HE2B ESI

Nom : _____

Prénom : _____

Identifiant : _____ Groupe : _____

/ 20

Haute École de Bruxelles-Brabant
École Supérieure d'Informatique
Bachelor en Informatique

Mai 2018

DEV2 – Développement

Examen première session

Algorithmique

- *l'examen dure 3h ;*
- *sauf mention contraire, on peut considérer que les données lues ou reçues ne comportent pas d'erreurs ;*
- *veillez à rendre vos solutions modulaires. Ceci est **très** important.*

1 Sudoku (8 points)

Le but du jeu est de remplir la grille avec une série de chiffres (ou de lettres ou de symboles) tous différents, qui ne se trouvent jamais plus d'une fois sur une même ligne, dans une même colonne ou dans une même région (également appelée « bloc »,« groupe »,« secteur » ou « sous-grille »). La plupart du temps, les symboles sont des chiffres allant de 1 à 9, les régions étant alors des carrés de 3×3. Quelques symboles sont déjà disposés dans la grille, ce qui autorise une résolution progressive du problème complet. (wikipédia 27/04/18)

8	3		1			6		5
							8	
			7			9		
	5			1	7			
		3				2		
			3	4			1	
		4			8			
	9							
3		2			6		4	7

8	3	7	1	9	4	6	2	5
5	4	9	6	2	3	7	8	1
6	2	1	7	8	5	9	3	4
2	5	6	8	1	7	4	9	3
4	1	3	5	6	9	2	7	8
9	7	8	3	4	2	5	1	6
1	6	4	2	7	8	3	5	9
7	9	5	4	3	1	8	6	2
3	8	2	9	5	6	1	4	7

http://codinghelmet.com/?path=exercises/sudoku-solver

On vous demande d'écrire un algorithme modulaire qui permettra de vérifier une solution d'une grille de sudoku. Votre algorithme recevra une grille complétée et retournera la valeur VRAI si elle est exacte.

Version du document : 18 mai 2018

2 Voyage, voyage (6 points)

Pour organiser un voyage itinérant nous avons encodé dans une liste une suite de tronçons. Un tronçon est une structure comprenant les attributs :

- numéro : numéro d'ordre dans la suite chronologique du voyage (entier)
- lieu : nom du lieu (chaîne)
- distance : nombre de kilomètres pour ce tronçon (entier)

Exemple

```
1, Koekelberg, 13
2, Watermaal, 14
3, Ohain, 21
4, Gentinnes, 23
5, Moustier-sur-Sambre, 21
6, Namur, 21
7, Houyet, 19
```

Le voyage se faisant à vélo, il faudrait faire des étapes de moins de 60 km. La liste des tronçons est triée sur le numéro de tronçon. Votre travail consistera donc en un découpage du trajet en étapes. Chaque étape regroupera un maximum de tronçons successifs tout en respectant la limite des 60 km. Vous pouvez supposer que les tronçons donnés font une distance supérieure ou égale à 0, et inférieure ou égale à 60km.

On vous demande d'écrire :

1. la définition de la structure ;
2. un algorithme qui reçoit la liste et affiche
   - le numéro de l'étape ;
   - la description de tous les tronçons triés sur leur numéro de cette étape ;
   - à chaque fin d'étape, le nombre de kilomètres pour cette étape ;
   - le nombre total d'étapes ;
   - le nombre total de kilomètres.

Exemple

```
Étape 1
1 - Koekelberg - 13 km
2 - Watermaal - 14 km
3 - Ohain - 21 km
Total étape = 48 km

Étape 2
4 - Gentinnes - 23 km
5 - Moustier-sur-Sambre -  21 km
Total étape = 44 km

Étape 3
6 - Namur - 21 km
7 - Houyet - 19 km
Total étape = 40 km

Au final, 3 étapes et 132 km
```

3 Réflexion : bataille navale (6 points)

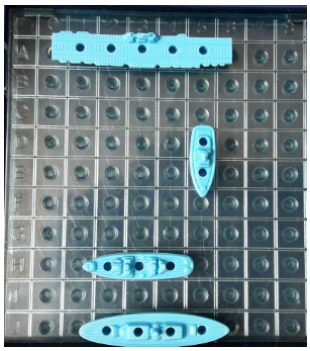
Pour cette question, vous ne devrez pas coder des algorithmes mais les décrire en texte selon 2 propositions alternatives de représentation des données, et présenter les arguments pour ou contre une représentation des données pour chaque algorithme.

La bataille navale, appelée aussi touché-coulé, est un jeu de société dans lequel deux joueurs A et B doivent placer leurs 4 navires sur une grille tenue secrète et tenter de toucher les navires adverses. Nous considérons ici une version simplifiée du jeu. Les règles sont donc différentes de celles que vous pourriez connaître.

Chaque joueur dispose donc de 2 tableaux. Un vide dans lequel il placera à chaque tour les informations reçues de l'adversaire à propos des bateaux de celui-ci (et l'eau qui entoure ceux-ci) et un tableau secret avec ses bateaux.

Chaque joueur à son tour indique à l'adversaire une position. L'adversaire répond en indiquant le bateau touché (ou coulé lorsque toutes les cases d'un bateau ont été touchées) ou « dans l'eau »si aucun bateau n'est touché.

Le gagnant est celui qui parvient à couler tous les navires de l'adversaire avant que tous ses navires ne le soient.



Les 4 navires sont classés ci-dessous par la place qu'ils occupent sur le plateau de jeu :

- 1 porte-avions de 5 cases ;
- 1 croiseur de 4 cases ;
- 1 torpilleur de 3 cases ;
- 1 sous-marin de 2 cases.

Ce qu'il faudra décrire

Nous considérons ici un morceau du jeu, le cas restreint du tableau secret du joueur A. Pour programmer ce jeu, il faut disposer des deux algorithmes suivants :

- un coup proposé par B touche-t-il un navire de A ?

  algorithme *vérifierCoup*(ligne↓, colonne↓ : entiers) → entier

  L'entier retourné vaut
  - ▷ 0 : à l'eau ;
  - ▷ n : un bateau de taille n a été touché ;
  - ▷ $-n$: un bateau de taille n a été coulé.
- le jeu est-il terminé (pour A), c'est-à-dire que tous ses bateaux ont-été coulés ?

  algorithme *estTerminé*() → booléen

Nous vous proposons deux structures de données différentes pour implémenter le plateau de jeu.

Représentation 1 : un tableau

On utilise un tableau de $n \times m$ entiers

- la valeur 0 représente une case vide ;
- les différents bateaux sont représentés par un entier dépendent de sa taille (ex : 5 pour un porte-avion de 5 cases) ;
- un élément touché est représenté par une valeur négative (ex : -3 pour un élément de torpilleur touché)

Voici un état possible du tableau [1]

	0	1	2	3	4	5	6	7
0	0	**5**	**-5**	**5**	**5**	**5**	0	0
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	**-2**	0	0
4	0	0	0	0	0	**2**	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	**-3**	**-3**	**-3**	0	0	0
8	0	0	0	0	0	0	0	0
9	0	0	**4**	**4**	**4**	**4**	0	0

Représentation 2 : une liste de listes

Un élément de bateau est une structure

```
structure ÉlémentBateau
    identifiant : entier    // 5 pour le porte-avions, 4 pour le croiseur...
    ligne : entier
    colonne : entier
    touché : booléen    // Indique si CE bout de bateau est touché
fin structure
```

Un bateau est une liste de ces éléments et le jeu est une liste de bateaux.

Si on reprend le même exemple que pour la première représentation, on obtient :

```
[
    [ {2,3,5,T}, {2,4,5,F} ],
    [ {3,7,2,T}, {3,7,3,T}, {3,7,4,T} ],
    [ {4,9,2,F}, {4,9,3,F}, {4,9,4,F}, {4,9,5,F} ],
    [ {5,0,1,F}, {5,0,2,T}, {5,0,3,F}, {5,0,4,F}, {5,0,5,F} ]
]
```

1. Dans le tableau suivant, la première ligne et la première colonne donnent les indices des colonnes et des lignes respectives.

Ce qu'on vous demande

Pour chacun des deux algorithmes évoqués ci-dessus

1. donnez, en français, une brève description de l'algorithme pour chacune des représentations (donc 4 descriptions : 2 représentations pour les 2 algorithmes) ;
2. pour chaque algorithme, discutez le pour et le contre des deux implémentations.

Attention ! Nous ne vous demandons pas de discuter des pour ou contre par rapport à d'autres algorithmes, même si cet impact est potentiellement important.