Feuille de travaux dirigés n° 1
Langage C++ et environnement

Exercice 1.1

On considère le programme ci-dessous :

```
#include <iostream>
#include <iomanip>
int main(void)
{
  int x = 45;
  int *px = &x;
  std::cout << std::hex << px << std::endl;
}
```

1. Quelle est la valeur de `*px` ?
2. À quel endroit de la mémoire est stockée la variable `x` ? Qu'en déduire sur le moment où la variable est créée et celui où elle est détruite ?

Exercice 1.2

On considère le code suivant :

```
unsigned int a = 0;
for (unsigned int i = 0, j = 10; i <= 10; ++i, j--) {
       a = i+j;
       std::cout << a << "\n";
}
```

1. Réécrire le programme avec une boucle `while` ;
2. Qu'affiche le programme ?

Exercice 1.3

Un point graphique possède une abscisse et une ordonnée de type `double` ou `int` ainsi qu'une couleur.

1. Définir le type `couleur_t` avec au moins cinq couleurs différentes ;
2. Définir le type `point_t` pour représenter un point graphique ;
3. Définir la fonction `display_point()` affichant les coordonnées et couleur d'un point passé en paramètre ;
4. Dans la fonction `display_point()`, que se passe t-il si l'on change les valeurs stockées dans le point passé en paramètre ?
5. Écrire un programme principal créant un point de coordonnées entières et un point de coordonnées réelles.

Exercice 1.4

1. Écrire la fonction `dominante()` prenant en entrée une matrice carrée de `double`s de taille quelconque et retournant « vrai » si la matrice est diagonalement dominante.
   Note : une matrice A de taille n est dite *diagonalement dominante* si l'on a :
   $$\forall i \in \{1, \dots, n\} : \quad |A_{i,i}| \geqslant \sum_{j \neq i} |A_{i,j}|$$
2. Écrire un programme principal utilisant `dominante()`.

Exercice 1.5

1. Écrire la fonction `void reverse_str1(char *str)` retournant une chaîne de caractères en place ;
2. Écrire la fonction `void reverse_str2(const char *const strin, char *const strout)` retournant la chaîne de caractères `strin` dans la chaîne préalablement allouée `strout`.

Exercice 1.6

Écrire la fonction `swap()` prenant en entrée deux `int`s `x` et `y` et échangeant leurs valeurs.

Exercice 1.7

Écrire une fonction `eval()` prenant en entrée une fonction unaire `f` et un double `x` et retournant la valeur `f(x)`.

Exercice 1.8

On souhaite manipuler une date sous la forme d'un triplet (j, m, a).

1. Définir le type `date_t`. On utilisera des entiers sur 16 bits pour le jour, le mois et l'année ;
2. Écrire la fonction `create_date()` prenant en entrée trois entiers `j`, `m` et `a`, et retournant un objet de type `date_t` initialisé avec ces valeurs ;
3. Écrire la fonction `display_date()` prenant en entrée un objet `d` de type `date_t` et un flux `stream` (de type `ostream`) et envoyant dans le flux la représentation textuelle de la date (exemple : la date $(6, 1, 2019)$ doit être représentée par la chaîne `"6 janvier 2019"`).
4. Écrire un programme principal demandant à l'utilisateur le nombre de dates `ndates` à rentrer, allouant un tableau de `ndates` dates, puis lisant sur l'entrée standard les `ndates` dates et les stockant dans le tableau. Le programme devra ensuite afficher toutes les dates à l'écran.

Exercice 1.9

1. Écrire la définition de la structure `noeud_t` représentant une cellule de liste simplement chaînée d'entiers (type `int`) ;
2. Écrire la fonction `tab_vers_liste()` prenant en entrée un tableau d'entiers et retournant une liste chaînée contenant tous les entiers du tableau dans le même ordre ;
3. Écrire la fonction `detruire_liste()` libérant la mémoire occupée par une liste dont la tête est passée en paramètre ;
4. Écrire la fonction `afficher()` imprimant à l'écran la liste des entiers de la chaîne ;
5. Écrire la fonction récursive `afficher_inverse()` affichant la liste à l'envers.

Exercice 1.10

Écrire le programme `echo` affichant tous les paramètres passés sur la ligne de commande.