP 8: Les sous programmes **Informatique Fondamentale 1**
Semaine du 17 au 21 Janvier 2009

Passage de paramètres :

écrire un programme qui appelle une procédure de parmutation circulaire de trois entiers : a,b,c .
Tester les deux modes de passage de paramètres

- Par copie
- Par variable

TP 9: La récursivité

La récursivité

Nous allons réaliser maintenant un programme permettant d'évaluer un nombre romain, mais auparavant on va introduire les chiffres romains:

- M = 1000
- D = 500
- C = 100
- L = 50
- X = 10
- V = 5
- I = 1

On constate que les nombres s'arrêtaient au milliers; cela montre le progrès des mathématiques depuis le temps des romains.

L'écriture des nombres romains

- 1 s'écrit I.
- 2 s'écrit II.
- 3 s'écrit III.
- 4 s'écrit IV.
- 5 s'écrit V.
- 6 s'écrit VI.
- 7 s'écrit VII.
- 8 s'écrit VIII.
- 9 s'écrit IX.
- 10 s'écrit X.

TP 10: **Types Structurés définis par l'utilisateur** Semaine du 03 au 7 Avril 2009

Les fichiers :

Rappel : les commandes principales sur les fichiers en pascal sont :

Commande	Signification
assign (nom logique, nom physique).	Affecter le nom logique à un emplacement physique
rewrite (nom *logique*) :	Pour ouvrir le fichier en ecriture
Reset (nom logique)	Pour ouvrir le fichier en lecture
read (nom logique , variable) :	mettre un élément de fichier dans la variable et avancer
write (nom logique , variable) :	Mettre la variable à la fin de fichier
EOF (nom *logique*) :	Tester l'accés à la fin de fichier true ou false
close*(nom logique)*	Fermer le fichier
filepos (nom logique)	position actuelle du repère de fichier
filesize (nom logique)	taille du fichier en enregistrements
erase (nom logique)	efface le fichier File
rename (nom logique, < nouveau_nom >);	renomme le fichier
seek (nom logique, < Num >)	positionne le repère sur l'enregistrement de numéro Num

Exercice :

Soit f un fichier des étudiant dont les enregistrements ont la structure suivante:

```
record
num-insc : integer
nom, prenom: string;
note: integer;
end;
```

1- Ecrire une procédure permettant de remplir ce fichier à partir du clavier.
2- Ecrire une procédure permettant d'afficher ce fichier
3- Ecrire une procédure permettant d'accéder à un élément dont son numéro d'enregistrement est donné par l'utilisateur
4- Ecrire une procédure permettant d'afficher l'étudiant dont son num d'inscription est donné par l'utilisateur.
5- Ecrire une procédure permettant de modifié l'élément trouvé par la procédure précédente.
6- Ecrire une procédure permettant de supprimer un étudiant dont le num d'inscription est donné par l'utilisateur.
7- Ecrire une procédure permettant de trier le fichier par note
8- Ecrire le programme appelant qui propose à l'utilisateur le menu suivant :

Saisie1 affichage2 accès direct3 recherche 4 modification5 suppression6 tri7

Et demande à l'utilisateur d'entrer un numéro de 1 à 7 pour réaliser la procédure correspondante.

TP 1 1: Types Structurés définis par l'utilisateur Semaine du 02 au 7 Mai 2009

Les fichiers text :

Rappel : les commandes principales sur les fichiers text en pascal sont :

Commande	Signification
assign (nom logique, nom physique).	Affecter le nom logique à un emplacement physique
rewrite (nom *logique*) :	Pour ouvrir le fichier en ecriture
Reset (nom logique)	Pour ouvrir le fichier en lecture
read (nom logique , variable) :	mettre un élément (ligne ou caractere) de fichier dans la variable .
Readln(nom logique)	Retour à la ligne en mode lecture
write (nom logique , variable) :	Mettre la variable à la fin de fichier
Writeln(nom ligique)	Mettre une marque de fin de ligne à la fin de la ligne courante et passer à la ligne suivante
EOF (nom *logique*) :	Tester l'accés à la fin de fichier true ou false
EOLN(f)	Tester l'accés à la fin de ligne true ou false
close*(nom logique)*	Fermer le fichier

Exercice :

Définir une procédure permettant de crypter un texte saisie par l'utilisateur, puis le sauvegarde sous forme crypté.
L'algorithme de cryptage doit remplacer chaque caractère par son successeur- a→ b ; b→c …z→ a
Définir une procédure de décryptage permettant d'afficher le fichier texte sous forme décrypté soit sur l'écran soit sur un fichier texte ;

TP 12: les listes linéaires chaînées Semaine du 09 - 14 Mai 2009

Exercice1 :

Soit le type liste suivant :

```
type
liste = ^cellule;
cellule = record
info:integer;
suiv:liste;
end;
var
k,p : liste ;
```

1/ Exécutez chaque séquence d'instructions manuellement et notez les résultats dans la colonne M.
2/Reexécutez les séquences automatiquement et notez les résultats affichés dans la colonne A.
3/ comparez les résultats de M et A de chaque séquence, puis justifiez la différence entre les deux .

1/

```
New (p)
P^ .nfo :=10 ;
K := p ;
New ( k) ;
Write (k^.info) ;
Write( p^.info);
```

	M	A
K		
P		

2/

```
New (p)
P ^.info:=10 ;
K := p ;
New ( p) ;
Write (k^.info) ;
Write( p^.info);
```

	M	A
K		
P		

3/New (p) ;

```
P ^.info:=10 ;
K := p ;
P.^info:=6;
Write (k^.info) ;
Write( p^.info);
```

	M	A
K		
P		

4/

```
New (p)
P ^.info:=10 ;
K := p ;
New(p);
New(k);
P.^info:=6;
k^.info:= 7;
Write (k^.info) ;
Write( p^.info);
```

	M	A
K		
P		

5/

```
New (p);
New (k);
P^.info:=10;
K^.info:=20;
K:=p;

P^.info:= p.info+k^.info;
Write (k^.info) ;
Write( p^.info);
```

	M	A
K		
P		

6/

```
New (p)
P^.info :=10 ;
K := p ;
New ( p) ;
P.^info:= 6;
Write (k^.info) ;
Write( p^.info);
```

	M	A
K		
P		

Exercice 2 :

Soit l une liste de caractères :

1/ Ecrire une procédure d'initialisation d'une liste.

2/ écrire une procédure d'ajout d'un élément dans la liste.

3/ écrire une procédure de suppression d'un caractère de la liste.

5/ écrire une procédure d'affichage de tous les éléments de la liste.

6/ écrire le programme appelant qui :

-Demande à l'utilisateur d'entrer un nombre quelconque de caractères se terminant par F ; chaque caractère doit s'enregistrer dans un nœud de la liste :

- affiche la liste finale

-supprime les doublons.

- affiche la liste résultat après la suppression.