المعهد الوطني للعلوم التطبيقية والتكنولوجيا
Insat

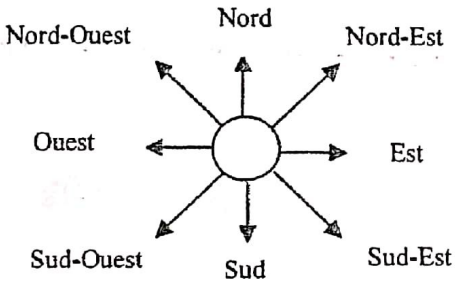
Devoir surveillé ☐	Examen ■	Session : principale ☐ de contrôle ■

Matière : Algorithmique et structures de données I	Semestre: 1
Enseignant :Aymen Sellaouti et Majdi Jribi	Date: 23/07/2019 à 10h30
Filière(s) : MPI	Durée: 1h30
Nombre de pages : 3	Documents : autorisés ☐ non autorisés ■

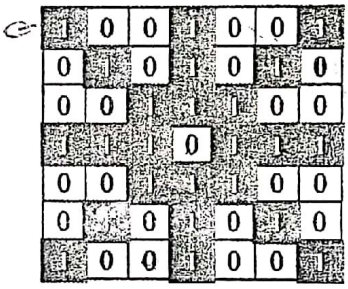
Exercice 1 :

On étudie une forme de dessin qui permet de représenter les 8 directions d'une boussole B sous forme d'une matrice carrée de taille impaire ≥ 3. Une boussole est dite à **l'état initial** si les cases qui caractérisent une direction sont mises à 1 et les autres à 0.

Voici les différentes directions d'une boussole :



Voici un exemple de boussole B de taille 7 à l'état initial :



1. Ecrire en **langage C** une fonction ***Creation_Aff*** qui permet de créer et d'afficher une boussole B en **état initial** de taille t.

   On considère maintenant un tableau de lettres qui représente un chemin. Un chemin est un ensemble de directions prises pour atteindre une cible. Une direction **valide** est représentée par une chaîne de deux caractères parmi (NN, SS, EE, OO, NE, NO, SE, SO) tel que:

NN	*: Nord*	**OO**	*: Ouest*
NE	*: Nord-Est*	**SS**	*: Sud*
NO	*: Nord-Ouest*	**SE**	*: Sud-Est*
EE	*: Est*	**SO**	*: Sud-Ouest*

2. Ecrire en **algorithmique** puis en **langage C** une fonction ***Remplissage*** qui permet de remplir le tableau chemin ***TabCh*** de type caractère de taille 2n avec n est le nombre de directions **valides** (voir exemple 1).

Exemple 1

Avec 7 directions valides voici le résultat du remplissage du tableau ***TabCh*** de taille 14.

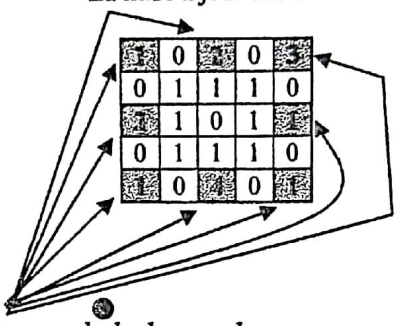
N	N	N	E	N	E	S	S	S	S	S	S	O	O

3. Une fois le tableau chemin est rempli, on met à jour les cases de la boussole B d'une taille t et on l'affiche. Pour cela, on parcourt deux à deux les cases du tableau ***TabCh***, reconnaître la direction d'après ces deux cases, et incrémenter l'extrémité supérieure qui correspond à cette direction par 1 (voir exemple 2 qui correspond au tableau ***TabCh*** de l'exemple 1).

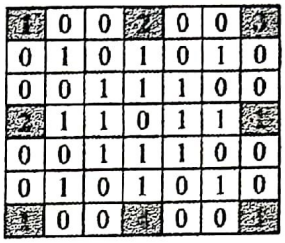
Exemple 2

Si la Boussole B est de taille 3
La mise à jour devient :

Si la Boussole B est de taille 5
La mise à jour devient :



Si la Boussole B est de taille 7
La mise à jour devient :



Les extrémités supérieures de la boussole

Ecrire en **langage C** la fonction ***Mise_a_jour*** qui permet la mise à jour d'une boussole de taille t selon un tableau chemin donné.

Exercice 2 :

On dispose des fichiers texte suivants :
« **skills.txt** »
Chaque ligne de ce fichier contient les données suivantes :

Code	: 3 caractères (entier)
Désignation	: 15 caractères
Catégorie	: 15 caractères

Le code ainsi que la désignation d'une compétence sont uniques.
« **candidat.txt** »
Chaque ligne de ce fichier contient les données suivantes :

Code	: 3 caractères (entier)
Nom	: 25 caractères
Age	: 3 caractères (entier)

Le code est unique.
« **compétences.txt** »
Qui regroupent les compétences de chaque candidat et une note évaluant cette compétence.
Chaque ligne contient le code candidat et le code d'une des compétences qu'il possède.

CodeSkills	: 3 caractères (entier)
CodeCandidat	: 3 caractères (entier)
evalSkill	: 2 caractères (entier)

On vous demande de :

1) Ecrire la fonction getNameCandidatByCode qui retourne le nom d'un candidat à partir de son code.

```
Void findPersonWithSkill(File* fp, int codePersonne)
{
}
```

2) Ecrire une fonction qui affiche la liste des noms des personnes possédant un skill d'un code donné.

```
Void findPersonsWithSkill(File* fp, int codeSkill)
{
}
```

3) Ecrire la fonction updateCategorySkill qui permet à partir de mettre à jour la catégorie d'un skill de code donné. Si le code n'existe pas la fonction retourne 0 sinon elle le met à jour et retourne 1.

```
int updateSkillCategory(File* fp, int codeSkill, char* newCategory)
{
}
```

4) Soit la structure UserEval qui contient trois champs : userCode, nbSkill et avgSkill.

```
TypdeDef Struct {
int userCode ;
int nbSkill ;
float avgSkill;
} UserEval ;
```

Ecrire la fonction statsUserSkill qui permet à partir du fichier « compétence.txt » de générer un tableau de UserEval. Chaque utilisateur présent dans le fichier « compétence.txt » aura une et une seule structure UserEval dans le tableau qui contiendra son code, le nombre de skill qu'il possède dans le fichier et la moyenne de ses skills.

Faite en sorte que le fichier compétence.txt soir parcouru une seule fois.

```
int getPersonStatSkill(File* fp, UserEval* t)
{
}
```

Bon Travail