TP2. Boucles et début des graphismes

Dimitri Galayko

1 Structures de contrôle

Exo 1.1. Ecrivez une fonction qui analyse le nombre saisi par l'utilisateur. L'analyse est la suivante:

Si le nombre est paire, afficher la phrase "Le nombre est paire".

Si le nombre est divisible par 3, afficher la phrase "Le nombre est divisible par 3".

Si le nombre est divisible par 7, afficher la phrase "Le nombre est divisible par 7".

Si le nombre n'est divisible ni par 2, ni par 3, ni par 7, afficher la phrase "Le nombre est quelconque".

Pour faire cet exercice, vous utiliserez les structures *if...else* incorporées que vous avez vu en cours.

Exo 1.2. Ecrivez un programme qui calcule la somme de N premiers nombres naturels. Vous devez utiliser les boucles (la formule de Gauss n'est pas autorisée, c'est trop simple).

La valeur N est stockée dans une variable initialisée au début du programme. Le résultat s'affiche à la fin:

```
#include <stdio.h> // bibliothèque incluant les fonctions mathématiques

int main(){

int N=2000;
int s; // on mettra le résultat dans cette variable
.....
// le programme ici
.....
printf("La somme de %d premiers nombres naturels est %d\n", N, s);
return 0;
}
```

Ecrivez 3 versions de ce programme: l'un utilisant une boucle *for*, l'autre la boucle *while*, la troisième avec *do...while*.

Exo 1.3. Traitement d'un flux de valeurs Soit un programme qui traite les valeurs que l'utilisateur saisit au clavier. Les valeurs à traiter sont positives et entiers. Pour arrêter la lecture des valeurs, l'utilisateur saisit un nombre négatif.

Ecrivez un programme qui fait une lecture de valeurs au clavier de cette manière-là. Après la fin de la lecture (une fois l'utilisateur a saisi un nombre négatif), le programme génère la valeur moyenne des valeurs saisies, la valeur maximale et la valeur minimale.

Contraintes. Vous n'avez pas le droit d'utiliser les tableaux. Réfléchissez aux variables intermédiaires de stockage que vous devez créer pour faire les calculs demandés.

Exo 1.4. Soit $rand(a,b)$ est une fonction générant un nombre aléatoire dans un intervalle $[a,b]$, où a, b sont des entiers.

Ecrivez un programme qui génère N nombres aléatoires se trouvant dans l'interfalle $[1,100]$ et les affiche à l'écran. N est une variable entière initialisée. Essayez le programme avec $N = 5$ et $N = 20$.

La fonction *rand* vous sera fournie par l'enseignant, avec un corps vide de la fonction *main*.

2 Prise en main de la bibliothèque graphique

Exo 2.1. Écrivez un programme qui crée une fenêtre de taille $N \times M$ où N et M sont donnés par les directives `#include`. N donne le nombre de points sur la verticale (Y), M donne le nombre de points sur l'horizontale (X).

Le programme doit :

- Afficher un point au centre de la fenêtre,
- Dessiner deux diagonales
- Dessiner un rectangle de taille $N/2 \times M/2$ au milieu de la fenêtre, à l'aide des fonctions permettant de dessiner des lignes, avec couleur différente de celle de la fenêtre.

Ces trois dessins doivent apparaître l'un après l'autre, et la séquence d'apparition doit être cadencée par un appui d'une touche au clavier (utiliser la fonction `attendre_clavier`).

- Modifiez le programme pour que les 4 cotés du rectangle se dessinent avec les couleurs différentes (choisissez vous même les couleurs).

Exo 2.2. Écrivez un programme qui dessine un carré plein au centre de l'écran, de taille $a \times b$ pixels. La taille de l'écran est donnée par les constantes N (la verticale) et M (l'horizontale).

Écrivez deux versions de ce programme: l'une remplira la surface du rectangle avec les points, l'autre avec les lignes.

Exo 2.3. Écrivez un programme qui affiche K points sur un écran de taille $M \times N$. Les coordonnées de chaque point sont générées aléatoirement. Chaque point peut avoir une couleur parmi le rouge, le noir, le vert et le jaune. Pour chaque point, la couleur est également choisie aléatoirement.

Exo 2.4. Soit un écran graphique de $M \times N$ pixels. Affichez une grille contenant 10 pas sur l'horizontale et sur la verticale.

3 Annexe: la fonction `main()` pour les programmes graphiques

voici la fonction `main()` que vous utiliserez pour vos fonctions graphiques:

```
int main()

{
printf("debut du programme\n");
// pour créer une fenêtre graphique
creer_fenetre(N, M, "white","black");
// pour définir la couleur des pixels
changer_couleur_stylo("blue");
// .....
// ici votre code
// .....
attendre_fermeture_fenetre();
return 0;
}
```