Université Echahid Hamma Lakhdar
Département d'Informatique 2ème LMD Informatique
Module : Algorithmique et Structures de Données
El-Oued le : 14/01/2025

Correction de l'Examen du troisième semestre (Durée:1h:30)

Exercice 01: 07 points

Écrivez un programme en C qui :

1. Demande à l'utilisateur la taille d'un tableau dynamique d'entiers.
2. Alloue dynamiquement la mémoire pour ce tableau.
3. Permet à l'utilisateur de saisir les éléments du tableau.
4. Calcule et affiche la somme des éléments du tableau.
5. Libère correctement la mémoire allouée.

Solution:

```
#include < stdio.h >
#include < stdlib.h > // Pour malloc et free

int main() {
int n, i;
int *array; // Pointeur pour le tableau dynamique 0.5 point

// Étape 1 : Demander la taille du tableau
printf("Entrez la taille du tableau : ");
scanf("%d", &n); 0.5 point

// Étape 2 : Allocation dynamique de mémoire 01 point
array = (int *)malloc(n * sizeof(int));
if (array == NULL) {
printf("Erreur d'allocation mémoire.\n");
return 1; }

// Étape 3 : Saisir les éléments du tableau 02 points
printf("Entrez les %d éléments du tableau :\n", n);
for (i = 0; i ¡ n; i++) {
printf("Élément %d : ", i + 1);
scanf("%d", &array[i]);
}

// Étape 4 : Calculer la somme des éléments 02 points
int sum = 0;
for (i = 0; i ¡ n; i++) {
sum += array[i];
}
printf("La somme des éléments est : %d\n", sum);

// Étape 5 : Libérer la mémoire
free(array); 01 point

return 0;
}
```

Exercice 02: 07 points

Correction de l'Examen du troisième semestre (Durée:1h:30)

Soit une liste simple chainée dont chaque élément de la liste est défini de la façon suivante :

```
typedef struct liste {
int val;
struct liste *suivant;
}Liste;
```

Ecrire un programme en langage C qui assure les sous tâches ci-dessous:

1. **supprimerEnFin(Liste *L):** permet de supprimer un élément en fin d'une liste L et donne comme résultat une nouvelle liste ne contenant pas de l'élément supprimé;
2. **ajouterEnDebut(Liste *L, int N):** est une fonction qui permet d'ajouter un entier N en Debut de la liste L et donne une nouvelle tête de liste.

Solution:

```
1- (05 points)
Liste* supprimerEnFin(Liste* L)
{ Liste* tmp; 0.5 point
Liste* liste=L; 0.5 point
if(L==NULL) 0.5 point
printf("echec:la liste est vide");
while(L− >suivant!=NULL) 01 point
{ tmp=L; 0.5 point
L=L− >suivant; 0.5 point
}
tmp− >suivant=NULL; 0.5 point
free(L); 0.5 point
return liste; 0.5 point
}
2-(02 points)
Liste *ajouterEnDebut(Liste *L, int valeur)
{
Liste* nouveau=malloc(sizeof(element)) ; 0.5 point
nouveau− >val=valeur; 0.5 point
nouveau− >suivant=L;0.5 point
return nouveau; 0.5 point
}
```

Exercice 03: 06 points

L'explication du programme est détaillée pour chaque instruction. Le contenu ainsi que les positions de la tête et de la queue de la file sont spécifiés. Il suffit de compléter avec un schéma explicatif. Un point est attribué pour chaque élément correctement positionné dans la file (donc 05 points), et un autre 01 point pour l'indication précise des positions de la tête et de la queue de la file d'attente

Cont.

```
char C1, C2;
Enfiler(F,'A');\\F contient (A) qui est la tête et la queue en même temps
Enfiler(F,'B');\\F contient (A,B) A tête et B queue
Enfiler(F,'C');\\F contient (A,B,C) A tête et C queue
C1=Defiler(F);\\C1 contient A et B devient tête de file
C2=Defiler(F);\\C2 contient B et C devient tête et queue de file
Enfiler(F,'a');\\F contient (C,a) a devient queue
C1=Defiler(F);\\C1 contient C et a devient tête et queue de file
Enfiler(F,'b');\\F contient (a,b) a tête et b queue
Enfiler(F,C2);\\F contient (a,b,B) a tête et B queue
Enfiler(F,C1);\\F contient (a,b,B,C) a tête et C queue
Enfiler(F,C2);\\F contient (a,b,B,C,B) a tête et B queue
```


Bon Courage

The End.