Algorithmique et programmation c

Chapitre 2 : Les structures conditionnelles et répétitives

Slimene Ons

INSTITUT SUPÉRIEUR DES SCIENCES APPLIQUÉES ET DE TECHNOLOGIE DE SOUSSE

2023-2024

Structures conditionnelles

• On appelle structure conditionnelle toute structure dont les instructions permettent de tester si une condition est vraie ou fausse.

Structures conditionnelles

1. Structure conditionnelle simple (un choix)

Syntaxe:

```
if(condition)
    instruction;
```

```
if(condition)
{
    instruction1;
    instruction2;
}
```

Si la condition est vraie alors le bloc d'instructions sera exécuté sinon il sera ignoré

Exemple

```
#include <stdio.h>

int main() {
    int a, b;

    printf("Entrez la valeur de a : ");
    scanf("%d", &a);
    printf("Entrez la valeur de b : ");
    scanf("%d", &b);

    if (b != 0) {
        int resultat = a / b;
        printf("Le résultat de la division de %d par %d est :
  %d\n", a, b, resultat);
     }
   return 0;
}
```

Ce programme permet d'afficher la valeur de la division de a sur b si la valeur de b est différent de 0

Structures conditionnelles

2. Structure conditionnelle à deux choix

Syntaxe:

```
if(condition)
    instruction1;
else
    instruction2;
```

```
if(condition) {
    instruction1;
    instruction2;
}
else{
    instruction1;
    instruction2;
}
```

Exemple

```
#include <stdio.h>

int main() {
    int a, b;

    printf("Entrez la valeur de a : ");
    scanf("%d", &a);
    printf("Entrez la valeur de b : ");
    scanf("%d", &b);

    if (b != 0) {
        int resultat = a / b;
        printf("Le résultat de la division de %d par %d
  est : %d\n", a, b, resultat);
    }
    else
        printf("la division par 0 est impossible");
   return 0;
}
```

Structures conditionnelles

3. Structure conditionnelle imbriqué (multiple choix)

Syntaxe:

```
if(condition) {
    instructions;
}
else if (condition){
    instructions;
}
else if (condition) {
    instructions;
}
else {
   instructions;}
```

Exemple

```
#include <stdio.h>
int main() {
   int nombre;
   printf("Entrez un nombre : ");
   scanf("%d", &nombre);

   if (nombre > 0){
     printf("Le nombre est positif.\n"); }
   else if (nombre == 0){
     printf("Le nombre est égal à zéro.\n"); }
   else {
     printf("Le nombre est négatif.\n"); }
   return 0; }
```

Ce programme demande à l'utilisateur d'entrer un nombre, puis il utilise des instructions if, else if et else pour vérifier si le nombre est positif, nul ou négatif, et affiche le message approprié en conséquence.

Exercice

Ecrire un programme qui permet de lire la valeur de la température de l'eau et de l'afficher son état:

-GLACE si la température est inférieur à 0
- LIQUIDE si la température est strictement supérieur à 0 et strictement inférieur à 100
- Vapeur si la température est supérieur à 100

Correction

```
#include <stdio.h>
int main() {

float temperature;

printf("Entrez la température de l'eau en degrés Celsius : ");
scanf("%f", &temperature);
if (temperature < 0) {
    printf("L'eau est à l'état solide (glace) à %.2f°C.\n", temperature); }
else if (temperature > 0 && temperature < 100) {
     printf("L'eau est à l'état liquide à %.2f°C.\n", temperature); }
 else {
    printf("L'eau est à l'état gazeux (vapeur) à %.2f°C.\n", temperature);
}
return 0; }
```

Structures conditionnelles

4. Structure à choix multiple : switch

- Switch est un moyen qui permet de décider dans une structure à plusieurs cas. Cette instruction teste si une expression prend une valeur parmi un ensemble de constante et fait le branchement en conséquence.

- Syntaxe:

```
switch (expression)
{
case expr_cst1: liste d'instructions 1; break;
case expr_cst2: liste d'instructions 2; break;
...
default : liste d'instructions N;  break;

 }
```

Structures conditionnelles

- Le choix de l'ensemble d'instructions à exécuter est calculé par l'évaluation de expression qui doit envoyer un type entier.

-expr_cste doit être un int et doit être unique

Exemple:

```
int mois ;
scanf (%d, &mois) ;
switch (mois)
{
        case 1 : printf (janvier) ; break ;
        case 2 : printf (fevrier) ; break ;
        ...
        case 12 : printf (décembre) ; break ;
        default : printf (erreur) ;
}
```

ce programme permet d'afficher le mois de l'année en fonction de numéro de mois

Structures répétitives

• On appelle structure répétitive toute structure qui permet d'exécuter plusieurs fois une séquence d'instructions.

•Dans une boucle, le nombre de répétition peut être fixé à l'avance, comme il peut dépendre d'une condition permettant l' arrêt et la sortie de cette boucle.

Structures répétitives

1. La boucle for

Cette boucle permet d'exécuter une séquence d'instructions un nombre de fois connu fixé à l'avance.

Syntaxe:

```
for (Initialisation ; Condition ; Incrémentation) {
bloc d'instructions;
}
```

Structures répétitives

1. La boucle for

-Initialisation: est évaluée une fois avant le passage de la boucle. Elle est utilisée pour initialiser les données de la boucle.

-Condition: Elle est utilisée pour décider si la boucle est répétée ou non.

-Incrémentation: Cette instruction est exécutée à la fin de chaque tour de boucle pour mettre à jour la variable.

Structures répétitives

1. La boucle for

Exemple 1 :

```
for (i=0 ; i<5 ; i++) {
  printf (" Bonjour ");
}
```

Cette boucle affiche le message « bonjour » 5 fois

Structures répétitives

1. La boucle for

Exemple 2 :

```
#include <stdio.h>
 int main()
 { // Nombre pour la table de multiplication
 int nombre = 8;
 int p;

 for (int i = 1; i <= 10; ++i)
{ p= nombre*i;
  printf("%d x %d = %d\n", nombre, i, p);
}
 return 0; }
```

Exercice

- Ecrire un programme qui permet de calculer la somme de 20 premiers entiers positifs.

Correction

```
#include <stdio.h>

int main() {
    int s, i;
    s=0;

     for(i=0; i<=20; i++)
        s=s+i;

     printf(" la somme des 20 premiers entiers :%d " ,s);


   return 0;
}
```

Structures répétitives

2. La boucle while

Cette boucle permet de répéter un bloc d'instructions tant que une condition est vraie. Sinon sortir de la boucle .

Syntaxe:

```
while (condition) {

    bloc d'instructions;
}
```

Structures répétitives

2. La boucle while

-Les instructions dans le bloc while sont exécutées tant que la condition est vraie .
- Lorsque la condition devient fausse, la boucle while s'arrete, et l'exécution se poursuit à l'instruction suivant le bloc

Exemple 1:

```
/* Afficher les nombres de 0 à 9 */
int i = 0;
while (i < 10)
{  printf(%d \n ,i);
   i++;
}
```

Structures répétitives

2. La boucle while

Exemple 2:

```
#include <stdio.h>

int main() {
    float note;

    printf("Entrer une note entre 0 et 20 ");
    scanf("%f", &note);

    while ( note<0|| note>20) {
       printf("Entrer une note entre 0 et 20 ");
       scanf("%f", &note);}
   printf ("bonjour");
   return 0;
}
```

Ce programme vise à obtenir une note valide de l'utilisateur, en affichant "bonjour".

Structures répétitives

3. La boucle do while

Cette boucle permet de répéter un bloc d'instructions tant que une condition est vraie.

Syntaxe:

```
do {
    bloc d'instructions;
}
while (condition);
```

Remarque : La vérification de la condition s'effectue après l'exécution des instructions. Celles-ci sont donc exécutés au moins une fois.

Structures répétitives

3. La boucle do while

Exemple:

```
float N;
do
{ printf ("Introduisez un nombre entre 1 et 10 ");
scanf ("%f ", &N);
}
 while (N<1 || N > 10)
```

Exercice

Ecrire un programme qui demande à l'utilisateur de saisir un nombre entier strictement supérieur à 1, et qui calcule la somme des entiers jusqu'à ce nombre.
Par exemple, si l'on entre 5 le programme doit calculer :
1+2+3+4+5