|  |
| --- |
| **Devoir Surveillé** |
| Matière: **Algorithme et Programmation** Documents: **Non autorisés** Enseignants : S. Refai, S. Majdoub, H. Bouajina , N. Ferjani, A. Ezzaich Durée: **1h00'**Classes : TI.11, 12, 13, 14, 15, 16 Date : 08/11/2023 |

*Il vous est demandé d’apporter un soin particulier à la présentation de votre copie.*

**Exercice 1 : Programme C 7 *Pts***

Une table de vérité est une matrice qui représente des entrées (en colonne) exprimées sous forme des états binaires (0/1) et une sortie, également représentée sous forme de colonne, est la résultante des états d'entrée, elle-même exprimée sous forme d'état binaire.

|  |  |  |
| --- | --- | --- |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Ecrire un programme C qui :

1. Lit un entier **n** qui représente le nombre des entrées (2<=**n**<=4). (1pt)
2. Remplit la table de vérité de **n** entrées et une sortie. (4pts)

* C’est une matrice **T** de 2n lignes (Les combinaisons possibles des entrées) et n+1 colonnes (n colonnes pour les n entrées et la dernière colonne pour la sortie)
* Pour le calcul de puissance vous pouvez utiliser la fonction pow(x,y)=xy. Cette fonction se trouve dans le fichier d’en-tête « math.h ».
* Pour avoir toutes les combinaisons possibles des entrées, on remplit la matrice par colonne :
  + Pour la première colonne, remplir la moitié avec des 0 et le reste avec des 1
  + Pour la deuxième colonne, remplir le quart des lignes avec des 0 puis un quart avec des 1 ensuite un quart avec des 0 et enfin le dernier quart avec des 1.
  + Poursuivre le même principe pour les autres colonnes qui représentent les entrées.
* La dernière colonne représente la sortie. Les valeurs de cette colonne sont entrées au clavier avec des valeurs égales à 0 ou à 1.

*Exemple***: n=**3

|  |  |  |  |
| --- | --- | --- | --- |
| ***Entrée1*** | ***Entrée2*** | ***Entrée3*** | ***Sortie*** |
| 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 |
| 0 | 1 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 0 |
| 1 | 1 | 1 | 1 |

1. Affiche la matrice **T**. (2pt)

**Exercice 2 : Algorithme 13*Pts***

Développement d’un programme qui effectue une attribution optimale de **m** paires de skis de différentes tailles (t1 .. tm) à **n** skieurs de différentes longueurs (l1 .. ln) avec m>n. Chaque skieur doit obtenir une paire de skis correspondante à sa taille si elle est disponible. Dans le cas où une paire de skis de la taille exacte n'est pas disponible, le skieur peut prendre la plus petite taille supérieure à sa longueur, avec une différence maximale de 3.

Il est important de donner la priorité au skieur arrivé en premier.

Ecrire un algorithme qui :

1. Déclare les constantes **MAX\_SKIEURS** et **MAX\_PAIRES** représentants respectivement la taille maximale du tableau **L** et **T** avec les valeurs **50** et **150**. (1pt)
2. Lit la taille **n** du tableau **L** qui ne doit pas dépasser ***MAX\_SKIEURS***. (1pt)
3. Remplir un tableau **L** avec les longueurs des skieurs qui sont des entiers entre 25 et 45. (1.5 pts)

**Exemple : n=5**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 37 | 38 | 45 | 25 | 29 |

1. Lit la taille **m** du tableau **T** qui doit être supérieur ou égale à **n** et inférieur ou égale à **MAX\_PAIRES /2**. (1pt)
2. Remplir un tableau **T** avec les tailles des paires des skis disponibles qui sont aussi des entiers entre 25 et 45. (1.5pts)

**Exemple : m=7**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 28 | 35 | 45 | 28 | 30 | 26 | 37 |

1. Modifie le tableau **T** de tel sorte que le skieur de longueur **L**[i] prend une paire de taille **T**[i] si elle existe, sinon **T**[i] vaut **0**. (Sachant que le skieur arrivé en premier a toujours un rang inférieur dans le tableau **L**). (3pts)

*NB* : Les paires de ski qui n'ont pas été attribuées seront déplacées vers la fin du tableau **T**. Donc, la taille m du tableau T va être changée.

**T après exécution du programme : m devient 8**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 37 | 0 | 45 | 26 | 30 | 28 | 28 | 35 |

1. Affiche pour chaque skieur la taille de ses paires de ski si elles existent, sinon, afficher que ce joueur n’a pas des paires. (2pts)
2. Calcule et affiche le nombre des skieurs qui n’ont pas eu une paire de ski. (1pts)
3. Calcule et affiche le nombre des paires de ski non attribuées. (1pts)

**Bon travail**