FIT 2 [INF224] – 2021/22

Algorithmique et structures de données
TP 06

Gulfem Isiklar Alptekin – Ozgun Pinarer

Partie 1 : L'allocation dynamique

- Une variable occupe plus ou moins d'espace en mémoire en fonction de son type.
- On peut connaître le nombre d'octets occupés par un type à l'aide de l'opérateur sizeof().
- L'allocation dynamique consiste à réserver manuellement de l'espace en mémoire pour une variable ou un tableau.
- L'allocation est effectuée avec malloc()et il ne faut surtout pas oublier de libérer la mémoire avec free()dès qu'on n'en a plus besoin.
- L'allocation dynamique permet notamment de créer un tableau dont la taille est déterminée par une variable au moment de l'exécution.

Question 1

Créez un menu et écrivez les fonctions :

- Tapez 1 pour *get(i)* : renvoie l'élément à l'emplacement i*
- Tapez 2 pour *set(i, val)*: Définit l'élément i à val *
- Tapez 3 pour *pushBack(val)*: Ajoute val à la fin
- Tapez 4 pour *remove(i)* : supprime l'élément à l'emplacement i
- Tapez 5 pour *taille()* : le nombre d'éléments
- Tapez 6 pour *getCapacity()* : la dimension
- Tapez 7 pour quitter

Indiquez le nombre d'élément dans la liste et la dimension de liste apres chaque opération.

Partie 2 : Disjoint Set

Question 2

Ecrivez les fonctions et testez-les :

- *disjointSet(int numElems)* : construit un nouvel ensemble contenant le nombre donné d'ensembles de singletons. Par exemple, nouveau DisjointSet(3) → {{0}, {1}, {2}}.
- *boolean areInSameSet(int elemIndex0, int elemIndex1)* : teste si les deux éléments donnés sont membres du même ensemble. Notez que les arguments sont sans ordre.
- *boolean mergeSets(int elemIndex0, int elemIndex1)* : fusionne les ensembles auxquels appartiennent les deux éléments donnés. Cette méthode est également appelée « union » dans la

littérature. Si les deux éléments appartiennent à des ensembles différents, alors les deux ensembles sont fusionnés et la méthode renvoie true. Sinon, ils appartiennent au même ensemble, rien n'est modifié et la méthode renvoie false. Notez que les arguments sont sans ordre.

- *int getNumElements()* : renvoie le nombre d'éléments parmi l'ensemble d'ensembles disjoints ;
- *int getNumSets()* : Renvoie le nombre total d'ensembles disjoints. Ce nombre diminue de façon monotone au fur et à mesure que le temps avance ; chaque appel à mergeSets() décrémente le nombre de un ou le laisse inchangé. 0 ≤ résultat ≤ getNumElements().
- *int getSizeOfSet(int elemIndex)* : renvoie la taille de l'ensemble dont l'élément donné est membre. 1 résultat ≤ getNumElements().
- union()
- find()

IMPORTANT:

- Vous devez effectuer des études TP en tant que « fichier d'en-tête – fichier source – fichier de test » : écrivez les en-têtes des fonctions que vous avez écrites dans le fichier d'en-tête, les codes des fonctions dans le fichier source et les codes de test dans le fichier d'essai.
- Nous préférons faire des études de TP dans un environnement linux, sans avoir besoin d'IDE supplémentaire.
- Compressez vos fichiers et téléchargez-le sur le système en les nommant « NumeroEtudiant_Prenom_Nom_TPX.zip ».