Informatique

Sequence 1 : Premiers pas

TP 1.1

v2025-09-05

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

Hello World

Ce premier TP vous guide dans l'écriture de vos premiers programmes en langage C. Les premiers sont très guidés puis les derniers vous demanderont plus de réflexion.

Chaque *cahier des charges* présenté dans ce TP est associé à une commande permettant de faire vérifier la validité de votre programme par l'ordinateur.

Sommaire

1 Environnement de Trvail

1.1 Première configuration

Manipulation 1 : Créer un compte GitHub

- ☐ Si vous n'avez pas encore de compte GitHub, créez-en un :
  - ◇ **Nom d'utilisateur :** `<p>-<nom>` avec p, la première lettre de votre prénom, puis votre nom.
    - ○ Exemple : j-doe pour John Doe
  - ◇ **Adresse Mail :** `<adresse email académique>`
  - ☐ Lien pour inscription : GitHub.com.
- ☐ Rejoignez le cours du semestre 1, en cliquant sur le lien correspondant à votre groupe.

Groupe	Lien
A	Groupe A
B	Groupe B
C	Groupe C
D	Groupe D

- ☐ Lancez l'IDE, pour cela :
  - ☐ Rendez-vous sur la page de cs50.dev.
  - ☐ Cliquez sur `Log in`. *La première ouverture peut prendre quelques minutes*

GitHub ? Qu'est-ce que c'est ?

GitHub est une plateforme de développement collaboratif qui permet de stocker et de partager du code. Elle permet notamment le versionning *(Pouvoir revenir à toutes les versions précédentes du code)*, le suivi des modifications et facilite grandement la collaboration. C'est un outil incontournable dans le monde professionnel du développement logiciel. Pour ce premier semestre, l'utilisation de GitHub est invisible pour vous. Elle se fait via les commande de cs50

1.2 Le terminal

1.2.1 Présentation

En bas de la fenêtre de vsCode, vous trouverez un terminal. C'est un outil qui vous permet d'interagir avec votre système d'exploitation en utilisant des commandes textuelles.

Liste des commandes utiles

Voici quelques commandes de base que vous utiliserez fréquemment :

- `clear` : nettoie l'affichage dans le terminal

Les éléments entre chevrons <>, sont à remplacer en fonction du contexte

- `cd <nom_du_dossier>` : change le répertoire courant pour `<nom_du_dossier>`.
  - ◇ `cd ..` permet de remonter dans le dossier parent
- `code <nom_du_fichier>` permet d'ouvrir le fichier `<nom_du_fichier>` en le créant si besoin
- `cp <source> <destination>` : copie un fichier ou dossier de `<source>` à `<destination>`.
- `ls` : liste les fichiers et dossiers dans le répertoire courant.
- `mv <source> <destination>` : déplace ou renomme un fichier ou dossier de `<source>` à `<destination>`.
- `mkdir <nom_du_dossier>` : crée un nouveau dossier nommé `<nom_du_dossier>`.

- `rm <nom_du_fichier>` : supprime le fichier nommé `<nom_du_fichier>`.
- `rmdir <nom_du_dossier>` : supprime un dossier nommé `<nom_du_dossier>` (le dossier doit être vide).

L'autocomplétion dans le terminal

Bien souvent, le terminal peut deviner ce que vous voulez faire à partir du début de la commande, à l'aide de la touche `Tab`.

Manipulation 2 : On s'exerce !

- ☐ Exécuter les lignes suivantes, **les unes après les autres**.
  - ☐ Pour chacune, écrire ce que fait la commande.

```
mkdir monPremierDossier
ls
cd monPremierDossier
mkdir sous-dossier1 sous-dossier2
ls
cd ..
ls
cd monPremierDossier/sous-dossier1
cd ../sous-dossier2
cd ..
mv sous-dossier1 nouveau-nom
ls
cd
rm monPremierDossier
rm -r monPremierdossier
```

A vous de jouer !!

- ☐ Créer un dossier portant votre nom, à l'intérieur duquel vous créerez deux dossiers : un à votre prenom, l'autre tp1
- ☐ Faire vérifier par l'enseignant

Informations importantes à propos du terminal :

- Certains raccourcis ne sont pas les même qu'habituellement
  - ◇ `Ctrl-Maj-C` pour copier
  - ◇ `Maj-INSER` pour coller
- Les commandes ne répondent souvent rien lorsqu'elle réussissent
- Il est impossible de bouger le curseur avec la souris. Il faut utiliser les flèches

1.3 Téléchargement du dossier de TP

Prêts ? Allons-y ! Récupérons les fichiers pour commencer à coder !

Manipulation 3 : Premières commandes dans le terminal

```
wget https://github.com/IUT-GEII-Annecy/squelettes/releases/download/branch-2025/tp1.zip
unzip tp1.zip
rm tp1.zip
ls
cd tp1/
code hello.c
```

- ☐ Télécharger les fichiers du tp1 :
  - ☐ Copier la commande suivante puis la coller la dans le terminal `Maj-INSER`. Appuyer sur Entrée

```
wget https://github.com/IUT-GEII-Annecy/squelettes/releases/download/branch-2025/tp1.zip
```

- ☐ Décompresser le dossier et supprimer le *.zip*
  - ☐ **(Essayer l'autocomplétion en tapant `unzip t` puis en appuyant sur `Tab`)** :

    ```
    unzip tp1.zip
    ```

  - ☐ Supprimer le fichier `tp1.zip` :

    ```
    rm tp1.zip
    ```

  - ☐ Répondre `y` s'il demande confirmation
- ☐ Vous pouvez afficher la liste des fichiers et dossiers avec la commande `ls`.

  ```
  ls
  ls tp1/
  ```

- ☐ Naviguer dans le dossier `tp1/0_hello`

  ```
  cd tp1/0_hello
  ```

- ☐ Créer un fichier `hello.c` :

  ```
  code hello.c
  ```

  - ☐ Un nouvel onglet devrait s'ouvrir dans vsCode avec un fichier vide nommé `hello.c`.
    - ◇ *Vous pouvez également créer un fichier en cliquant sur l'icône "Nouveau fichier" dans l'explorateur de fichiers à gauche.*

2 Votre premier programme : Hello World

2.1 Hello World en C

Manipulation 4 : Hello World

Trois étapes sont nécessaires pour créer un Hello World en langage C :

1. Inclure la bibliothèque standard d'entrée/sortie (`stdio.h`). -> Ecrire `#include <stdio.h>` en haut du fichier
2. Définir la fonction principale `main`. -> `int main(void)  ...`
3. Utiliser la fonction `printf` à l'intérieur de la fonction main pour afficher le message.

   ```
   printf("Hello, World!\n");
   ```

4. Ajouter `return 0;` à la fin pour indiquer que le programme est terminé.

Votre programme devra donc ressembler à ça :

```c
#include <stdio.h>

int main(void)
{
    printf("Hello, World!\n");
    return 0;
}
```

Besoin d'aide ?

CS50 Duck Debugger est là pour vous aider ! Cliquer sur l'icône en forme de canard dans la colonne de gauche pour ouvrir l'extension. Vous pouvez lui demander "En Français" à la fin de votre prompt pour obtenir de l'aide en français. S'il vous dit qu'il ne parle qu'anglais, insistez, il finit souvent par abdiquer.

Manipulation 5 : Compiler et exécuter le programme

- ☐ Dans le terminal, assurez-vous d'être dans le dossier `TP1/0_hello` (où se trouve votre fichier `hello.c` – sinon, utilisez la commande `cd` pour naviguer jusqu'à ce dossier).
- ☐ Compiler le programme avec la commande `make hello`.
- ☐ Exécuter le programme compilé avec la commande `./hello`. Vous devriez voir "Hello, World!" s'afficher dans le terminal.

```
tp1/0_hello/ $ ls
hello.c
tp1/0_hello/ $ make hello
tp1/0_hello/ $
tp1/0_hello/ $ ./hello
Hello, World!
tp1/0_hello/ $
```

- Si vous voyez le message, félicitations ! Vous venez d'écrire votre premier programme en C. **Dans le cas contraire, demandez de l'aide à l'enseignant, à vos camarades ou au CS50 Duck Debugger.**

3 Notre première variable

Nous allons à présent utiliser nos premières variables. Pour cela, commençons simplement avec l'exemple du cours et le cahier des charges suivant :

Cahier des charges 1 : Afficher le nom de l'utilisateur

- Le programme doit demander le nom de l'utilisateur avec le message de votre choix.
- Le programme doit afficher "Hello, <nom>" où <nom> est le nom entré par l'utilisateur.

Sortie attendue :

Entrée	Affichage attendue
`<nom>`	Hello, `<nom>`

Exemples de tests :

Entrée	Affichage attendue
Alice	Hello, Alice

Pour lancer les tests :

```
check50 iut-geii-annecy/exercices/2025/info1/tp1/0_hello/variable
```

Le cahier des charges mentionne la sortie attendue. **Votre programme doit ABSOLUMENT afficher la sortie attendue pour être correct.**

Manipulation 6 : Afficher le nom de l'utilisateur

Pour répondre à ce cahier des charges, vous devez réaliser les 4 prochaines étapes. A vous de trouver où effectuer chacune des modifications.

- ☐ Inclure la bibliothèque `cs50.h` pour utiliser la fonction `get_string`.

```
#include <cs50.h>
```

- ☐ Déclarer une variable de type `string` pour stocker le nom de l'utilisateur.

```
string nom;
```

- ☐ Utiliser `get_string` pour obtenir le nom de l'utilisateur.

```
nom = get_string("Entrez votre nom : ");
```

- ☐ Modifier `printf` pour afficher le message "Hello , [nom] !" où [nom] est le nom entré par l'utilisateur.

Manipulation 7 : Vérification

A l'aide de la fiche "Aide Mémoire" :

- ☐ Vérifier la mise en page de votre code (`style50`)
- ☐ Faire passer les tests automatiques (`check50`)

4 Fix Me : Cherchez l'erreur

Oups ! Le fichier fixme.c contient plein d'erreurs qui rendent la compilation impossible. A vous de les corriger !

Manipulation 8 : Corrige le fichier

- ☐ Se rendre dans le dossier `1_fixme`
  - ☐ Soit en revenant d'abord à la racine avec `cd` seul

```
cd
cd tp1/1_fixme
```

  - ☐ Soit directement depuis `0_hello` :

```
cd ../1_fixme
```

  - ☐ Dans les deux cas, votre terminal doit maintenant commencer par

```
tp1/1_fixme $
```

- ☐ Tenter de compiler `make fixme` -> Des erreurs apparaissent
- ☐ Corriger le fichier `fixme.c`
- ☐ Compiler, puis tester votre code

```
make fixme
./fixme
```

- ☐ **Si tout fonctionne**, exécuter

```
check50 iut-geii-annecy/exercices/2025/info1/tp1/1_fixme/fixme
```

5 Parlons surface !

5.1 Premier calculs : aires et périmètres

Manipulation 9 : Géométrie - Niveau1

- ☐ Rendez-vous dans le dossier `tp1/2_geometrie`
- ☐ Compléter le programme `rectangle.c` pour qu'il respecte le cahier des charges Rectangle - Niveau 1
- ☐ Compléter le programme `cercle.c` pour qu'il respecte le cahier des charges Cercle - Niveau 1

5.1.1 Aire d'un rectangle

Cahier des charges 2 : Aire d'un rectangle

- Le programme doit demander la largeur et la longueur du rectangle
- Le programme calcule alors l'aire du rectangle.

Sortie attendue :

Entrée 1	Entrée 2	Affichage attendue
`<largeur>`	`<longueur>`	Aire : `<aire>`

Avec `<aire>` L'aire du rectangle **arrondie au centième**.

Exemples de tests :

Entrée 1	Entrée 2	Affichage attendue
10	13	Aire : 130.00
5.5	3	Aire : 16.50

Check :

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp1/2_geometrie/rectangle/niveau1
```

5.1.2 Aire et périmètre d'un cercle

Cahier des charges 3 : Cercle Niveau 1

- Le programme doit demander le rayon du cercle puis afficher l'aire et le périmètre.

Avec [aire] et [perimetre] respectivement l'aire et le périmètre du cercle.

Sortie attendue :

Entrée	Affichage attendue
`<rayon>`	Aire : `<aire>` Périmètre : `<perimetre>`

Avec `<aire>` et `<perimetre>` **arrondis au centième**.

Exemples de tests :

Entrée 1	Affichage attendue
10	Aire : 314.16 Périmètre : 62.83

Check :

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp1/2_geometrie/cercle/niveau1
```

6 Nos premières conditions !

Si vous donnez des valeurs négatives en entrée des programmes précédents, ils vous répondront une surface négative. Testez-le !

Tout le monde sait qu'une surface est toujours positive... Alors interdisons au programme de répondre n'importe quoi

Cahier des charges 4 : Valeurs négatives interdites

- Même cahier des charges que précédemment sauf que :
- Après que l'utilisateur a rentré **toutes les longueurs** :
  - ◇ Si l'une ou l'autre des valeur est négative, le programme répondra

```
    ERREUR : Valeurs negatives interdites.
```

Checks :

```
IUT-GEII-Annecy/exercices/2025/info1/tp1/2_geometrie/cercle/niveau2
IUT-GEII-Annecy/exercices/2025/info1/tp1/2_geometrie/rectangle/niveau2
IUT-GEII-Annecy/exercices/2025/info1/tp1/2_geometrie/geometrie/all
```

Manipulation 10 : Valeurs négatives impossibles

- ☐ Compléter les programmes *cercle.c* et *rectangle.c* pour remplir le cahier des charges.
- ☐ Tester vos programmes à la main

7 Êtes-vous majeur ?

Exerçons-nous avec quelques conditions simple

Manipulation 11 : Majeur ou mineur

- ☐ aller dans le dossier `tp1/3_age` puis ouvrir age.c

```
cd
cd tp1/3_age
code age.c
```

- ☐ Compléter le programme pour les Niveaux 1 et 2

7.1 Niveau 1

Cahier des charges 5 : Age - Niveau 1

- ☐ Le programme demande quel age a l'utilisateur
- ☐ Le programme répond alors selon la logique suivante :
  - ☐ Sinon, si l'utilisateur a strictement moins de 18 ans : `Vous êtes un mineur.`
  - ☐ Sinon `Vous êtes un adulte.`

Checks :

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp1/3_age/niveau1
```

7.2 Niveau 2

Cahier des charges 6 : Age - Niveau 2

- ☐ Le programme demande quel age a l'utilisateur
- ☐ Le programme répond alors selon la logique suivante :
  - ☐ Si l'utilisateur a strictement moins de 12 ans : `Vous êtes un enfant.`
  - ☐ Sinon, si l'utilisateur a strictement moins de 18 ans : `Vous êtes un mineur.`
  - ☐ Sinon, si l'utilisateur a strictement moins de 60 ans : `Vous êtes un adulte.`
  - ☐ Sinon, si l'utilisateur a strictement moins de 120 ans : `Vous êtes un sénior.`
  - ☐ Sinon : `Vous êtes un menteur.`

Checks :

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp1/3_age/niveau2
```

8 La pause Tacos

8.1 Niveau 1

Cahier des charges 7 : Pause Tacos - Niveau 1

Votre programme tient un Tacos dont vous devez inventer le nom. Il reçoit un client, qui lui commande un certain nombre de tacos et de kebab et votre programme lui dit annonce alors le montant total de la commande

- ☐ Le programme Affiche d'abord une phrase de bienvenue

```
Bonjour, bienvenu chez <nom_du_tacos>
```

- ☐ Le programme demande alors le nombre de Tacos, puis le nombre de Kebab voulu
- ☐ Le programme écrit alors le prix total de la commande, arrondi au centième.
- ☐ Le programme écrit ensuite le message de remerciement

```
Montant total : <montant_total> euros
Merci pour votre commande chez <nom_du_tacos>
```

Produit	Prix
Tacos	6,30 €
Kebab	5,50 €

Checks :

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp1/3_tacos/niveau1
```

Exemple de sortie pour cet exercice

```
tp1/3_tacos/ $ ./tacos
Bonjour, bienvenu chez Tacocos
Combien voulez-vous de Tacos ? 5
Combien voulez-vous de Kebab ? 2
Montant total : 42.50 euros
Merci pour votre commande chez Tacocos
tp1/3_tacos/ $ logout
```

Fiche méthode

Pour ce programme, comme pour beaucoup, posez-vous ces questions, dans l'ordre.

1. "Si j'étais à la place du programme, comment je ferai ?"
   - Vous pourrez en déduire les étapes de l'algorithme
2. "Si j'étais vraiment très mauvais en mémorisation, qu'est-ce que je devrais écrire ?"
   - Vous pourrez en déduire les variables du programme
   - ☐ Lesquelles sont des entiers `int` ?
   - ☐ Lesquelles sont des nombres à virgule `float` ?
   - ☐ Autres ?
3. "Y a-t-il des 'cas bizarre' ?"

8.2 Niveau 2

Même chose avec gestion des stocks

Cahier des charges 8 : Pause Tacos - Niveau 2 - Gestion des stocks

- ☐ Même cahier des charges que le niveau 1 auquel s'ajoute :
- ☐ Le restaurant a un stock limité.
  - ☐ Après avoir demandé le nombre de tacos et de kebab :
    - ☐ Si le client demande unnombre négatif de l'un, l'autre ou les deux :
      - ☐ affiche le message

```
ERREUR : Valeurs négatives interdites.
```

      - ☐ Arrête le programme sans autre affichage -> `return 1;`
    - ☐ Si les stocks sont insuffisant, le programme
      - ☐ affiche un message selon le tableau ci-dessous
      - ☐ affiche ensuite le message de remerciement du Niveau 1
    - ☐ Sinon, pas de changement par rapport au niveau 1

Produit hors stock	Sortie attendue
Kebab	`Désolé, nous n'avons pas assez de Kebab`
Tacos	`Désolé, nous n'avons pas assez de Tacos`
Tacos et Kebab	`Désolé, nous n'avons pas assez de Tacos, ni de Kebab`

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp1/3_tacos/niveau2
```

8.3 Bonus : Niveau 3

Cahier des charges 9 : Pause Tacos - Niveau 3 - Réduction

- ☐ Même cahier des charges que précédemment
- ☐ Si le client commande plus de 5 articles au total, une réduction de 10% est appliquée sur le montant total de la commande
- ☐ Les sorties attendues sont identiques aux cahiers des charges précédents

```
check50 IUT-GEII-Annecy/exercices/2025/info1/tp1/3_tacos/niveau3
```

9 Conclusion (A faire même si vous n'avez pas fini)

Manipulation 12 : A rendre

- ☐ Placer vous dans le dossier tp1
- ☐ Exécuter la commande suivante pour soummettre votre TP, pour en permettre le suivi.

```
submit50 IUT-GEII-Annecy/exercices/2025/info1/tp1/progression
```

Félicitations !

Vous venez de terminer votre premier TP de C. Vous avez appris à utiliser le terminal, à écrire un programme simple en C, à compiler et exécuter ce programme, et à soumettre votre travail sur cs50.