Algorithmique et programmation

Programmation impérative

R.Gosswiller

Sommaire

Retour sur les bases

Les opérateurs

Definition

Un opérateur est un symbole permettant de réaliser une action mathématique ou logique sur un ensemble d'élements.

Principe

Opérateurs mathématiques : addition, soustraction, modulo
Opérateurs logiques : décalage, ET, OU, ...

Les opérateurs

Opérateurs arithmétiques

$+, -, *, /, \%, \ll, \gg$

Opérateurs logiques

$\&, |, \&\&, ||, ? :$

Opérateurs relationnels

$<, >, <=, >=, =, , ==,$

Opérateurs composés

$+=, -=, *=, /=, \%=, \&=, \hat{}=, |=, !=, <<=, >>=$

Les variables

Syntaxe

```
1          type nom;
2
3          int monEntier;
4          char maLettre;
5          double monGrandNombre;
```

Les conditions

Syntaxe

```
1        if(condition){//Instructions}
2
3        if(condition){//Instructions}
4        else{//Instructions}
5
6        if(condition){//Instructions}
7        else if(condition){//Instructions}
8        else{//Instructions}
```

Les conditions

Syntaxe

```
1          if(a==42){
2            f();
3            }
4          else{
5            g();
6          }
```

Les boucles

Syntaxe

```
1          while(condition) {
2            //Instructions
3          }
4
5          for(depart;arret;incrementation) {
6            //Instructions
7          }
```

Les boucles

Syntaxe

```
1          while(a == 42) {
2            do();
3          }
4
5          for(int i=0; i<=42; i++) {
6            do();
7          }
```

Appliquer l'approche fonctionnelle en C

L'approche fonctionnelle en C

Qu'est-ce qu'une fonction ?

Principe

- Nom
- Type et valeur de retour
- Paramètres
- Instructions

L'approche fonctionnelle en C

Modèle algorithmique

Utilisation

Prévoir l'exécution répétée d'instructions
Définir un **prototype** de fonction
Combiner propotype et instructions dans une fonction nouvellement codée

Syntaxe

```
1 typeretour nomFonction(typeargument arg1, typearg arg2, ...) {
2   Instructions
3 }
```

Exemple

```
1 float mult(int a,int b) {
2   return a*b;
3 }
```

Prototypes de fonctions

Définition

Le prototype d'une fonction est l'ensemble des éléments décrivant son utilisation.

Principe

- Nom
- Type de retour
- Paramètres

Syntaxe

```
1          int f(int a,char b);
```

L'approche fonctionnelle en C

Principe

Chaque fonction possédant un **type de retour** produit un résultat
Ce résultat est récupérable dans la fonction **appelante**
Mot-clé : **return**

Principe

Fonction principale : main.
La fonction main est executée lorsque le programme se lance. **Dans l'ordre du fichier, écrire toutes les fonctions du programme avant la fonction main**

Syntaxe

```
1 int main(int argc, char * argv[])
2 {
3         //instructions
4         return 0;
5 }
```

Types de retour usuels

- Numériques : int, float, double, long, short
- Signés : unsigned int, unsigned float, ...
- Caractères : char
- Composés : long long, short short

Principe

Définir une variable, c'est aussi définir sa portée
Variable locale(limitée à une fonction), globale(étendue au programme), constante

La portée d'une variable

Syntaxe

```
1  int a = 42;
2  int f() {
3    int a = 5;
4    printf("\%d",a); // Affiche 5
5  }
6
7  printf("\%d",a); // Affiche 42
```

Concevoir un code modulaire

Concevoir un code modulaire

Objectifs

- Séparer son code
- Rendre des fonctions réutilisables
- Structurer un projet
- Séparer la documentation et l'implémentation
- Travailler en équipe

Concevoir un code modulaire

Principe

- Inclure les fichiers les uns dans les autres
- Appeler les fonctions d'un fichier depuis un autre
- Séparer prototypes et implémentation
- Mot-clé : **include**

Syntaxe

```
1 #include <stdio.h>  // Path OS
2 #include <stdlib.h>  // Path OS
3 #include "myfile.c" // Chemin relatif
```

Les fichiers .h

Définition

Un fichier .h est un fichier contenant l'ensemble des inclusions et des prototypes des fonctions d'un programme.

Principe

- Définir les prototypes dans le fichier .h
- Inclure le fichier .h dans le fichier .c
- Réaliser les autres inclusions dans le fichier .h

La compilation en C

Qu'est-ce que compiler ?

Définition

Compiler signifie transformer le langage écrit (la syntaxe) en langage machine (binaire)

Principe

Cela permet de transformer un code écrit en programme compréhensible par la machine (exécutable).

Compiler en C

Le compilateur

Le compilateur est un programme qui permet de compiler un code source

Exemples

gcc, clang

Compiler en C

Syntaxe

```
1        gcc fichier.c
2
3        gcc a.c b.c c.c
```

Exécution

```
1        ./a.out
```

La fonction printf

Printf

Définition

La fonction printf est la fonction d'affichage en console en C
Elle permet d'afficher un message donné
Renvoie le nombre de caractères imprimés (sans compter le '\0' final).

Syntaxe

```
1       int printf(const char* format, ...);
```

Exemple

```
1       printf("Toto");
2
3       printf("It works !");
4
5       printf("Message");
```

Arguments

Principe

Il est possible d'afficher le contenu d'une variable avec la fonction printf. On utilise alors la notion de pattern.

Exemple

```
1 printf("%d", variable);
2
3 printf("La variable A vaut:%d et
4   la variable B vaut:%d",variableA,variableB);
```

Printf

Les formats

Les formats permettent de spécifier le type des arguments attendus dans la chaîne à afficher

Format	Type
%d	int
%ld	long
%f	float
%lf	double
%c	char
%s	char */string
%p	pointeur/adresse
%hd	short
%x	hexadécimal

Printf et return

Printf

printf est une fonction
Elle affiche une valeur ou un message en console

return

Return est un mot-clé
Il renvoie une valeur inscrite dans une case mémoire

Conclusion

- Mécanismes algorithmiques similaires au Python
- Fonctions, boucles, conditions, variables
- Types
- Langage compilé
- Structure modulaire, Fichiers .h et .c