Rappels importants

1. Conventions de nommage des variables

Des identifiants bien choisis permettent au codeur de comprendre beaucoup plus facilement ce que fait un système et comment corriger ou étendre le code pour répondre à de nouveaux besoins. Par exemple, bien que

a = b * c ;

est syntaxiquement correct, son but n'est pas évident. Comparez cela avec :

salaire_hebdo = heures_travaillées * taux_de_salaire_horaire ;

1. Conventions de nommage des variables

➔ Des noms plus courts mais “parlants” peuvent être préférés car plus pratiques ;

❌ **x, y, temp123**
✅ **nomClient, totalPrix, dateCreation**

1. Conventions de nommage des variables

➔ Pas d'utilisation d'espace dans les noms => utilisation de majuscules ou du symbole underscore (_);

❌ **une variable, date creation**
✅ **uneVariable, date_creation**

1. Conventions de nommage des variables

➔ Ne doit pas commencer par un chiffre ou un caractère alphanumérique ;

❌ **1erClient, @adresse**
✅ **premierClient, adresseClient**

1. Conventions de nommage des variables

➔ Ne doit pas être un mot réservé ;

❌ **for, while, int**
✅ **boucleFor, compteurWhile, nombreEntier**

1. Conventions de nommage des variables

➔ Chaque nom de variable doit être unique ;

❌ **Deux variables nommées totalPrix dans le même algorithme**
✅ **Noms uniques pour chaque variable, comme totalPrix et totalQuantite**

1. Conventions de nommage des variables

➔ Un nom de variable ne peut contenir que des caractères alphanumériques et des traits de soulignement (a-z, A-Z, 0-9, et _) ;

❌ **nom-client, prix total, client@nom**
✅ **nom_client, prixTotal, nomClient**

1. Conventions de nommage des variables

➔ Les noms des variables sont sensibles à la casse.

La variable **prixTotal** et une autre **PrixTotal** sont différentes.

1. Conventions de nommage des variables

➔ De manière générale, le nom d'une variable doit commencer par une lettre minuscule.

2. Syntaxes

Afficher une valeur :
Utilisée pour afficher du texte ou une valeur à l'écran.
afficher("Texte à afficher")
afficher(variable)

Saisir une valeur :
Utilisée pour demander à l'utilisateur de saisir une valeur et l'affecter à une variable.
saisir(variable)

2. Syntaxes

Afficher une valeur :
Utilisée pour afficher du texte ou une valeur à l'écran.
ecrire("Texte à afficher")
ecrire(variable)

Saisir une valeur :
Utilisée pour demander à l'utilisateur de saisir une valeur et l'affecter à une variable.
lire(variable)

2. Syntaxes

Pour afficher du texte et une variable dans la même instruction en algorithmique, il est courant de concaténer le texte avec la valeur de la variable.

afficher("Un texte à afficher : ", variable)

ecrire("Un texte à afficher : ", variable)

Algorithme et Programmation C

Séance 3

Les structures de contrôle

Objectifs

- ➔ Utiliser les structures alternatives ou conditionnelles
- ➔ Utiliser les structures itératives

Contenu

1. Les structures alternatives ou conditionnelles
2. Les structures itératives

1. Les structures alternatives ou conditionnelles

1.1. Structure **Si**

Si Condition alors

Instruction 1

...

Instruction n

Finsi

Exemple :

Ecrire un algorithme qui calcule et affiche la valeur absolue d'un nombre saisi par l'utilisateur.

1.2. Structure **Si**

Si condition **alors**

Instruction 1

Sinon …

Instruction 2

Finsi

➔ Exemple : Écrire un algorithme qui demande l'âge à l'utilisateur et dit s'il est mineur ou adulte.

NB : On admettra qu'un mineur est une personne ayant moins de 18 ans.

1.3. Structure **Selon...que**

➔ Permet une présentation plus claire d'un ensemble d'alternatives imbriquées.

Selon que variable **faire**
    cas valeur1:
        Instructions1
    cas valeur2:
        Instructions2
    autre:
        Instructions par défaut
Finselonque

1.2. Structure de choix

➔ Permet une présentation plus claire d'un ensemble d'alternatives imbriquées.

Exemple : À partir d'un menu affiché à l'écran, effectuer la somme, le produit ou la moyenne de 3 nombres. Nous appelons menu l'association d'un séquentiel aux différentes opérations proposées par un programme.

2. Les structures itératives

2.1. Structure **Pour**

- Permet de répéter une action pendant un nombre de fois **connu à l'avance.**

Pour I=valeur_intiale à valeur_finale pas=val **faire**

Ensemble des instructions à répéter

Fin pour

- **I** est appelé **compteur**,
- **pas** est appelé **incrément**; s'il n'est pas précisé, il vaut automatiquement **+1** ou **-1** suivant les valeurs extrêmes.

2.1. Structure **Pour**

Exemple : Écrire un algorithme qui affiche la table de multiplication d'un chiffre entré par l'utilisateur.

2.2. Structure **Tant que**

➔ Répéter un bloc d'instructions tant qu'une condition est vérifiée

Tant que Condition **faire**

Ensemble des instructions à répéter

Fin Tant que

Exemple : Écrire un algorithme qui demande à l'utilisateur un identifiant et un mot de passe qu'il compare à des données contenues dans des constantes. Tant que les valeurs saisies sont incorrectes, le programme refait la demande.

2.2. Structure **Répéter**

➔ Répéter un bloc d'instructions tant qu'une condition est vérifiée

Répéter

Ensemble des instructions à répéter

Jusqu'à Condition

Exemple : Écrire un algorithme qui demande à l'utilisateur un identifiant et un mot de passe qu'il compare à des données contenues dans des constantes. Tant que les valeurs saisies sont incorrectes, le programme refait la demande.

Algorithme et Programmation C

Séance 3

Types de données et opérations

Objectif

➔ connaître les types de données et les manipuler à travers les opérations

Contenu

1. Types prédéfinis
2. Opérations
3. Types énumérés
4. Exercices

1. Types prédéfinis

1.1. Type entier

- Représente le domaine des nombres entiers.
- Exemple :

```
Variables  x : entier
```

1.2. Type réel

- Représente le domaine des nombres réels.
- Exemple :

```
Variables  x : réel
```

1.3. Type logique (booléen)

- Représente le domaine logique qui contient deux valeurs logiques (vrai et faux)
- Exemple :

  Variables x : booléen

1.4. Type caractère

- Représente le domaine des caractères qui contient :
  - les lettres alphabétiques minuscules,
  - les caractères numériques,
  - les caractères spéciaux (., ?, !, <, >, =, *, +, ...etc)
  - le caractère espace
- Exemple :

  Variables x : caractère

1.5. Type chaîne de caractères

- Représente le domaine des caractères accolés
- Une chaîne peut contenir zéro, un ou plusieurs caractères accolés
- Exemple :

  Variables x : chaîne

2. Les opérations

2.1. Opérateur et opérande

- Un **opérateur** est un outil qui permet d'agir sur une variable ou d'effectuer des calculs.
- Un **opérande** peut être : une constante, une variable, un appel de fonction qui sera utilisé par un opérateur.

Exemple

- Dans l'expression « **8 + y** », «**+**» désigne l'opérateur ; « **8** » et « **y** » sont les opérandes.

2.1. Opérateurs

- Les opérateurs arithmétiques : permettent d'effectuer des opérations arithmétiques entre des opérandes numériques : **+, -, *, /, mod**
- Les opérateurs de comparaison : permettent de comparer deux opérandes et produisent une valeur booléenne, en s'appuyant sur des relations d'ordre : **<, <=, >, >=, =, ≠**
- Les opérateurs logiques : combinent des opérandes booléens pour former des expressions logiques plus complexes : **non, et, ou, ou_exclusif (xor)**

3. Types énumérés

3.1. Définition

- définit la liste complète des valeurs qui peuvent être attribuées à une variable appartenant à ce type énuméré.
- est défini par programmeur.

3.2. Déclaration

➔ Se fait avec une syntaxe particulière

Exemples :

➔ Déclaration d'un type contenant la liste des jours de la semaine

types

t_jours= (Lundi, Mardi, Mercredi, Jeudi, Vendredi, Samedi,Dimanche)

➔ Déclaration d'un type contenant la liste des couleurs

types

t_couleurs= (Rouge, Jaune, Vert, Marron, Bleu, Violet)

3.3. Utilisation

➔ **Déclaration de variables**

Variables

jour1, jour2 : t_jours

couleur : t_couleurs

➔ **Affectation de valeurs**

jour1 ← Vendredi

jour2 ← jour1

couleur ← Rouge

3.4. Utilisation

- Il existe trois fonctions pour manipuler les types énumérés
  - ***pred*** : retourne la valeur précédente

    Exemple : pred(Jeudi) = Mercredi
  - ***succ*** : retourne la valeur suivante

    Exemple : succ(Rouge) = Jaune
  - ***ord*** : retourne l'ordre d'un élément (dans la déclaration)

    Exemple : ord(Samedi) = 5

4. Exercices

4.1. Exercice 1

Écrire un algorithme qui calcule la somme de deux nombres complexes saisis par l'utilisateur et affiche le résultat sous forme (a + ib) + (x + iy) = (a+x) + i(b+y).

Exemple : (1 + i6) + (3 -i2) = (4 + i4).

4.2. Exercice 2

Un établissement scolaire souhaite envoyer le message suivant aux parents d'élèves :

Cher parent / tuteur,
A l'issu de l'interrogation du ***11/01/203*** *en* ***Français****, l'élève* ***TOTO Ayi*** *dont vous êtes le parent / tuteur a obtenu la note de* ***12,00*** *sur 20.*

Ecrire un programme qui demande à l'utilisateur de saisir les informations date, matière, nom et note (en gras dans le texte du message).

Ensuite, le programme génère et affiche le message à envoyer au parent.

Références

[1] Y. A.. GBEDEVI, « Initiation à l'algorithmique », Université de Lomé, Support de cours, 2022-2023.

[2] R. Christophe, « Bases d'algorithmique. Support de Cours au Lycée Vincent d'Indy ». 2015-2016.

[3] L, Baba Ahmed et S, Hocine, algorithmique et structure de données statistiques. OPU, 2016.

[4] E. Thiel, « Support de cours Algorithmes et programmation en Pascal ». Faculté des Sciences de Luminy, Université d'Aix-Marseille AMU, 1997.

[5] A. Rabia, F. Rachid, A. O. Mohand, B. Moufida, et Y. Smain, « Algorithmique :Cours et Exercices en Programmation Pascal ». Cours, Exercices et Programmation Pascal Première Année Universitaire, USTHB, 2018-2017