PDF-XChange Editor Click to BUY NOW! www.pdf-xchange.com

TP 2 : Structure de donnée linéaire (Partie Boucle)

Pour l'ensemble des exercices, écrire l'algorithme en pseudo-code avant de coder la solution en langage C.

Vous pouvez utiliser Scratch pour écrire le pseudo code.

Exercice 1

Demander à l'utilisateur deux nombres, les mémoriser dans deux variables, multiplier leurs valeurs en affectant le résultat à une troisième variable, puis l'afficher.

Exercice 2

Même chose que l'exercice 1 avec la division à la place de la multiplication.

Vérifier que le 2ème nombre est différent de zéro, sinon afficher "Erreur : division par 0 !".

Exercice 3

Demander à l'utilisateur de saisir un nombre entier.

Afficher si ce nombre est pair ou impair (le reste de la division entière de ce nombre par deux égal à 0 ou non).

Pour obtenir le reste d'une division, on utilise l'opérateur "Modulo". En langage C l'opérateur Modulo est représenté par le symbole % :

Exercice 4

Demander à l'utilisateur un nombre entier positif. Afficher tous les nombres pairs entre 0 et le nombre saisi."

TP 3 : (Tableau – Structure conditionnelle SELON QUE-structure répétitive)

Vous répondrez tous les exos en C.

Exercice 1 : (Restitution du cours) Vrai/Faux – (5points)

- Un algorithme est une suite d'instruction ordonnée permettant de crée un ou plusieurs problèmes
- Un algorithme est une suite d'instruction ordonnée permettant d'indiquer la démarche à suivre pour crée un problème donné
- Un algorithme est composé d'un entête, des déclarations et d'un corps
- Le type de variable permet de définir les actions à exécuter pour un algorithme
- Le type de variable permet de définir les valeurs que peut prendre cette variable.

Exercice 2 (3points):

En utilisant l'instruction « **SELON QUE – SWITCH CASE** », Écrire un algorithme qui demande à l'utilisateur de saisir le numéro du mois entre (1-12) et afficher le nombre de jours de ce mois.
Le nombre total de jours dans un mois est donné par le tableau ci-dessous.

Mois	Nombre de jours
Janvier, Mars, Mai, Juillet, Août, Octobre, Décembre	31 jours
Février	28/29 jours
Avril, Juin, Septembre, Novembre	30 jours

Exercice 3(2points) :

Ecrire un algorithme (CODE) qui demande à l'utilisateur de saisir 2 nombres entiers et qui doit ensuite afficher si le premier et multiple du second OU non.

Exercice 4 (6 Points) :

Ecrire un algorithme qui demande à l'utilisateur de saisir 20 entiers dans un tableau puis, le programme doit afficher à l'aide d'une **procédure** les 10 derniers entiers du tableau (simplement le 10 dernier entier)
Ensuite le programme retourne la somme de 10 premiers nombres entiers du table (à l'aide d'une **fonction**).
Finalement, mettez en programme le programme principal

Attention : le tableau doit contenir que des nombres entiers POSITIFS

Exercice 5 (boucle répéter jusqu'à) (4points):

Ecrire un algorithme qui demande à l'utilisateur de saisir 100 entiers dans un tableau puis, le programme doit retourner, à l'aide d'une fonction, le nombre des fois qu'une occurrence revienne dans le tableau (c'est-à-dire, le nombre de fois qu'un nombre est répéter dans le tableau)
.

TP : 4
Exercices de programmation en langage C (les tableaux)

Exercice 1

Déclarer un tableau d'entiers de 10 éléments et l'initialiser avec les nombres 1 à 10.

Afficher le tableau en séparant les valeurs par des virgules.

Exercice 2

Déclarer un tableau d'entiers de 100 éléments et l'initialiser avec les nombres 0 à 99 (utiliser une boucle !).

Afficher le tableau en séparant les valeurs par des virgules (limiter à 10 valeurs par lignes).

Résultat attendu :

```
 0,  1,  2,  3,  4,  5,  6,  7,  8,  9
10, 11, 12, 13, 14, 15, 16, 17, 18, 19
20, 21, 22, 23, 24, 25, 26, 27, 28, 29
30, 31, 32, 33, 34, 35, 36, 37, 38, 39
40, 41, 42, 43, 44, 45, 46, 47, 48, 49
50, 51, 52, 53, 54, 55, 56, 57, 58, 59
60, 61, 62, 63, 64, 65, 66, 67, 68, 69
70, 71, 72, 73, 74, 75, 76, 77, 78, 79
80, 81, 82, 83, 84, 85, 86, 87, 88, 89
90, 91, 92, 93, 94, 95, 96, 97, 98, 99
```

Exercice 3

Soient deux tableaux de nombres réels **`tSource`** et **`tDestination`** de 10 éléments chacun.

Écrire un programme permettant de recopier, dans **`tDestination`**, tous les éléments positifs de **`tSource`**, en complétant éventuellement **`tDestination`** par des zéros (initialiser **`tSource`** avec des valeurs au moment de sa déclaration).

Afficher les deux tableaux.

Exercice 4

Écrire un programme qui demande 10 nombres entiers à l'utilisateur, les range dans un tableau avant d'en rechercher le plus grand et le plus petit.

Afficher le tableau, ainsi que le nombre le plus petit et le plus grand.

Exercice 5

Demander à l'utilisateur de saisir des notes (entre 0 et 20) et lui expliquer qu'une valeur hors de cet intervalle arrêtera la saisie.

- **A.** Saisir les notes et les mémoriser dans un tableau
- **B.** Compter les notes saisies et afficher leur nombre
- **C.** Calculer et afficher la moyenne
- **D.** Comparer chaque note à la moyenne et ajouter, dans l'affichage précédent "égal", "inférieur" ou "supérieur à la moyenne"
- **E.** Compter et afficher combien il y a de notes supérieures à la moyenne
- **F.** Dans le tableau de notes, chercher la note la plus petite. Afficher cette note et sa position dans le tableau
- **G.** Même chose pour la note la plus grande.

Le programme affichera un message d'erreur si le nombre de note saisi est 0.

PDF-XChange Editor
Click to BUY NOW!
www.pdf-xchange.com

Exercice 1 :

Ecrire un programme C qui utilise la notion de pointeur pour la permuter le contenu de deux variables de type *char*.

```
#include<stdio.h>
#include<stdlib.h>
int main()
{
   char a,b,*pa,*pb;
   char tmp;
   pa = &a;
   pb = &b;
   printf("Entrez le premier charactere (a): ");
   scanf("%c",pa);
   getchar();
   printf("Entrez le deuxieme charactere (b): ");
   scanf("%c",pb);
   printf("a = %c et b = %c.\n",a,b);
   tmp = *pa;
   *pa = *pb;
   *pb = tmp;
   printf("a = %c et b = %c.\n",a,b);
   system("pause");
   return 0;
}
```

Exercice 2 :

Soit la liste chaînée L, définie ci-dessous, représentant des données d'individus.Pour chacun on mémorise les données suivantes : son matricule, son nom et sa taille.

typedef struct{
int mat;
char nom[30];
float taille;
Liste * suiv;}
Liste;Liste *L;1.

Ecrire une fonction ajoute un individu à la fin de la liste. C

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

```
// Définition de la structure Liste
typedef struct Liste {
   int mat;
   char nom[30];
   float taille;
   struct Liste* suiv;
} Liste;

// Déclaration du pointeur de liste
Liste* L;

// Fonction pour ajouter un individu à la fin de la liste
void ajouterFin(Liste** liste, int mat, const char* nom, float taille) {
   // Allouer de la mémoire pour un nouveau nœud
   Liste* nouveauNoeud = malloc(sizeof(Liste));
   if (nouveauNoeud == NULL) {
      // Gestion d'erreur si l'allocation de mémoire échoue
      fprintf(stderr, "Erreur d'allocation de mémoire\n");
      exit(EXIT_FAILURE);
   }

   // Initialiser les champs du nouveau nœud avec les données fournies
   nouveauNoeud->mat = mat;
   strncpy(nouveauNoeud->nom, nom, sizeof(nouveauNoeud->nom) - 1);
   nouveauNoeud->nom[sizeof(nouveauNoeud->nom) - 1] = '\0'; // Assurer une terminaison nulle
   nouveauNoeud->taille = taille;
   nouveauNoeud->suiv = NULL;

   // Si la liste est vide, le nouveau nœud devient la tête de la liste
   if (*liste == NULL) {
      *liste = nouveauNoeud;
   } else {
      // Sinon, parcourir la liste jusqu'à la fin
      Liste* noeudCourant = *liste;
      while (noeudCourant->suiv != NULL) {
         noeudCourant = noeudCourant->suiv;
      }

      // Ajouter le nouveau nœud à la fin de la liste
      noeudCourant->suiv = nouveauNoeud;
   }
}

// Fonction pour afficher les éléments de la liste
void afficherListe(Liste* liste) {
   while (liste != NULL) {
      printf("Matricule: %d, Nom: %s, Taille: %.2f\n", liste->mat, liste->nom, liste->taille);
      liste = liste->suiv;
```

PDF-XChange Editor Click to BUY NOW! www.pdf-xchange.com

```
    }
}

int main() {
    // Initialiser le pointeur de liste
    L = NULL;

    // Ajouter quelques individus à la fin de la liste
    ajouterFin(&L, 1, "John Doe", 1.75);
    ajouterFin(&L, 2, "Jane Smith", 1.68);
    ajouterFin(&L, 3, "Bob Johnson", 1.80);

    // Afficher les éléments de la liste
    afficherListe(L);

    // Libérer la mémoire allouée
    while (L != NULL) {
        Liste* temp = L;
        L = L->suiv;
        free(temp);
    }

    return 0;
}
```

Faculté des Sciences, Université de Djibouti

2023/2024

Module : Structurede données Linéaire

Série de TP N°5 : Chaine des caractères

Exercice : 1

Lesquelles des chaînes suivantes sont initialisées correctement ? Corrigez les déclarations fausses et indiquez pour chaque chaîne de caractères le nombre d'octets qui sera réservé en mémoire.

a) char a [ ] = "un\ndeux\ntrois\n";

b) char b [12] = "un deux trois";

c) char c [ ] = 'abcdefg';

d) char d [10] = 'x';

e) char e [5] = "cinq";

f) char f [ ] = "Cette " "phrase" "est coupée";

g) char g [2] = {'a', '\0'};

h) char h [4] = {'a', 'b', 'c'};

i) char i [4] = "'o'";

Exercice : 2

On donne la déclaration de la chaîne suivante :

```
#include <stdio.h>
#include <string.h>

int main () {
  char src[50], dest[50];

  strcat(dest, src);

  printf("Final destination string : |%s|", dest);

  return(0);
}TAF :
Compiler et dites que fait ce programme ?
```


Exercice : 3

```
#include <stdio.h>
#include <string.h>

int main () {
  char src[50], dest[50];

  strcpy(src, "This is source");
  strcpy(dest, "This is destination");

  strcat(dest, src);

  printf("Final destination string : |%s|", dest);

  return(0);
}
```

TAF :

Compiler et dites que fait ce programme ?

Exercice : 4

Ecrire un programme permettant de demander à l'utilisateur de saisir son nom, son prénom et son adresse et qui, ensuite les affiche à l'écran.

Exercice 5 :

On considères la chaine des caractères suivante :

```
char str[] = "string";
```

1. Ecrire un programme de permettant de d'afficher la chaine
2. Ecrire une fonction permettant de retourner le nombre des caractères de cette
3. Ecrire une procédure permettant d'afficher la chaine caractère par caractère
4. Ecrire un programme principal

Exercice 6 :

```
#include <stdio.h>
#include <string.h>

int main()
{
      char src[] = " Ali";
      char dest[50] = "papa";

      strcat(dest, src);
     // Affiche : "debutfin"
      printf(dest);

      return 0;
```