Informatique

Sequence 1 : Premiers pas

TP 1.1

v(None)

IUT d'Annecy, 9 rue de l'Arc en Ciel, 74940 Annecy

Hello World

Sommaire

1 Introdution

Une structure de contrôle permet de modifier le flux d'exécution d'un programme en fonction de conditions ou de répétitions. Nous avons déjà vu les structures conditionnelles `if`, `else if` et `else` dans le TP précédent. Dans ce TP, nous allons explorer une autre structure conditionnelle : `switch case`.

La structure `switch case`

La structure `switch case` est utilisée pour sélectionner l'une parmi plusieurs options basées sur la valeur d'une expression. Elle est particulièrement utile lorsque vous avez de nombreuses conditions basées sur la même variable. Voici la syntaxe de base d'une structure `switch case` en C :

```
switch (expression) {
    case valeur1:
        // Code à exécuter si expression == valeur1
        break;
    case valeur2:
        // Code à exécuter si expression == valeur2
        break;
    // Vous pouvez avoir autant de cases que nécessaire
    default:
        // Code à exécuter si aucune des valeurs ne correspond
}
```

- `expression` est évaluée une fois et comparée aux valeurs dans chaque `case`.
- Si une correspondance est trouvée, le code associé à ce `case` est exécuté.
- Le mot-clé `break` est utilisé pour sortir de la structure `switch` après l'exécution d'un `case`. Sans `break`, l'exécution continue dans les `case` suivants (ce comportement est appelé "fall-through").
- Le `default` est optionnel et s'exécute si aucune des valeurs ne correspond.

1.1 Exercice 0 : Exemple de `switch case`

Manipulation 1 : Jours de la semaine

Compléter l'exemple suivant de `switch case` pour afficher le jour de la semaine en fonction d'un numéro (1 pour lundi, 2 pour mardi, etc.) :

```
#include <cs50.h>
#include <stdio.h>
int main() {
    int jour; // Variable pour stocker le numéro du jour
    jour = get_int("Entrez un numéro de jour (1-7) : ");

    switch (jour) {
        case 1:
            printf("Lundi\n");
            break;
        case 2:
            // Compléter pour les autres jours
        default:
            printf("Numéro de jour invalide. Veuillez entrer un nombre entre 1 et 7.\n");
    }
}
```

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp2/semaine
```

Le type `char`

Dans cet exercice, vous aurez besoin de demander un caractère simple. Le type le plus adapté est le type `char`. C'est un type qui peut contenir un caractère codé sur 1 octet (ASCII)

Attention : En langage C, un caractère s'entoure d'apostrophes `' '` et non avec des guillements. *Exemple :* `char caractere = 'A'`

2 Convertisseur de température - Echauffement !

On souhaite créer un programme permettant de convertir une température en degrés Celsius en degrés Fahrenheit et vice versa.

On donne la formule de conversion suivante :

- Pour convertir des degrés Celsius (C) en degrés Fahrenheit (F) : F = C * 9/5 + 32

Donner la formule de conversion inverse (Fahrenheit vers Celsius).

Manipulation 2 : Conversion de température

Le programme affichera un menu permettant à l'utilisateur de choisir entre deux options :

- F : Convertir Celsius en Fahrenheit
- C : Convertir Fahrenheit en Celsius

L'utilisateur entrera son choix (F ou C) puis appuiera sur Entrée. Ensuite, le programme demandera la température à convertir. Le programme effectuera la conversion et affichera le résultat.
Attention : Le choix de l'opération doit être fait avec un `switch case`. Structure `if` interdite.
La dernière sortie doit être de la forme : '`<temperature_utilisateur> <unite_utilisateur> = <temperature_convertie> <unite_convertie>`
Exemple d'exécution :

```
Choisissez l'opération (F pour Celsius->Fahrenheit, C pour Fahrenheit->Celsius) :
F
Entrez la température en degrés Celsius :
25
25.00 C = 77.00 F
```

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp2/temperatures
```

3 Calculatrice - Entrainement !

On souhaite créer une calculatrice basique qui effectue les opérations addition, soustraction, multiplication et division.

3.1 Niveau 1 : Opérations

Manipulation 3 : Calculatrice

Dossier : 2_calculatrice
Écrire un programme qui demande à l'utilisateur un nombre, une opération (addition, soustraction, multiplication, division), puis un second nombre. **L'utilisateur appuiera sur Entrée après chaque nombre/opération**.
Pour choisir l'opération, l'utilisateur entrera `+`, `-`, `*`ou `/` respectivement.
Le programme affichera alors l'opération et son résultat sur une ligne

Entrée 1	Entrée 2	Entrée 3	Sortie
`<n1>`	`<operation>`	`<n2>`	`<n1> <operation> <n2> = <resultat>`

Avec `<n1>` et `<n2>` les deux nombres entrés par l'utilisateur, `<operation>` l'opération choisie et `<resultat>` le résultat de l'opération. **Attention :** Le choix de l'opéaration doit être fait avec un `switch case`. Structure `if` interdite. Exemple :

```
Bonjour, entrez votre opération :
3
*
2
Merci. Voici le résultat de votre opération :
3 * 2 = 6
```

check50 IUT-GEII-Annecy/exercices/2025/info1/tp2/calculatrice/niveau1

3.2 Niveau 2 : Gestion des erreurs

Manipulation 4 : Gestion des erreurs

- ☐ Ajouter la gestion des erreurs dans le programme de la calculatrice
  - ◇ Si l'utilisateur entre une opération invalide, le programme doit afficher `ERREUR : Opération invalide` et terminer.
  - ◇ Si l'utilisateur entre une division par zéro, le programme doit afficher `ERREUR : Division par zéro` et terminer.

L'utilisation de `if` est autorisé pour cette partie.

- ☐ Après avoir vérifié manuellement votre programme, faire valider par l'enseignant.

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp2/calculatrice/niveau2
```

4 Validateur de date

4.1 Niveau 1 — Valider une date (jour/mois/année)

Manipulation 5 : Validateur de date

Dossier : `3_dates` Écrire un programme qui lit trois entiers dans cet ordre : **jour**, **mois**, **année** (l'utilisateur appuie sur Entrée après chaque saisie).
Le programme doit **vérifier** si la date est **valide** en tenant compte des différentes longueurs de mois **et** des **années bissextiles**.

Contraintes / attentes :

- Utiliser un `switch (mois)` pour déterminer **le nombre de jours** dans le mois (`maxJours`).
- Penser aux cas particuliers :
  - ◇ Le `mois` doit être valide
  - ◇ Le `jour` doit exister (attention aux années bixestiles)
  - ◇
- **Affichages attendus :**
  - ◇ Si la date est valide : `Date valide`
  - ◇ Sinon : `ERREUR : Date invalide.`

Rappels utiles :

- **Année bissextile** :

Une année est bissextile si l'une de ces deux conditions est remplie :

- L'année est un mutiple de 400
- L'année est un multiple de 4 mais pas un multiple de 100.

`(année % 400 == 0) || (année % 4 == 0 && année % 100 != 0)`

- **Longueur des mois** (hors février) :
  - ◇ 31 jours : 1, 3, 5, 7, 8, 10, 12
  - ◇ 30 jours : 4, 6, 9, 11

Exemples d'exécution :

```
Entrez le jour :
29
Entrez le mois :
2
Entrez l'année :
2024
Date valide
```

```
Entrez le jour :
31
Entrez le mois :
4
Entrez l'année :
2025
ERREUR : Date invalide.
```

Types conseillés : `int` pour jour, mois, année. **Pas de boucles ni de tableaux** nécessaires pour ce niveau.

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp2/date/niveau1
```

Structure possible (pseudo-code)

1) Lire `jour`, `mois`, `annee` 2) `Si mois n'est pas entre 1 et 12` → invalide 3) `switch (mois)` → donner `maxJours` (28 par défaut pour février) 4) `Si l'année est bissextile` → `maxJours = 29` 5) `Si le jour est plus grand que maxJours (ou inférieur à 1)` → invalide, sinon valide

4.2 Niveau 2 — Ajouter la saison (météorologique)

Manipulation 6 : Saison (en plus de la validation)

Reprendre le programme du **Niveau 1**.
S'il détecte une **date valide**, il doit **afficher** la **saison météorologique** correspondante en **France** :

- **Printemps** : du **21 mars** au **20 juin**
- **Été** : du **21 juin** au **20 septembre**
- **Automne** : du **21 septembre** au **20 décembre**
- **Hiver** : du **21 décembre** au **20 mars**

- **Affichages attendus :**
  - ◇ Si la date est valide :

`Date valide` (à la ligne suivante) `Saison : <saison>`

- Sinon : `ERREUR : Date invalide`

Exemples d'exécution :

```
Entrez le jour :
15
Entrez le mois :
6
Entrez l'année :
2025
Date valide
Saison : été
```

```
Entrez le jour :
1
Entrez le mois :
12
Entrez l'année :
2025
Date valide
Saison : Hiver
```

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp2/date/niveau2
```