LYCEE PRIVE HANNIBAL 📖 DISCIPLINE : Informatique	PROFESSEUR: Mr Bayahi Raouf
Chapitre 6	LES STRUCTURES DE DONNEES

Activité 1 :

On se propose de calculer et afficher le périmètre d'un cercle de rayon donné R de type entier.
On vous demande d'établir la grille d'analyse et le tableau de déclaration des objets.

Analyse

Grille d'analyse

	NOM : ……	
S	**L.D.E**	**O.U**
….	**Résultat** = ……	……
….	……	……
….	……	……
….	……	
….	**Fin** ……	

Tableau de déclaration des objets

Objet	Nature / Type
….	……
….	……
….	……

I. Les objets

Un programme est une suite d'instructions permettant de manipuler des objets. Un objet est une case mémoire dont sa gestion est prise en charge par le programme qui l'utilise.

Un objet peut être :

- Une constante
- Une variable

I.1 ***Les constantes***

Une constante est une zone mémoire dans laquelle est stockée une valeur. Comme son nom l'indique, cette valeur restera la même pendant toutes les périodes d'exécution du programme. Une constante est caractérisée par :

- Son nom (unique, simple, parlant, efficace, sans accent ni caractères spéciaux).
- Sa valeur.

Exemple : Pi = 3.14 {real} Y = false {boolean} T = 'm' {char}

A = 147 {integer} C = 'salut' {string}

Tableau de déclaration des objets

Objet	Nature / Type
Pi	Constante = 3.14

En pascal, la déclaration d'une constante se fait comme suit :

CONST « nom_constante » = « valeur_constante » ;

Exemple : CONST Pi = 3.14 ;

Remarque : la valeur prise par la constante nous informe sur le type de cette constante.

I.2 ***Les variables***

Comme son nom l'indique, une variable est une zone mémoire qui peut changer de valeur d'une exécution à une autre.

Toute variable est caractérisée par :

- Son nom (unique, simple, parlant, efficace, sans accent ni caractères spéciaux).
- Son type : qui décrit son utilisation.
- Son contenu : qui représente les différentes valeurs au cours de l'exécution.

Exemple :

Tableau de déclaration des objets

Objet	Nature / Type
A	Entier
R	Réel
C	Caractère
CH	Chaîne de caractères

Remarque : l'opération permettant de changer le contenu d'une variable est appelée ***affectation*** qu'on la désigne par le symbole←.

Activité 2 :

Soit la séquence d'affectations suivante :

1) x ← 10
2) y ← 2
3) z← x
4) x ← y
5) y ← z

Q1/ Donner le résultat d'exécution de cette séquence en complétant le tableau suivant :

Trace de la séquence			
N° de l'instruction	**x**	**Y**	**z**
1			
2			
3			
4			

5			

Q2/ Quelles sont les valeurs finales de x et y ?

Réponse : ……..

Q3/ Quel est le rôle de cette séquence ?

Réponse : ……..

Q4/ Quel est le rôle de la variable z ?

Réponse :

……..

……..

……

En pascal, la déclaration d'une variable se fait comme suit :
VAR « nom_variable » : «type_variable » ;

Exemple : A : INTEGER ; R : REAL ; C : CHAR ; CH : STRING ;

II. Types des objets

II.1 *Les types simples prédéfinis*

A / Le type entier (INTEGER)

Les valeurs de type entier forment un sous- ensemble de Z (entiers relatifs). Signalons que les valeurs prises par les variables de ce type ne correspondent pas à l'ensemble infini que l'on rencontre en mathématiques. Des bornes inférieures et supérieures, sont en fait fixées par le constructeur de la machine et qui sont reliées aux nombres d'octets (1 octet = 8 bits) ou de bits utilisés pour représenter ces entiers.

Si les entiers par exemple sont représentés sur n bits, l'ensemble des entiers positifs et négatifs sera :

$$-2^{n-1} \leq \text{entier} \leq 2^{n-1}-1$$

Pour n= 16 (cas de la plupart des micro-ordinateurs), on a :

$$-2^{16-1} \leq \text{entier} \leq 2^{16-1}-1$$

$$-32768 \leq \text{entier} \leq +32767$$

Les types entiers prédéfinis			
Type		***Limites [MinEnt, MaxEnt]***	***Nombres de bits***
En algorithme	***En pascal***		
Entier court	SHORTINT	[-128, +127]	Signé 8
Octet	BYTE	[0, 255]	Non signé 8
Entier	INTEGER	[-32768, +32767]	Signé 16
Mot	WORD	[0, 65535]	Non signé 16

Entier long	LONGINT	$[-2^{31}, +2^{31}-1]$	Signé 32

Exemple de nombres entiers : -32520 0 29560 10 -50 6

Remarque : quand il y a débordement au delà des valeurs MinEnt et MaxEnt, les calculs deviennent erronés.

Exemple : soit x une variable de type entier

X ← 20500

X← X + 15000 → la valeur finale de X est 35500, dépasse l'intervalle [-32768, +32767]

Les opérateurs arithmétiques sur les entiers		
Syntaxe	***Rôle***	***Exemple***
Toutes les comparaisons		-50 < 0, 150 > 30, …
+	Addition de deux entiers	35 + 20 = 55
-	Soustraction de deux entiers	100-20 = 80
*	Multiplication de deux entiers	12 * 3 = 36
DIV	Division entière	25 DIV 4 = 6 ; 30 DIV 6 = 5
MOD	Reste de la division entière	25 MOD 4 = 1 ; 30 MOD 6 = 0
Dans (IN)	Appartenance à un intervalle	10 Dans [0, 10]

En pascal, la déclaration d'une variable entière se fait comme suit :

VAR « nom_variable » : INTEGER ;

Exemple :

VAR R, S, P : INTEGER ;

B / Le type réel (REAL)

Les valeurs de type réel forment un sous–ensemble de l'ensemble R**.**

Les types réels prédéfinis			
Type	***Valeurs autorisées***	***Nombre de chiffres significatifs***	***Occupation en mémoire***
SINGLE	$[1.5*10^{-45}, 3.4*10^{38}]$	7 chiffres	4 octets
REAL	$[-2.9*10^{-39}, 1.7*10^{38}]$	11 chiffres	6 octets
DOUBLE	$[5.0*10^{-324}, 1.7*10^{308}]$	15 chiffres	8 octets
EXTENDED	$[3.4*10^{-4932}, 1.1*10^{4932}]$	19 chiffres	10 octets
COMP	$[-9.2*10^{18}, 9.2*10^{18}]$	20 chiffres	8 octets

Exemple : 0.0 2.0 -1.5 3.10^4 -14.5

Le nombre réel 3.10^4 (30000) peut s'écrira aussi 3E+4 ; 3 représente la mantisse du nombre et +4 constitue l'exposant qui doit être entier. La lettre E se lit « dix puissance ».

Les opérateurs arithmétiques sur les réels		
Syntaxe	***Rôle***	***Exemple***
Toutes les comparaisons		2.4 < 2.6, -1.5 > -3.7, …
+	Addition	3.5 + 2.2 = 5.7
-	Soustraction	10.9-2.1 = 8.8
*	Multiplication	1.5 * 3 = 4.5
/	Division réelle	25 / 4 = 6.25 ; 12.5 / 5 = 2.5

En pascal, la déclaration d'une variable réelle se fait comme suit :

VAR « nom_variable » : REAL ;

Exemple :

```
VAR
        X, Y : REAL ;
```

Les opérateurs arithmétiques et leurs priorités

Soit l'opération suivante : C = A + B avec { C : résultat ; A et B : opérandes ; + : opérateur }

Désignation de l'opération	Priorité des opérateurs	Opérateur		Type des opérandes
		En algorithme	***En Pascal***	
Parenthèses	1	(…)	(…)	Tout type
Multiplication	2	X	*	Entier ou réel
Division réelle		/	/	Réel
Division entière		DIV	DIV	Entier
Reste de la division entière		MOD	MOD	Entier
Addition	3	+	+	Entier ou réel
Soustraction		-	-	Entier ou réel
Egale	4	=	=	Tout type ordonné
Différent		≠	<>	Tout type ordonné
Inférieur		<	<	Tout type ordonné
Supérieur		>	>	Tout type ordonné
Inférieur ou égale		≤	<=	Tout type ordonné
Supérieur ou égale		≥	>=	Tout type ordonné
L'appartenance	5	DANS	IN	Type scalaire

Remarque : les opérateurs de même niveau de priorité seront évalués de gauche vers la droite (vous pouvez utiliser les parenthèses pour modifier l'ordre d'évaluation des expressions). ***Exemple***:

Evaluer les expressions suivantes :
1/ 120 + 12 * 5 – 3
……
Vous pouvez modifier cet ordre en ajoutant des parenthèses. L'expression 120+12*(5 – 3) sera évaluée de la manière suivante :
……..
2/ 6 + 2 * 5 DIV 3
……..
3/ 6 + 4 * 7 – 2 * (8 MOD 3+5)
……..

Les fonctions arithmétiques standard					
Syntaxe en algorithme	***Syntaxe en Pascal***	***Rôle de la fonction***	***Type de x***	***Type de résultat***	***Exemples***
Abs (x)	**ABS (x)**	Retourne la valeur absolue de x.	***Entier ou réel***	***Même type que x***	R := ABS (-10) ; ⇨R= 10. R := ABS (-5.5) ; ⇨R= 5.5.
Arrondi (x)	**ROUND (x)**	Retourne l'entier le plus proche de x.	***Réel***	***Entier***	R := ROUND (8.4) ; ⇨R= 8. R := ROUND (8.5) ; ⇨R= 9. R := ROUND (8.7) ; ⇨R= 9.
Carré (x)	**SQR (x)**	Retourne le carré de x.	***Entier ou réel***	***Même type que x***	R := SQR (3) ; ⇨R= 9. R:= SQR (3.5); ⇨R= 12.25.
Cos (x)	**COS (x)**	Retourne le cosinus de x (x en radians).	***Réel***	***Réel***	R:= COS (PI/2); ⇨R= 0. R:= COS (PI); ⇨R= -1.
ent (x)	**int (x)**	Retourne la partie entiére de x	***Réel***	***Entier***	Int(2.5)=2 Int(-3.5)=-4
RacineCarré (x)	**SQRT(x)**	Retourne la racine carrée de x si x est positif sinon il provoque une erreur.	***Réel***	***Réel***	R:= SQRT (4); ⇨R= 2. R:= SQRT (40.5); ⇨R= 6.36.
Sin (x)	**SIN (x)**	Retourne le sinus de x (x en radians)	***Réel***	***Réel***	R:= SIN (PI/2); ⇨R= 1. R:= SIN (PI); ⇨R= 0.
Tronc (x)	**TRUNC (x)**	Retourne un entier, en ignorant la partie décimale de x.	***Réel***	***Entier***	R:= TRUNC (-1.5); ⇨R= -1. R:= TRUNC (9.5); ⇨R= 9.
Aléa(x)	**Random (x)**	Retourne un réel aléatoire entre 0 et n	***entier***	***réel***	

Exemples:
1/ Soit x une variable qui a pour valeur l'expression suivante :
x ← carré (abs (-2))
……
Après avoir exécuté cette instruction, la valeur finale de x est égale à …….

2/ Soit y une variable qui a pour valeur l'expression suivante :

y← Arrondi (RacineCarré (4+3*4) + tronc (-12-1.8/2))

……

……

……

Après avoir exécuté cette instruction, la valeur finale de y est égale à …….

C / Le type Booléen (BOOLEAN)

Appelé aussi type logique (du nom du mathématicien la Boole, qui en a développe une algèbre).

La valeur d'une variable booléenne peut prendre comme valeur soit VRAI (TRUE) soit FAUX (FALSE).

Chaque variable booléenne est stockée en mémoire sur 1 octet.

Les opérateurs logiques sur les booléens					
Syntaxe en algo	***Syntaxe en Pascal***	***Rôle***	***Exemple***		
Toutes les comparaisons avec FAUX < VRAI (FALSE < TRUE)					
ET	**AND**	ET logique (conjonction)	**X**	**Y**	**X *ET* Y**
			Vrai	Vrai	…….
			Vrai	Faux	…….
			Faux	Vrai	…….
			Faux	Faux	…….
OU	**OR**	OU logique (disjonction)	**X**	**Y**	**X *OU* Y**
			Vrai	Vrai	…….
			Vrai	Faux	…….
			Faux	Vrai	…….
			Faux	Faux	…….
OUex	**XOR**	OU exclusif	X ***XOR*** Y est vrai si X et Y n'ont pas la même valeur logique.		
NON	**NON**	NON logique	X		***NON*** (X)
			Vrai		…….
			Faux		…….

Les opérateurs logiques et leurs priorités				
Désignation de l'opération	**Priorité des opérateurs**	**Opérateur**		**Type opérande**
		En algorithme	***En Pascal***	
Négation logique	1	NON	NOT	Booléen
Conjonction ET	2	ET	AND	Booléen
Disjonction OU	3	OU	OR	Booléen
OU exclusif		OUex	XOR	Booléen

Exemples :

Evaluer les expressions logiques suivantes :

1/ (-2 <3) ET (5 < 0)

……..

2/ NON (4 >-5) ET (10 > 8)

……..

3/ (3*2+1 < 9) OU (6 MOD 2 > 9)

……..

En pascal, la déclaration d'une variable booléenne se fait comme suit :

VAR « nom variable » : BOOLEAN;

Exemple :

```
VAR
    Trouve, ok : BOOLEAN ;
```

D / Le type caractère (CHAR)

Il est réservé aux variables contenant ***un*** et ***un seul caractère***. Il s'agira en l'occurrence des lettres (minuscules et majuscules), des chiffres, des signes de ponctuation et des symboles spéciaux.

Pour représenter un caractère en Pascal, on peut placer sa valeur entre 2 apostrophes (***ex***:'a','+', ...).

Il est possible de déterminer les successeur / prédécesseur / position d'un caractère dans la liste des codes ASCII (voir livre page 200). Ainsi le successeur de "B" est "C", son prédécesseur "A" et son code ASCII 66.

Un caractère est stocké sur un octet.

L'espace est un caractère « blanc ».

Les opérateurs sur les caractères		
Syntaxe	***Rôle***	***Exemple***
Toutes les comparaisons		"A" < "B", "y" > "d",…
DANS (IN)	Appartenance à un intervalle	"C" DANS ["A","Z"]

En pascal, la déclaration d'une variable de type caractère se fait comme suit :

VAR « nom_variable » : CHAR;

Exemple :

```
VAR
    CAR, C1, C2 : CHAR ;
```

Les fonctions prédéfinies sur les caractères

Syntaxe en algorithme	***Syntaxe en Pascal***	***Rôle de la fonction***	***Type de x***	***Type de résultat***	***Exemples***
CHR (N)	**CHR (N)**	Retourne le caractère dont le code ASCII est N.	***Entier***	***Caractère***	R := CHR (65) ; ⇨R sera égale à A. R := CHR (97) ; ⇨R sera égale à a.
ORD (C)	**ORD (C)**	Retourne le code ASCII du caractère C.	***Caractère***	***Entier***	R := ORD ('D') ; ⇨R sera égale à 68. R := ORD ('%') ; ⇨R sera égale à 37.
PRED (C)	**PRED (C)**	Retourne le prédécesseur de C (c'est à dire qui précède C).	***Scalaire***	***Même type de C***	N :=PRED (4) ; ⇨N sera égale à 3. R :=PRED ('D') ; ⇨R sera égale à 'C'.
SUCC (C)	**SUCC (C)**	Retourne le successeur de C (c'est à dire qui suit C).	***Scalaire***	***Même type que x***	N := SUCC (3) ; ⇨ N sera égale à 4. R:= SUCC ('C'); ⇨R sera égale à 'D'.
MAJUS (C)	**UPCASE (C)**	Convertir le caractère C en majuscule s'il est possible.	***Caractère***	***Caractère***	R:= UPCASE ('e'); ⇨R sera égale à 'E'. R:= UPCASE ('F'); ⇨R sera égale à 'F'.

Remarque :

"x" : désigne le caractère x.
x : désigne une variable.
4 : désigne l'entier4.
"4": désigne le caractère 4.
CHR (ORD (c))= c
ORD (CHR (n))= n

Exemple :

1/ Soit x une variable qui a pour valeur l'expression suivante :
x ← MAJUS (CHR (PRED(110)))
……

Après avoir exécuté cette instruction, la valeur finale de x est égale à …….

Remarques :

- une variable de type scalaire est une information qui possède un successeur et un prédécesseur.
- Le type entier et caractère sont des types scalaires.
- Les types scalaires bénéficient de 2 fonctions Succ et Pred.

Exemple :

Succ ('B') = 'C'

Pred (5) = 4

II.2 ***Les types structurés***

A / Le type chaîne de caractères (STRING)

Une variable de type chaîne de caractères peut contenir :

- soit une suite de caractères (un mot, une phrase, ...),
- soit un caractère (mais dont, par exemple, il est impossible de déterminer le suivant),
- soit aucun caractère (on parle alors de chaîne vide).

Cependant, Pascal permet aussi de préciser la taille maximale que pourra avoir la chaîne qui sera affectée à la variable. En l'absence de précision de longueur, Pascal réserve automatiquement la taille maximale à 255 caractères.

Une chaîne de caractères est délimitée par deux guillemets en algorithme et deux apostrophes en Pascal.

Exemple **:**

Tableau de déclaration des objets

Objet	Nature / Type
Ident_chaine	***Chaîne[longueur]***
Nom Adresse	Chaîne [20] Chaîne de caractères

En pascal, la déclaration d'une variable de type chaîne de caractères e se fait comme suit :

VAR « nom_variable » : STRING;

Exemple :

```
VAR
    Nom : STRING[20] ;
    Adresse : STRING ;
```

Accès aux éléments d'une chaîne de caractères

Pour accéder en lecture et en écriture au $i^{ième}$ élément d'une chaîne de caractères, il suffit de donner le nom de la chaîne suivi de l'indice i entre deux crochets avec $1 \leq i \leq$ long (nom_chaîne).

Exemple :

Soit CH une variable de type chaîne de caractères.

CH← "Enseignant"

Dans la RAM, la chaîne "Enseignant" est stockée de la manière suivante :

CH

E	n	S	e	i	g	n	a	N	t
1	2	3	4	5	6	7	8	9	10

CH[1] donne "E"

CH[2] donne "n"

CH[5] donne "i"

CH[11]← "e" CH devient "Enseignante"

Les opérateurs sur les chaînes de caractères		
Syntaxe	***Rôle***	***Exemple***
Toutes les comparaisons		"Cours" < "cours",…
+	Concaténation de deux ou plusieurs chaînes	"Cours" + " " + " exercices" = "Cours exercices"

Les fonctions standard sur les chaînes de caractères

Syntaxe en algo	*Syntaxe en Pascal*	*Rôle de la fonction*	*Exemples*
Long (ch)	**LENGTH (ch)**	Retourne ***un entier*** représentant la longueur de la chaîne **ch**.	L := LENGTH ('Algorithme') ; L= ……. L := LENGTH ('Pascal') ; L= …….
Concat (ch1, ch2, …, chn)	**CONCAT(ch1, ch2, …,chn)**	Retourne ***une chaîne*** qui est la somme de plusieurs chaînes dans l'ordre.	CH := CONCAT ('micro-', 'ordinateur') ; ⇨CH= '……'. CH := CONCAT ('Turbo', ' ', 'Pascal') ; ⇨CH= '……'.
Sous_chaîne (ch,p,n)	**COPY (ch,p,n)**	Retourne une ***sous chaîne*** de longueur **N** à partir de la position **p** dans **ch**.	CH := COPY ('Baccalauréat',1,3) ; ⇨CH= '……'. CH := COPY ('micro-ordinateur',7,10) ; ⇨CH= '……'.
Pos (ch1,ch2)	**POS (ch1,ch2)**	Retourne un ***entier*** représentant la position de la première occurrence de la chaîne **ch1**. Si **ch1** n'est pas dans **ch2**, elle retourne 0.	P := POS('m', 'programmation') ; ⇨P= ….. P := POS('r', 'programmation') ; ⇨P= ……. P := POS('R', 'programmation') ; ⇨P= ……

Les procédures standard sur les chaînes de caractères

Syntaxe en algo	*Syntaxe en Pascal*	*Rôle de la fonction*	*Exemples*
Efface (ch,p,n)	**DELETE (ch,p,n)**	Enlève **n** caractères de la chaîne **ch** à partir de la position **p**.	CH := 'programmation' ; DELETE ('programmation',8,6) ; ⇨CH= '……'
Insère (ch1,ch2,p)	**INSERT (ch1,ch2,p)**	Insère la chaîne **ch1** dans la chaîne **ch2** à partir de la position **p**. Le caractère n°p et les suivants seront décalés vers la droite.	CH1 := '-' ; CH2 := 'Hautparleurs' INSERT (CH1,CH2,5) ; ⇨CH2= '……'
Convch (n,ch)	**STR (n,ch)**	Convertit une valeur numérique en une chaîne de caractères et l'affecte à la variable **ch**.	STR (2002, CH) ; ⇨CH= '……' STR (15.54, CH) ; ⇨CH= '1.5540000000E+01'
Valeur (ch,d,pe)	**VAL (ch,n, pe)**	Convertit une chaîne **ch** en une valeur numérique et l'affecte à la variable **n**. Le paramètre **pe** est une variable entière qui contiendra la position de l'erreur.	VAL ('2003',n,pe) ; ⇨ n= ……. et pe=…. VAL ('06/08/1970', n,pe) ; ⇨ n= …… et pe= … (le caractère / n'est pas un chiffre).

B / Les tableaux

Activité :
on veut calculer puis trier par ordre croissant les moyennes de 30 élèves. Nous utiliserons 30 variables différentes de type réel moy 1, moy 2, moy 3, …… moy 30. Nous remarquons que ce nombre de variables deviendra plus grand si nous avons un nombre important d'élèves.

Quelle est donc la solution ?

Pour résoudre ce problème, on propose de déclarer un tableau de 30 éléments pour stocker les moyennes au lieu d'utiliser 30 variables différentes.

Ces moyennes se distinguent avec des indices allant de 1 à 30

MOY	15.57	15.43	14.98	13.5	12.90	……	9.78	9.48
	1	2	3	4	5	……	29	30

15.57 correspond au 1ère élément du tableau MOY

14.98 correspond au 3ème élément du tableau MOY

9.48 correspond au 30ème élément du tableau MOY

Définition

Un tableau est une structure de données permettant de ranger un nombre fixe d'éléments de même type. Chaque élément du tableau est désigné par un indice qui doit être forcément de type scalaire (entier, caractère).

Un tableau est caractérisé par :

- Son nom (identificateur)
- Le nombre de ses éléments et
- Le type de ses éléments.

Tableau de déclaration des objets

Objet	Nature / Type
Ident_tableau	***Tableau de taille et de type élément***
MOY	Tableau de 30 réels

En pascal, la déclaration d'un tableau se fait comme suit :

VAR Ident_tableau : ARRAY [borne_inf..borne_sup] OF type_élément;

Exemple : la déclaration du tableau MOY est la suivante :

VAR

MOY : ARRAY [1..30] OF REAL ;

Accès aux éléments d'un tableau

Pour accéder en lecture ou en écriture au $i^{ième}$ élément d'un tableau, il suffit de donner l'identificateur du tableau suivi de l'indice i entre deux crochets (avec borne_inf ≤ i ≤ borne_sup).

Exemple :

MOY[1] donne 15.57

MOY[3] donne 14.98

MOY [29] ← 9.80 MOY[29] devient 9.80

Remarques :

- les types simples ne permettent pas de mémoriser plus d'une information par variable, à l'exception de type STRING. Pour stocker plusieurs données de même type dans une même variable, il faut faire appel au type ARRAY.
- On peut lire, écrire et modifier un élément d'un tableau.
- Les opérations possibles sur un élément du tableau sont les mêmes que celles définies sur une variable de même type.
- Il est possible de déclarer le type d'un tableau.

Exemple :

Tableau de déclaration de nouveaux types

Type
TAB = tableau de 30 réels Classe = tableau de 30 chaînes de caractères

Tableau de déclaration des objets

Objet	Nature / Type
MOY	TAB
NOM	Classe