RESUME DU COURS DE MATHEMATIQUES

APPLICATIONS

DENOMBREMENT

INFORMATIQUE

LANGAGE PASCAL

I–GENERALITES

1) Le langage Pascal est fondé sur l'algorithmique, c'est-à-dire l'utilisation et la mise en place d'un ensemble de règles opératoires dont l'application permet de résoudre un problème énoncé en un nombre fini d'opérations.

2) Pour ce faire, l'opérateur manipule des objets qu'on appelle des données. Au niveau des composants électroniques, les données sont repésentées par des suites d'éléments binaires (0 ou 1) ou bits (binary digits).

3) Un type de données définit un ensemble de valeurs que peut prendre une variable informatique (case mémoire). Les types de données utilisés en Pascal sont dits simples et sont soit prédéfinis dans l'ordinateur soit créés par l'utilisateur.

TYPES UTILISES EN PASCAL

TYPE BOOLEEN (boolean) : une valeur booléenne est une valeur vraie (true) ou fausse (false)

⟶ and : A and B =true $\iff$ A=true et B=true.

⟶ or : A or B =true $\iff$ A=true ou B=true (l'une au moins des deux propositions est vraie).

⟶ xor : A xor B =true $\iff$ A=true ou B=true (le ou est exclusif : une seule des deux propositions est vraie).

⟶ not : not A=true $\iff$ A est fausse.

⟶ odd : s'applique aux entiers ; si x est entier, odd(x)=true $\iff$ x est impair.

TYPE ENTIER (integer) : c'est la partie des entiers relatifs que peut écrire l'ordinateur. Il y en a en général 2^{16} possibilités (cela dépend en fait de l'ordinateur). Il existe une extension, le type longint (long entier) : théoriquement 2^{32} possibilités (en fait c'est une valeur minimale)

 © EDUKLUB SA
Tous droits de l'auteur des oeuvres réservés. Sauf autorisation, la reproduction ainsi que toute utilisation des oeuvres autre que la consultation individuelle et privée sont interdites.

Extrait gratuit de document, le document original comporte 9 pages.

$\mathbb{Z}$ → En général un tel entier ne peut pas figurer dans une boucle.

⟶ **Opérateurs s'appliquant aux entiers et donnant (rendant) un résultat entier**

⟶▷ : + , ∗ , −. Addition, multiplication et soustraction.

⟶▷ : div est quotient entier dans la division de deux entiers : 7 div 2 donne comme résultat 3.

⟶▷ : mod est reste entier dans la division euclidienne de deux entiers : 7 mod 2 donne comme résultat 1 (dans le cas général a mod b=a-(a div b)*b)

⟶ **Opérateurs :** = ; <> (différent) ; <= (inférieur ou égal) ; < (inférieur strict) ; >= (supérieur ou égal) ; > (supérieur strict), donnent un résultat booléen quand on les applique aux entiers.

⟶ abs est la valeur absolue d'un entier et sqr est le carré d'un entier.

TYPE REEL (real) : c'est la partie des nombres réels que peut écrire l'ordinateur

⟶ fonctions prédéfinies : sin , cos , tang , arctang , ln , exp , sqrt (racine carré).

Sans oublier les opérateurs s'appliquant aux entiers et qui peuvent s'appliquer aux réels.

TYPE CHARACTERE (char) :

⟶ : c'est le type des lettres de l'alphabet, des chiffres 0,1,...,9, du blanc.

⟶ : un caractère entre apostrophes représente une constante de type char. 'G' représente la lettre G , mais G, sans apostrophe sera considéré comme une variable informatique, une case mémoire, et l'ordinateur va évaluer son contenu.

⟶ : deux fonctions sont définies sur le type char.

▷ ord : ord(c) est égal au rang du caractère c dans une chaine de caractère.

▷ char : chr(i) est égal au caractère de rang i dans une chaine de caractère.

TYPE DE DONNEES A DEFINIR PAR L'UTILISATEUR

- **a) Type tableau** : c'est une liste de cases mémoires.

Déclaration : type tableau = array[1..N] of trucs .

Le nombre N doit être connu **numériquement** au moment de la déclaration et trucs représente un type prédéfini dans le Pascal ; integer, real, boolean, char. Il est clair que le nom tableau n'est pas imposé, c'est notre choix, on peut déclarer un type tab par exemple.

Remarque : Si t est un tableau de longueur N, la case numéro k sera notée t[k] ; c'est la k $^{\text{ème}}$ case mémoire de la liste.

- **b) Type matrice**

Déclaration : type matrice = array[1..n,1..p] of real ; c'est une matrice appartenant à $\mathcal{M}_{n,p}(\mathbb{R})$. Il faut évidemment que n et p soient connus numériquement.

Remarque : Si m est une matrice de type matrice, le terme de la i $^{\text{ème}}$ ligne et j $^{\text{ème}}$ colonne est noté m[i,j].

Tous droits de l'auteur des oeuvres réservés. Sauf autorisation, la reproduction ainsi que toute utilisation des oeuvres autre que la consultation individuelle et privée sont interdites.

Extrait gratuit de document, le document original comporte 9 pages.

- **c) Type polynôme**

Déclaration : type poly = array[0..n] of real ; c'est la liste (ou le tableau) des coefficients.

II–STRUCTURES DE BASE

1) VARIABLE INFORMATIQUE

C'est une case mémoire dans laquelle on met un type de données. Pour créer une variable informatique, il faut utiliser un identificateur - son nom - et indiquer (déclarer) son type.

Exemples : var x : integer ; var z : real ; var b : boolean ; var t : tableau (**à condition, bien-sûr, que le type tableau, qui n'est pas prédéfini dans le langage Pascal, ait été défini par l'utilisateur antérieurement).**

2) INSTRUCTION FONDAMENTALE : l'affectation.

C'est le fait de mettre dans une variable informatique une donnée du type correspondant à la variable en question.

Exemples : x:=5 ; x est une variable de type entier dans laquelle on met la valeur 5 ; y:=2.3 ; y est une variable de type réel dans laquelle on met la valeur 2.3 ; z:=false ; z est une variable de type boolean dans laquelle on met la valeur false.

$\mathbb{Z}$ → si l'on effectue la suite d'instructions x:=5 ; x:=0 ; et que l'on demande d'afficher la valeur de x, on obtiendra 0 (on dit que l'on a écrasé la valeur 5 - penser à une pile d'assiettes que l'on regarde par dessus)

On séparera toujours deux instructions par un point-virgule.

3) PROCEDURES D'ENTREE-SORTIE

⟶▷ **a) Entrée** read ; readln ;

⟶ read(x) ; l'ordinateur attend que l'utilisateur entre, au clavier, une donnée du type de la variable informatique x, **variable qui doit avoir été déclarée auparavant**; lorsque la valeur est entrée, l'ordinateur la saisit, l'écrit à l'écran à l'endroit où se trouve le curseur, qui se positionnne alors juste après, et, **c'est le plus important,** l'ordinateur affecte cette valeur à la variable x.

⟶ readln(x) ; c'est la même chose sauf que le curseur passe à la ligne.

⟶▷ **b) Sortie** write ; writeln ;

Il y a deux situations :

⟶ Ecrire une chaine de caractères (à l'endroit où se trouve le curseur) ; il faut écrire cette chaine entre apostrophes ;

⟶ Ecrire ce qu'il y a dans une variable : ne pas mettre d'apostrophes pour que l'ordinateur évalue ce qu'il y a dans la variable.

Exemple : x est une variable de type integer, dans laquelle il y a la valeur 5.

write('x') donnera x et write(x) donnera 5.

Sorties sur l'imprimante : il faut avoir déclaré, immédiatement après le titre du programme, que l'on va utiliser l'imprimante par l'instruction uses printer ;

 © EDUKLUB SA
Tous droits de l'auteur des oeuvres réservés. Sauf autorisation, la reproduction ainsi que toute utilisation des oeuvres autre que la consultation individuelle et privée sont interdites.

Extrait gratuit de document, le document original comporte 9 pages.