TP n°1 - Premiers programmes en C

Notions abordées

- programme : structure, compilation, exécution
- fonctions : définition, appel, jeu de test (avec `assert`)
- affichage et saisie (avec `printf` et `scanf`)
- structure conditionnelle (avec `if`)

Avant de commencer, vous devez **impérativement** configurer votre session Debian sur une des machines de la salle de TP, **même si vous comptez travailler sur votre machine personnelle**. Notez de quelle machine il s'agit, car votre session n'existera que sur cette machine, elle ne sera pas accessible en réseau depuis les autres machines.

→ Connectez-vous sur votre session sur le réseau du lycée (sous Windows) grâce aux identifiants donnés le jour de la rentrée.

→ Lancez le logiciel VMware en double-cliquant sur l'icône (jaune) présente sur votre bureau.

→ Dans les fenêtres qui s'ouvrent (la première fois seulement) cliquez sur Continue, puis Finish.

→ Une fois VMware ouvert, cliquez sur Open et choisissez la machine virtuelle Debian, c'est-à-dire le fichier C:/VMWARE/Debian_9.13_64 bits/Debian 9. x 64-bit.vmx

→ Une fois Debian installé, une machine virtuelle Debian apparaît dans VMware, cliquez dessus, puis cliquez sur Play virtual machine.

→ Une fois Debian ouvert, appuyez sur la touche VerrNum, **même si la touche est déjà allumée**.

→ Connectez-vous avec le login groupe2 et le mot de passe donné au tableau.

→ Réglez la taille de l'écran à 1440×900 dans Système/Préférences/Matériel/

→ Ouvrez Terminal MATE (via Applications/Outils système) pour changer votre mot de passe :

- tapez la commande `passwd` ;
- tapez votre ancien mot de passe, *i.e.* celui avec lequel vous venez de vous connecter *rien n'apparaît c'est normal : sur linux on entre souvent ses mots de passe à l'aveugle* ;
- tapez votre nouveau mot de passe, deux fois, comme demandé.

→ Ouvrez FirefoxESR pour configurer le proxy :

- ouvrez le menu en haut à droite, cliquez sur Préférences en bas de la liste ;
- allez en bas de page, section Paramètres réseau, cliquez sur Paramètres ;
- cochez Configuration manuelle du proxy, puis entrez 10.255.1.204 dans le champ Proxy HTTP, et 3128 dans le champ port ;
- cochez Utiliser ce serveur proxy pour tous les protocoles ;
- cliquez sur Ok, puis fermez le navigateur.

→ Ouvrez le navigateur de fichiers en cliquant sur l'icône Dossier personnel qui se trouve sur le bureau. Créez les répertoires nécessaires jusqu'à obtenir un répertoire dont le chemin d'accès finit par `MP2I/info/TPs/TP1`.
Vous stockerez tous les fichiers concernant ce premier TP dans ce répertoire.
Je vous laisse deviner ce qu'il faudra faire à la prochaine séance... ou lorsque vous travaillerez sur un DM.

Vocabulaire : Dossier ≡ répertoire (Folder ≡ directory en anglais)

Optionnel

Cette section décrit le réglage de certains paramètres sous Debian qui pourraient vous rendre l'utilisation plus facile. À faire sur la machine que vous comptez utiliser en priorité, et **après le TP si vous n'êtes pas en avance**.

La barre du haut, appelée **Tableau de bord**

Vous pouvez y ajouter des icônes pour lancer vos applications favorites. C'est l'équivalent des programmes épinglés à la barre des tâches sous Windows. Pour cela parcourez le menu Applications, et faites clic droit/Ajouter au tableau de bord sur les applications que vous voulez ajouter. Je conseille par exemple de le faire pour :

- Accessoires/Pluma pour l'éditeur de texte (équivalent de Gedit)
- Accessoires/Calculator pour une calculatrice
- Internet/FirefoxESR pour un navigateur internet
- Outils système/Caja pour un navigateur de fichiers
- Outils système/Terminal MATE pour un terminal

Vous pouvez aussi ajouter des choses en faisant un clic droit/Ajouter au tableau de bord sur cette barre du haut. Je conseille par exemple d'ajouter :

- Afficher le bureau
- Fermer la session
- Horloge

Le bureau

Vous pouvez y ajouter des icônes pour lancer vos applications favorites. C'est l'équivalent des raccourcis sur votre bureau Windows. Pour cela parcourez le menu Applications, et faites clic droit/Ajouter au bureau sur les applications que vous voulez ajouter. Vous pouvez aussi y créer des raccourcis vers des dossiers.

L'éditeur de texte **Pluma**

Vous pouvez modifier des paramètres d'affichage en allant dans Édition/Préférences. Quelques options sont particulièrement intéressantes :

- Affichage/afficher les numéros de lignes
- Affichage/surligner les parenthèses correspondantes
- Éditeur/activer l'indentation automatique

Premiers pas

Exercice 1 Exécuter un programme déjà compilé

Question 1 🖳
Allez sur la page de la MP2I sur cahier de prépa, puis sur la page des Documents à télécharger de la section informatique. `https://cahier-de-prepa.fr/mp2i-fermat/docs?Info`
Dans le répertoire `TP1` se trouve un fichier nommé `exo1`. Téléchargez-le puis placez-le dans votre répertoire `TP1`.

Question 2 🖳
Ouvrez un terminal dans votre répertoire `TP1`. Il existe pour cela deux méthodes : soit vous ouvrez Fichiers, vous naviguez jusqu'au dit répertoire, puis vous faites Ouvrir dans un terminal grâce à un clic droit; soit vous ouvrez Terminal puis vous naviguez jusqu'au dit répertoire grâce à la commande `cd`.

Question 3 🖳 ✍
Donnez au fichier **exo1** le droit d'être exécuté en tapant `chmod 755 exo1` dans le terminal.
Exécutez le fichier **exo1**. Décrivez ce que vous observez. Est-ce la même chose à chaque exécution ?

Exercice 2 Compiler puis exécuter un programme déjà écrit

Question 1 🖳
Téléchargez le fichier nommé `exo2.c` disponible sur cahier de prépa (*Cf.* Question **?? ??**) et placez-le dans votre répertoire `TP1`.

Question 2 🖳
Compilez le fichier `exo2.c` en un fichier exécutable nommé `exo2`.

Question 3 🖳 ✍
Exécutez le fichier `exo2`. Décrivez ce que vous observez. Est-ce la même chose à chaque exécution ?

Question 4 🖳
Rectifiez le code de ce programme pour obtenir un affichage lisible, sans erreurs d'orthographe ni de calculs.
Exécutez-le pour vous assurer que le programme fonctionne.

Exercice 3 Écrire un programme avec affichage et saisie

Question 1 🖳
Écrivez dans un fichier `exo3.c` un programme qui affiche `"Saisissez un entier compris entre 0 et 100"` puis saute une ligne. Testez votre programme.

Question 2 🖳
Modifiez votre programme pour qu'il affiche qui affiche `"Saisissez un entier naturel compris`

`entre` $N_{\min}$ `et` $N_{\max}$`."` où $N_{\min}$ et $N_{\max}$ sont les valeurs de variables que vous définirez.
Testez votre programme, avec différentes valeurs pour $N_{\min}$ et $N_{\max}$.

Question 3 💻
Complétez le programme pour récupérer l'entier naturel saisi par l'utilisateur ou l'utilisatrice dans une variable nommée `n`. Ajoutez un affichage qui vous permette de déterminer si la valeur saisie a bien été enregistrée. Testez votre programme.

Exercice 4 Définir une fonction

Question 1 💻
Copiez le code de `exo3.c` dans un fichier `exo4.c`. Modifiez `exo4.c` afin d'isoler le code concernant la saisie dans une fonction nommée `saisie_entier`, qui prend en argument les bornes de l'intervalle de saisie à afficher, et qui retourne l'entier saisi. Testez votre fonction en exécutant plusieurs fois le programme. *La partie de votre programme concernant l'affichage post-saisie ne doit pas être inclue dans cette fonction, mais être laissée dans le* `main` *pour les tests.*

Question 2 💻
Créez une fonction `saisie_entier_verif`, qui prend en argument les bornes de l'intervalle de saisie à afficher, et qui retourne l'entier saisi seulement s'il est bien entre les bornes, et qui renvoie -1 sinon. Testez votre fonction en exécutant plusieurs fois le programme, y compris en saisissant des valeurs non comprises entre les bornes.

Question 3 💻
Créez une fonction `affiche_saisie_entier`, qui prend en argument les bornes de l'intervalle de saisie à afficher, et qui affiche l'entier saisi seulement s'il est bien entre les bornes, et qui affiche un message d'erreur sinon. Testez votre fonction en exécutant plusieurs fois le programme, y compris en saisissant des valeurs non comprises entre les bornes.

Question 4 💻 bonus
Si vous avez déjà vu les boucles while, vous pouvez faire une fonction `saisie_robuste` qui demande la saisie d'un entier tant que l'entier saisi n'est pas valide. Attention à ne pas demander l'impossible...

Question 5 💻 bonus
Si vous avez déjà vu cela, il peut-être intéressant de transformer le programme précédent pour qu'il prenne en argument les valeurs de N_min et N_max. Ces valeurs seront passées au programme en ligne de commande. Un jeu de test peut alors être proposé dans un script bash.

Exercices de base (à rendre)

Vous devez commenter et tester chaque fonction. Lorsque cela est possible, vous utiliserez la fonction `assert` vue en TD. Pour cela vous devez inclure la bibliothèque `assert.h`.
Les questions siglées ✍ sont à faire sur feuille, et sont susceptibles d'être ramassées. Les questions siglées 💻 de chaque exercice sont à rendre dans un seul fichier `exoN.c`. L'ensemble de ces fichiers sera à rassembler dans une archive nommée `TP1_NOM.zip` pour le rendu.

Exercice 5 Recoder les opérateurs booléens

Dans cet exercice, on cherche à recoder les opérateurs `&&`, `||` et `!`, vos fonctions ne devront donc **pas utiliser** ces opérateurs. Pour rappel, on donne ci-dessous les tables de vérité associées à ces opérateurs.

a	b	a&&b
false	false	false
false	true	false
true	false	false
true	true	true

a	b	a||b
false	false	false
false	true	true
true	false	true
true	true	true

a	!a
false	true
true	false

Question 1 🖳
Donner la définition d'une fonction `conjonction` qui prend en entrée deux booléens et qui renvoie leur conjonction.

Question 2 🖳
Donner la définition d'une fonction `disjonction` qui prend en entrée deux booléens et qui renvoie leur disjonction.

Question 3 🖳
Donner la définition d'une fonction `négation` qui prend en entrée un booléen et qui renvoie sa négation.

Question 4 🖳 bonus

Donner la définition d'une fonction `implication` qui prend en entrée deux booléens `a` et `b` et qui renvoie si l'implication `a`⇒`b` est vraie. Cette fonction devra obligatoirement faire appel aux fonctions précédentes, et ne devra pas utiliser d'opérateurs booléens.

a	b	a⇒b
false	false	true
false	true	true
true	false	false
true	true	true

Exercice 6 Distinguer les types de triangle

Question 1 🖳
Donner la définition d'une fonction `type_triangle` qui prend en argument trois entiers donnant les longueurs des côtés d'un triangle exprimées dans la même unité, et qui affiche `équilatéral` si le triangle l'est, `isocèle` si le triangle est isocèle mais pas équilatéral, et `scalène` sinon. *On supposera ici que les utilisateurs respectent votre spécification.*

Question 2 ✍
Pour vérifier que les longueurs a, b, c passées en argument de la fonction précédente correspondent bien à celles des trois côtés d'un même triangle, on propose de tester si

$$(a+b \geqslant c) \text{ et } (a+c \geqslant b) \text{ et } (b+c \geqslant a)$$

Montrer que cette condition est bien nécessaire et suffisante. *Montrer que c'est une condition nécessaire c'est montrer que les longueurs des côtés d'un triangle vérifient forcément cette condition. Montrer que c'est une condition suffisante revient à monter que tout triplet de longueurs (a, b, c) vérifiant cette condition correspond à un triangle.*

Question 3 ✍

Peut-on trouver une condition nécessaire plus simple que celle proposée à la question précédente ?

Question 4 💻
Donner la définition d'une fonction `est_triangle` qui prend en argument trois entiers donnant des longueurs exprimées dans la même unité, et qui retourne si ces longueurs peuvent être celles des trois côtés d'un même triangle.

Question 5 💻
Donner la définition d'une fonction `est_plat` qui prend en argument trois entiers donnant des longueurs exprimées dans la même unité, et qui retourne si ces longueurs sont celles des trois côtés d'un triangle plat.

Question 6 💻
Donner la définition d'une fonction `est_rectangle` qui prend en argument trois entiers donnant des longueurs exprimées dans la même unité, et qui retourne si ces longueurs sont celles des trois côtés d'un triangle rectangle.

Question 7 💻
Donner la définition d'une fonction `type_triangle_complet` qui prend en argument trois entiers donnant des longueurs exprimées dans la même unité, qui affiche un message d'erreur si ces longueurs ne sont celles des côtés d'aucun triangle, et qui affiche l'information la plus complète possible sur ce triangle sinon.

Exercice 7 Évaluer les polynômes

Question 1 💻
Donner la définition d'une fonction `affine` qui prend en deux coefficients entiers `a` et `b`, ainsi qu'un entier `x`, et qui retourne la valeur de la fonction $t \mapsto at + b$ en `x`.

Question 2 💻
Donner la définition d'une fonction `eval_poly3` qui permet d'évaluer un polynôme de degré au plus 3 à coefficients entiers en une valeur entière. Le corps de cette fonction ne fera apparaître aucune addition ni multiplication, mais présentera en revanche des appels à la fonction `affine`.

Question 3 ✍
Dire combien de multiplications et combien d'additions sont effectuées lors d'un appel à la fonction `eval_poly3`. Cela vous paraît-il efficace pour l'évaluation d'un polynôme de degré 3 ?

Pour aller plus loin

Exercice 8 Validation d'un numéro de sécurité sociale

Question 1 💻
Faire l'exercice 6 de la feuille d'exercices n°1, en particulier donnez la définition d'une fonction qui prend en entrée un numéro de sécurité sociale sous la forme d'un entier à 15 chiffres, et qui renvoie si ce numéro est valide ou non (au sens de la clé).
On supposera ici que les utilisateurs respectent votre spécification et donnent un entier à 15 chiffres.