INPT
LA GRANDE ÉCOLE DES TÉLÉCOMMUNICATIONS ET DES TECHNOLOGIES DE L'INFORMATION

Filière Smart-ICT

Algorithmique et Programmation C

Mr N.EL FADDOULI

elfaddouli@emi.ac.ma

nfaddouli@gmail.com

Année Universitaire:2024/2025

Plan

Le langage C: Le sélection binaire (1/4)

- La **sélection binaire** permet de choisir **un seul bloc** d'instructions à exécuter parmi **deux blocs possibles**. Le choix est effectué selon une **condition** donnée.
- La syntaxe de la sélection binaire en langage C est la suivante:

```
if (condition)
      {
       bloc1 d'instructions
       }
else {
       bloc2 d'instructions
       }
```

- Le **bloc1** est exécuté lorsque la **condition** est **vraie**, **sinon** c'est le **bloc2** qui est exécuté.
- Les **accolades** sont **optionnelles** si le bloc contient une seule instruction.
- On peut avoir une sélection binaire sans la partie **else**.
- Le **bloc1** et/ou le **bloc2** peut inclure une autre sélection binaire (*sélection imbriquée*)

- **Exemple**: Afficher le maximum de deux entiers

```
void main() { int a, b; printf("Deux entiers:"); scanf("%d%d", &a, &b);
              if (a>b)  { printf(" La maximum est : %d, ", a);  printf(" c'est le premier\n"); }
              else
                        printf("le maximum est %d, c'est le deuxième\n", b);
            }
```

Le langage C: Le sélection binaire (2/4)

- La condition est une **expression booléenne** dont la valeur est **Vrai** ou **Faux** et qui est exprimée en utilisant les **opérateurs de comparaison** et les **opérateurs logiques**.

- En C standard (***avant la norme C99***), le langage **n'avait pas de type booléen natif**. On utilisait des entiers pour représenter les valeurs booléennes Vrai et Faux: **0** était considéré comme **faux** et toute valeur **différente de** 0 était considérée comme **vrai**.

Exemple: int A=5, B=3, C;

C = A>B; ⇨ C= 1

C= A<B; ⇨ C= 0

```
if (C) {...}
else  {…}
```
⇔
```
if (C==1) {…}
else  {…}
```
⇔
```
if (C != 0) {…}
else  {…}
```

```
if (A)  printf("Ok");
else  printf("Non")
```
⇨ OK

Le langage C: Le sélection binaire (3/4)

- Avec la norme **C99**, le langage C a introduit le type booléen sous la forme d'une macro. Le type booléen est défini dans l'en-tête **<stdbool.h>.**
- On a dans ce fichier d'entête:

- **bool :** Un alias pour le type **entier _Bool**, qui est utilisé pour les valeurs booléennes.
- **true** : Une constante égale à 1.
- **false** : Une constante égale à 0.

- Exemple:

```
#include <stdio.h>
#include <stdbool.h>
void  main() {
int A, B;
bool condition ;
….
condition=true;
….
condition = false;
….
condition = A==B;  // condition ← expression logique
```

```
if (condition==true) {…}
else {…}
```

⇔

```
if (condition) {…}
else {…}
```

```
if (condition==false) {…}
else {…}
```

⇔

```
if (!condition) {…}
else {…}
```

Le langage C: Le sélection binaire (4/4)

☞ La condition est exprimée en utilisant les opérateurs de comparaison et les opérateurs logiques

Algorithmique	*C*	*Exemple*

NON	**!**	V est de type **entier** ou **bool**:
		if (**V==0**). … ⇔ if(**!V**)…
		V est de type **bool**:
		if (**V==false**) … ⇔ if(**!V**)…

ET	**&&**	if (A>=B **&&** B>=C) …
OU	**||**	if (A<B **||** B<C) …
= (égale)	**==**	if (A **==** 0)…
≠ (différent)	**!=**	if (A **!=** 0)…

Le langage C: L'opérateur ternaire de sélection

- L'**opérateur ternaire** est une forme compacte de l'instruction de sélection **if-else**.
- Il permet d'évaluer une condition et de choisir entre deux expressions selon que la condition est vraie ou fausse.
- Sa syntaxe est la suivante: ***condition ? expression_si_vrai : expression_si_faux ;***

Expression exécutée si la condition est vraie

Expression exécutée si la condition est fausse

- Cet opérateur est souvent utilisé pour simplifier le code lorsqu'une simple affectation conditionnelle est nécessaire.
- **Exemple:** - Déterminer le maximum de deux entiers A et B

```
max = (A > B) ? A : B ;
```

- Déterminer le maximum de deux trois entier A,B et C

```
max = (a > b) ? (a > c ? a : c) : (b > c ? b : c);
```

Le langage C: Le sélection multiple (1/2)

- La **sélection multiple switch** en langage C permet de choisir une option parmi plusieurs cas en fonction de la **valeur d'une expression** donnée de type **entier** ou **caractère** *(int, long, char)*.

- Sa syntaxe est la suivante:

```
switch (expression) {
   case valeur1: ….. /* Code exécuté si  expression = valeur1 */
                 break;
   case valeur2:  …… /* Code exécuté si  expression = valeur2 */
                  break;
   …..
   default: …. /* Code exécuté si si aucune des valeurs ne correspond */
}
```

Le langage C: Le sélection multiple (2/2)

- **Exemple**: Déterminer si un entier A est 0, 1 ou autre

```
int A;
.....
switch (A)
        {  case  0 :    printf(" zero ") ; break ;
           case  1 :    printf(" un ") ; break ;
           default :    printf(" ni zero ni un ") ;
        }
```

- **Exemple**: Déterminer si un caractère E est voyelle ou consonne

```
char E;     scanf ("%c",&E);
switch (E)
        {  case  'a' :
           case  'e' :
           ……..        /* autres case des voyelles */
           case  'y' : printf (" voyelle ") ; break ;
           case  'b' :
           ……..        /* autres case des consonnes */
           case 'z' : printf (" consonne ") ; break ;
           default  : printf (" ni voyelle ni consonne ") ;
        }
```