IUT des Pays de l'Adour - RT1
Module I3 - Techniques de Programmation

Langage C

Sujet 10 : Fonctions et tableaux

On souhaite écrire en langage C une bibliothèque de sous-programmes indépendants permettant d'assurer un certain nombre de fonctionnalités sur un ensemble de valeurs réelles représenté par une structure de données de type tableau. On suppose que le nombre de valeurs à traiter ne dépassera jamais 100, quel que soit le contexte d'utilisation [1]. Cet exercice a deux objectifs :

- Le premier est d'écrire un ensemble de fonctions C qui pourra constituer une bibliothèque, et cela de façon indépendante de tout programme susceptible de les utiliser.
- Le second consiste à développer des programmes qui utiliseront effectivement ces sous-programmes.

Cette approche vous permettra d'appréhender des notions telles que la modularité, la ré-utilisabilité et la généricité.

⋆ ⋆ ⋆

La définition de ces modules fonctionnels a déjà été réalisée (phase de conception). Pour chaque module (ou fonction), il est spécifié :

– son nom (et vous devez respecter les majuscules/minuscules),
– l'identificateur de chaque paramètre formel,
– la fonctionnalité globale du module et la sémantique des paramètres

Il vous est formellement interdit d'ajouter des paramètres à ces modules ou d'utiliser des variables globales à vos programmes. Toujours dans le même ordre d'idée, vos modules ne doivent faire aucune lecture ou écriture de données, sauf si ceci est explicitement mentionné dans la description du module.

Description des modules :

1. Ecrire la fonction `ChargerTab(N,Tab)` qui permet de stocker en mémoire `N` valeurs réelles dans le tableau `Tab`. Le nombre `N` doit également être lu par cette fonction (qui doit donc le renvoyer au programme appelant).

2. Ecrire la fonction `EditerTab(N,Tab)` qui permet d'afficher (proprement) les `N` valeurs réelles stockées dans le tableau `Tab`.

3. Ecrire la fonction `SommeTab(N,Tab)` qui retourne la somme des `N` valeurs réelles stockées dans le tableau `Tab`.

4. Ecrire la fonction `MoyenneTab(N,Tab)` qui retourne la moyenne des `N` valeurs réelles stockées dans le tableau `Tab`. Cette fonction devra impérativement utiliser la fonction `SommeTab` écrite précédemment.

5. Ecrire les fonctions `MaxTab(N,Tab)` et `MinTab(N,Tab)` qui retournent respectivement la valeur maximale et la valeur minimale des `N` valeurs réelles stockées dans le tableau `Tab`.

6. Ecrire la fonction `SupValTab(N,Tab,Val)` qui retourne le nombre de valeurs du tableau `Tab` qui sont supérieures à la valeur `Val` passée en paramètre.

1. Rappel : la taille d'un tableau est fixée lors de sa déclaration (voire au moment du `malloc`), et cette taille ne peut plus varier par la suite, même si le tableau est plein ⇒ nous sommes obligés de **sur-dimensionner** nos tableaux.

Solution

```
#include <stdio.h>


void ChargerTab(int* N, float Tab[]) {
  int i;

  printf("Combien de valeurs ? ");
  scanf("%d",N);

  for(i=0;i<*N;i++) {
    printf("Valeur %d: ",i+1);
    scanf("%f",&Tab[i]);
  }
  printf("\n");
}

void EditerTab(int N, float Tab[]) {
  int i;
  for (i=0;i<N;i++) {
    printf("Valeur %d = %f\n",i+1,Tab[i]);
  }
  printf("\n");
}

float SommeTab(int N, float Tab[]) {
  int i;
  float resultat=0;
  for (i=0;i<N;i++) {
    resultat += Tab[i];
  }
  return resultat;
}

float MoyenneTab(int N, float Tab[]) {
  return SommeTab(N,Tab)/N;
}

float MaxTab(int N, float Tab[]) {
  int i;
  float resultat = Tab[0];
  for (i=1;i<N;i++) {
    if (Tab[i]>resultat)
      resultat = Tab[i];
  }
  return resultat;
}
```

```
float MinTab(int N, float Tab[]) {
  int i;
  float resultat = Tab[0];
  for (i=1;i<N;i++) {
    if (Tab[i]<resultat)
      resultat = Tab[i];
  }
  return resultat;
}

int SupValTab(int N, float Tab[], float Val) {
  int i;
  int resultat=0;
  for (i=0;i<N;i++) {
    if (Tab[i]>Val)
      resultat++;
  }
  return resultat;
}

int main(int argc, char** argv) {
  float myTab[100];
  int   nb;
  float v;

  ChargerTab(&nb,myTab);
  EditerTab(nb,myTab);
  printf("Somme   = %f\n",SommeTab(nb,myTab));
  printf("Moyenne = %f\n",MoyenneTab(nb,myTab));
  printf("Max     = %f\n",MaxTab(nb,myTab));
  printf("Min     = %f\n",MinTab(nb,myTab));

  printf("\nValeur seuil: ");
  scanf("%f",&v);
  printf("%d valeurs supÃ©rieures Ã  %f\n",SupValTab(nb,myTab,v),v);
}
```