Fonctions

1 Introduction

Exercice 1. Prototypes.

Parmi les prototypes de fonctions suivants, lesquels sont correctement déclarés ? Si une erreur est présente, explicitez-la et corrigez-la. N'hésitez pas à faire une recherche sur Internet concernant le type `void`.

1) `int func (int z);`
2) `float a (float a);`
3) `double f (int a, int b);`
4) `func ();`
5) `void toto (void var);`
6) `var func (char);`
7) `short a (int func);`

Exercice 2. Prototypes.

Pour chacun des prototypes mathématiques suivants, définissez, si possible un prototype de fonction C adéquat.

1) $f_1 : \mathbb{Z} \to \mathbb{Z}$
2) $f_2 : \mathbb{Q} \to \mathbb{N}$
3) $f_3 : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Q}$
4) $f_4 : \mathbb{Q}^2 \to \mathbb{Z}$
5) $f_5 : \mathbb{Z} \to \mathbb{Z} \times \mathbb{N}$

Pour chacun des prototypes de fonction C suivants, définissez, si possible un prototype de fonction mathématiques adéquat.

6) `int f1 (float a, float b);`
7) `void f2 (int a);`
8) `int f3 (void)`
9) `double f4 (int a, float b, float c);`
10) `char f5 (int a);`

Exercice 3. Addition.

Créez une fonction nommée `addition()`, prenant en entrée deux entiers et retournant la somme de ces deux entiers. Appelez cette fonction dans votre `main()` afin d'effectuer et d'afficher la somme de deux variables saisies par l'utilisateur. Vous pouvez également utiliser votre fonction générant un nombre aléatoire.

```c
#include<stdio.h>
#include<stdlib.h>

int addition(int a, int b)
{
    int add = a + b;

    return add;
}

int main()
{
    int add = addition(2,3);
    printf("%d\n", add);
    return 0;
}
```

2 Variables et fonctions

Exercice 4. La mémoire dans la peau.

Complétez le tableau suivant représentant l'état de la mémoire à la fin de l'exécution du code ci-dessous.

```
int power (int a, short b)
{
  short i;
  int res = 1;
  for (i=0; i<b; i++)
    res *= a;
  return res;
}

int main()
{
  int a = 2;
  short b = 3;
  int res;

  res = power (a, b);

  return 0;
}
```

Adresse	Nom	Valeur de la variable
0x120 0x121 0x122 0x123	a	2
0x124 0x125	b	3
0x126 0x127 0x128 0x129	res	~~?~~ 8
0x12A 0x12B 0x12C 0x12D	a	2
0x12E 0x12F	b	3
0x130 0x131	i	~~? 0 1 2~~ 3
0x132 0x133 0x134 0x135	res	~~1 2 4~~ 8
0x136 0x137 0x138 0x139 0x13A 0x13B 0x13C 0x13D 0x13E 0x13F		

Exercice 5. Multiplication.

Créez une fonction `multiplication()`, dont le rôle est d'effectuer la multiplication entre deux variables entières et de retourner le résultat. Utilisez cette fonction ainsi que la précédente dans votre `main()` afin d'obtenir et d'afficher les résultats des quatre opérations suivantes manipulant les entiers `a`, `b` et `c`.

1) $a \times b$;
2) $a \times (b + c)$;
3) $a \times b \times c$;
4) a^b, si $b > 0$;
5) $a!$.

Exercice 6. Génération de nombre aléatoire.
Créez la fonction `myRand()` prenant en paramètres deux entiers `min` et `max` et renvoyant un entier aléatoire entre `min` et `max`. Affichez le résultat depuis le `main()` afin de vérifier le résultat.

3 Compilation et fichier d'entête

Pour chacun des exercices de cette section, vous devez rédigez un fichier entête en plus du fichier source contenant la déclaration de la fonction. Vous réaliserez les tests de vos fonctions à partir d'un troisième fichier nommé `main.c`.

Exercice 7. Mini et maxi.
Écrivez une fonction `maximum()` prenant deux entiers et retournant le plus grand des deux. Utilisez-la afin d'afficher les variables `val1`, `val2` et `val3` de la plus grande à la plus petite.

Écrivez une fonction `minimum3()` qui prend trois entiers en entrée et retourne la plus petite valeur. Intégrez-la à votre programme pour afficher la plus petite des variables.

Exercice 8. Nombre miroir.
Une fonction miroir d'un entier est une fonction qui retourne l'entier symétrique de l'entier en entrée. Par exemple, le miroir de 1337 est 7331. Rédigez les fonctions suivantes.

1) `int Miror(int N)` : qui retourne le nombre miroir de l'entier `N`.
2) `int NbDigit(int N)` : qui retourne le nombre de chiffres du nombre `N`.
3) `int Palindromize(int N)` : qui retourne le nombre palindrome de l'entier `N`. On rappelle que le palindrome du nombre 1234 est 1234321.

Exercice 9. Équation du second degré.
Retour au lycée pour cet exercice ... Nous vous demandons de créer un programme qui résout une équation du second degré dans $\mathbb{R}$. Nous rappelons que pour résoudre l'équation $ax^2 + bx + c = 0$ avec $a, b, c \in \mathbb{R}$, il faut tout d'abord calculer la valeur du discriminant $\Delta = b^2 - 4ac$. Les solutions sont données par :

$$\begin{cases} \text{Si } \Delta > 0 \text{ alors} & x_1 = \frac{-b-\sqrt{\Delta}}{2a}\ ,\ x_2 = \frac{-b+\sqrt{\Delta}}{2a}. \\ \text{Si } \Delta = 0 \text{ alors} & x = \frac{-b}{2a}. \\ \text{Si } \Delta < 0 \text{ alors} & \text{pas de solution.} \end{cases}$$

Vous devez donc dans un premier temps demander à l'utilisateur de saisir les valeurs a, b et c avant d'afficher la solution. Vous devrez afficher la solution de deux façons différentes : la première sous forme de nombre réel (par exemple `x1 = 3.38 et x2 = -1.74`) et la deuxième sous forme de fraction (par exemple `x=(-2-rac(5))/14`). Le calcul du discriminant et l'affichage des solutions doit être réalisé au travers d'une fonction prenant en argument les trois réels `a`, `b` et `c`.

4 Point d'entrée du programme

Exercice 10. Arguments du `main`.
En vous inspirant du cours, rédigez **une** instruction C ainsi que l'instruction bash permettant de ...

1) afficher le nombre d'arguments présents sur la ligne de commande.
2) récupérer le prénom et le nom de l'utilisateur et les afficher.
3) récupérer l'année de naissance saisie par l'utilisateur et afficher son âge.

5 Exercices de synthèse

Exercice 11. Soirée arrosée.

Un étudiant de première année décide de réaliser un punch pour fêter son anniversaire. Un peu trop optimiste, il décide de vider une bouteille de 2L de rhum à 40 degrés dans 2L de jus de fruits. Ses amis le trouvant trop fort, et voulant faire la fête longtemps, ils décident de le diluer en buvant chaque heure 1L de punch, et en y ajoutant à la fin de l'heure 1L de jus de fruits. La soirée ayant commencé à 20h, quel est le degré d'alcool du punch à 23h30 ? à 00H15 ?

Exercice 12. Calcul de racine.

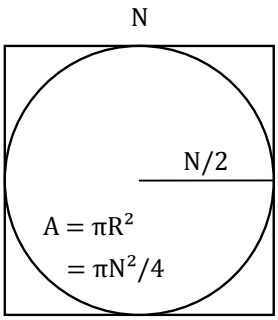
Continuons dans les mathématiques ! L'objectif est de calculer les N premiers termes d'une suite numérique. Voici la suite en question :

$$\text{Calcul de } \sqrt{A} : \begin{cases} u_0 & = c \\ u_{n+1} & = \frac{1}{2}\left(u_n + \frac{A}{u_n}\right) \end{cases}$$

1) Dans votre `main`, récupérez de la part de l'utilisateur les valeurs de A, c et n.
2) Créez la fonction `float racine (float A, float c, float n)` permettant d'approcher la valeur de $\sqrt{A}$ au rang n, et pour un A fixé.
3) Créez par la suite une fonction `float racineAuto (float A, float c, float s)` pour que le calcul s'arrête de lui même lorsque la différence entre u_n et u_{n+1} est inférieure au seuil s.

Exercice 13. Approximation de π.

De nombreuses méthodes existent afin d'approximer la valeur de π en utilisant une fonction ou un algorithme mathématique. Nous allons utiliser la méthode dite de Monte Carlo, basée sur le principe de placer un grand nombre de points aléatoirement dans un carré, et vérifier combien appartiennent au cercle inscrit dans ce carré. Afin de bien fixer les idées, le schéma ci-contre vous donne l'explication visuelle de l'approximation.



En posant $N = 2$, on retrouve $A = \pi$ car l'aire du cercle inclus dans un carré de dimension N est $A = \pi N^2/4$. L'idée est alors de placer des points aléatoirement dans le carré et de compter le nombre de points inscrits dans le cercle. En normalisant avec la surface du carré, il est alors possible de trouver une approximation convenable (et probabiliste) de la valeur π.

1) Dans un premier temps, codez la fonction `float TirageAleatoire (float b)` qui renvoie une valeur flottante aléatoire entre les bornes `-b` et `b`.
2) Afin de vérifier si le point placé aléatoirement est inscrit dans le cercle, il suffit de mesurer sa distance par rapport au centre du cercle, et de vérifier qu'elle est bien inférieure au rayon du cercle. En supposant que le centre c du carré (et donc du cercle) de côté N soit situé en $(0, 0)$, et que l'on ait tiré un point aléatoire p en position (x, y),

sa distance au centre est :

$$d = \sqrt{x^2 + y^2}.$$

Si $d \leq N/2$, cela signifie que le point est inscrit dans le cercle. Codez alors la fonction `int EstInscrit (float N)` qui réalise le tirage d'un point aléatoire inscrit dans le carré de dimension N centré en $(0, 0)$, et renvoie 1 si le point est inscrit dans le cercle, 0 sinon.

3) Il ne reste plus qu'à coder dans le `main()` le reste du programme permettant de demander à l'utilisateur la taille du carré (après tout, la formule fonctionne pour toute valeur de N), et d'appeler un grand nombre de fois la fonction `EstInscrit()` (demandez le nombre d'itérations à l'utilisateur). La dernière étape est d'effectuer le ratio (nombre de points inscrits) / (nombre de points tirés) et de le normaliser en fonction de N pour obtenir l'approximation de π.