Licence E.E.A.
TD N°2 de Langage C/C++

Ce TD a pour objectif de vous faire découvrir et utiliser le langage C et peut être C++. Il s'agira de transcrire sous la forme d'un programme C, des algorithmes afin de réaliser les fonctions demandées.

1 Calculer $n!$

Le but de cet exemple est de calculer n ! pour une valeur de n fournie par l'utilisateur et comprise entre 1 et 7. Si la valeur entière fournie par l'utilisateur n'est pas dans cet intervalle, il lui sera demandé une nouvelle valeur.
Notions introduites : définition d'une variable, affichage de la valeur d'une variable, saisie de la valeur d'une variable (fonction `scanf`), la boucle while, la boucle for, les instructions conditionnelles (`if-else`), le type `int`.

2 Une étoile est née

Le but de cet exercice est de construire un dessin de la lettre X , à l'aide d'espaces et d'une "lettre" fournie par l'utilisateur, auquel on demande aussi la "hauteur" du dessin qu'il désire obtenir. Avec les réponses a et 5 il vient :

```
a   a
 a a
  a
 a a
a   a
```

3 Travaillons à la chaîne

Le but de cet exercice est d'utiliser les fonctions `strcpy`, `strcat`, `strcmp`. Dans un premier temps, vous remplirez un tableau de caractères, avec la chaîne "`est le plus fort`", puis à l'aide d'un autre tableau de caractères (`tab` de longueur 51), vous réaliserez la phrase "`popeye est le plus fort`". Enfin vous ajouterez un morceau de phase afin d'écrire "`popeye est le plus fort car il mange des épinards`". Finalement, vous comparerez les deux chaînes de caractères suivantes : "`blanc`" et "`blague`".
Notions introduites : les fonctions `strcpy`, `strcat` et `strcmp`, l'initialisation d'un tableau à la définition.

4 Tableaux multidimensionnels

Le but de cet exercice est de réaliser le produit matriciel des deux matrices suivantes : $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}$ et $\begin{pmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{pmatrix}$.
Notions introduites : les tableaux multidimensionnels.

5 Trier n'est pas jouer

Il s'agit d'écrire un programme permettant à l'utilisateur de saisir des entiers (au plus 8), et de les voir affichés triés par ordre croissant. L'algorithme de tri utilisé est le tri par insertion.
Notions introduites :décomposition d'un programme en fonctions, prototypage des fonctions, définition des constantes entières , utilisation des tableaux statiques.
Spécifications :
Les entiers seront stockés dans un tableau prévu pour 8 entiers. Ils seront ensuite triés dans ce même tableau, qui servira à l'affichage. Ce tableau sera déclaré comme variable globale du programme. Ce tableau est rempli grâce à une fonction nommée `lire()`.

Le principe du tri insertion est le suivant.
On considère que la gauche du tableau est déjà triée. On considère alors l'élément suivant la partie triée et on le fait "descendre" à sa place en le comparant à l'élément qui le précède. On commence par "sauver" l'élément à mettre en place dans une variable "`cle`". On pousse ensuite d'une position vers la droite les éléments qui le précèdent et qui sont plus grands que lui. On le range dans le tableau lorsqu'on a trouvé sa place.
Attention : le premier indice d'un tableau est 0 !

6 Recherche dichotomique

Le but de cet exercice est de faire une recherche dichotomique d'une clé dans un tableau trié. Il s'agit d'écrire deux fonctions. La première, `indice_min`, doit, étant donnés deux indices d'un tableau, indiquer celui en lequel est rangée la plus petite donnée. La seconde, `trier`, doit obligatoirement utiliser la fonction `indice_min` pour trier le tableau. Pour cela, on pourra utiliser un "tri sélection". Ce tri commence par déterminer le plus petit élément du tableau pour le mettre en première position. Il cherche ensuite le second plus petit élément à partir de la deuxième position du tableau et ainsi de suite. Vous trouverez plus de détails dans le programme ci-dessous.

```
#include <stdio.h>

enum {MAXDONNEES=20};

int  tableau[MAXDONNEES];

int lire(void);
int IndiceMin(int,int);
void trier(int);
void afficher(int);

void main()
{
  int NbrDonnees;

  NbrDonnees=lire();
  trier(NbrDonnees);
  afficher(NbrDonnees);
}

int lire()
{
   int donnee, nbr=0;

  printf("\nEntrez les données entieres positives à trier, sur une ligne,"
```

```
     "separes par des blancs.\nTerminer la saisie en tapant -1.\n"
     "Vous avez droit a au plus %d valeurs\n",MAXDONNEES);
  scanf("%d",&donnee);
  while ((nbr<=MAXDONNEES)&&(donnee!=-1))
    {
      if(nbr==MAXDONNEES)
    {
      printf("\nVous avez donne plus de %d entiers, seuls les %d "
         "premiers seront tries\n",MAXDONNEES,MAXDONNEES);
      return nbr;
    }
      else
    {
      tableau[nbr]=donnee;
      nbr++;
      scanf("%d",&donnee);
    }
    }
  return nbr;
}

/*cette fonction prend en argument deux indices dans le tableau
et retourne celui en lequel se trouve le plus petit element*/
int IndiceMin(int indice1,int indice2)
{
  /* A COMPLETER*/
}

/*cette fonction trie le tableau tab qui comporte nombre donnees*/
void trier(int nombre)
{
  /*A COMPLETER en utilisant le tri selection et la fonction IndiceMin*/
}

void afficher(int nombre)
{
  int i;

  printf("\nVoici le tableau trié\n");
  for (i=0;i<nombre;i++) printf("%5d",tableau[i]);
  printf("\n");
}
```

7 Tri sélection

Il vous faudra ici écrire deux fonctions. La première utilisera deux paramètres entiers et retournera un entier ; elle devra déterminer, étant donnés deux indices d'un tableau, celui en lequel se trouve la plus petite donnée. Cette fonction devra être utilisée par la seconde fonction pour trier un tableau par un tri "sélection". Le principe de ce tri est le suivant : il commence par déterminer le plus petit élément du tableau pour le mettre en première position. Il cherche ensuite le second plus petit élément à partir de la deuxième position du tableau et ainsi de suite.

```
/*Il s'agit d'écrire deux fonctions. La première, indice_min,
doit, étant donnés deux indices d'un tableau, indiquer celui en
lequel est rangée la plus petite donnée. La seconde, trier, doit
```

```
obligatoirement utiliser la fonction indice_min pour trier le
tableau.*/


#include <stdio.h>

enum {MAXDONNEES=20};

int  tableau[MAXDONNEES];

int lire(void);
int IndiceMin(int,int);
void trier(int);
void afficher(int);

void main()
{
  int NbrDonnees;

  NbrDonnees=lire();
  trier(NbrDonnees);
  afficher(NbrDonnees);
}

int lire()
{
   int donnee, nbr=0;

  printf("\nEntrez les données entieres positives à trier, sur une ligne,"
     "separes par des blancs.\nTerminer la saisie en tapant -1.\n"
     "Vous avez droit a au plus %d valeurs\n",MAXDONNEES);
  scanf("%d",&donnee);
  while ((nbr<=MAXDONNEES)&&(donnee!=-1))
    {
      if(nbr==MAXDONNEES)
    {
      printf("\nVous avez donne plus de %d entiers, seuls les %d "
         "premiers seront tries\n",MAXDONNEES,MAXDONNEES);
      return nbr;
    }
      else
    {
      tableau[nbr]=donnee;
      nbr++;
      scanf("%d",&donnee);
    }
    }
  return nbr;
}

/*cette fonction prend en argument deux indices dans le tableau
et retourne celui en lequel se trouve le plus petit element*/
int IndiceMin(int indice1,int indice2)
{
  /* A COMPLETER*/
}

/*cette fonction trie le tableau tab qui comporte nombre donnees*/
```

```
void trier(int nombre)
{
  /*A COMPLETER en utilisant le tri selection et la fonction IndiceMin*/
}

void afficher(int nombre)
{
  int i;

  printf("\nVoici le tableau trié\n");
  for (i=0;i<nombre;i++) printf("%5d",tableau[i]);
  printf("\n");
}
```

8 Conversion

Le sujet de cet exercice est la conversion d'un entier en une chaîne de caractères représentant son écriture binaire.
Compléments de connaissances : Avec les entiers, vous disposez entre autres de deux opérations qui pourront vous être utiles : la division entière représentée par le symbole "/" et le reste modulo, représenté par le symbole "%" : par exemple, $7/3 = 2$ et $7\%3 = 1$. Par ailleurs, sachez que toute chaîne de caractères doit se terminer par le caractère "\0".

Énonce : Écrire un programme permettant de saisir un entier positif ou non, qui le transforme en une chaîne de caractères représentant son écriture binaire et qui imprime cette chaîne. On prendra la convention (non universelle) suivante :
– un entier négatif a pour écriture binaire l'écriture binaire de son opposé précédée du signe −.
– le programme ne traitera que le cas des entiers dont la valeur absolue ne dépasse pas 32767.

```
#include <stdio.h>

void convertir(void); /*fonction qui convertit la variable entier de type int
            en une chaîne de caractères notée chaine représentant
            son écriture binaire*/

int entier;      /*pour l'entier à convertir*/
char chaine[33]; /*devra contenir l'écriture de entier sous forme
           d'une chaîne de caractères. On pourra réfléchir
           au choix du 17*/

void main()
{
  printf("Indiquez un entier : ");
  scanf("%d",&entier);
  convertir();
  printf("revoilà votre entier sous forme binaire : %s\n",chaine);
}

void convertir()
{
  /*A COMPLETER*/
}
```