EXERCICES SUR LES DONNEES STRUCTUREES

Exercice 1

On donne la séquence d'instructions suivante (la fonction INSERER(L , n , e) consiste à insérer l'élément e en position n dans la liste L et la fonction LIRE(L, n) consiste à renvoyer la valeur en position n de la liste L) :

```
L1 = CREER_LISTE_VIDE()
L2 = CREER_LISTE_VIDE()
INSERER (L1 , 1 , 1)
INSERER (L1 , 2 , 2)
INSERER (L1 , 3 , 3)
INSERER (L1 , 4 , 4)
INSERER (L2 , LIRE(L1 , 1) , 1)
INSERER (L2 , LIRE(L1 , 2) , 1 )
INSERER (L2 , LIRE(L1 , 3) , 1)
INSERER (L2 , LIRE(L1 , 4) , 1)
```

1. Illustrer le résultat de chaque étape.
2. Quelle est l'opération effectuée ?

Exercice 2

On dit qu'une chaîne de caractères comprenant, entre autres choses, des parenthèses (et) est bien parenthésée lorsque chaque parenthèse ouvrante est associée à une unique fermante, et réciproquement.

Ecrire une fonction prenant en paramètres une chaîne bien parenthésée s et l'indice f d'une parenthèse fermante, et qui renvoie l'indice de la parenthèse ouvrante associée.

Indice : comme chaque parenthèse fermante est associée à la dernière parenthèse ouvrante non encore fermée, on peut suivre les associations à l'aide d'une Pile.

Exercice 3

On considère une chaîne de caractères incluant à la fois des parenthèses rondes (et) et des parenthèses carrées [et]. La chaîne est bine parenthésée si chaque ouvrante est associée à une unique fermante **de même forme**.

Ecrire une fonction prenant en paramètre une chaîne de caractères contenant, entre autres, les parenthèses décrites et qui renvoie True si la chaîne est bien parenthésée et False sinon.

Exercice 4

On donne la séquence d'instructions suivante :

```
F = CREER_FILE_VIDE()
ENFILER(F, 4)
ENFILER(F, 1)
ENFILER(F, 3)
N = DEFILER(F)
ENFILER(F, 8)
N = DEFILER(F)
```

Illustrer le résultat de chaque étape de cette séquence.

Exercice 5

On suppose que l'on a déjà une file F1 qui contient les éléments suivants saisis dans l'ordre alphabétique F1 = ['A', 'B', 'C,' 'D', 'E').

1. Quel est l'élément issu d'un défilage de F1 ?

2. Proposer une séquence d'instruction (à l'aide de deux piles P1 et P2) permettant la saisie d'affilée (sans sortie intermédiaire) des 5 éléments 'A', 'B', 'C', 'D' et 'E' et de sortir ces éléments comme s'ils sortaient d'une file.

3. Que faudrait-il faire pour avoir exactement le même fonctionnement qu'avec une file, c'est-à-dire avec sortie éventuelle d'élément.

Exercice 6

Quelle structure de données choisir pour chacune de ces tâches ?

1. Représenter un répertoire téléphonique

2. Stocker l'historique des actions effectuées dans un logiciel et disposer d'une commande Annuler.

3. Envoyer des fichiers au serveur d'impression.

Exercice 7

Une usine européenne fabrique des voitures.
Chaque véhicule en cours de fabrication ou terminé possède un numéro de série pour l'identifier.

1. Quelle structure de données est à privilégier pour gérer les véhicules sur la chaîne de production sur laquelle les portes sont fixées à la carrosserie ?

2. Quelle structure de données est à privilégier pour gérer le lien entre le numéro de série du véhicule terminé avec l'ensemble de ces caractéristiques ?

3. Les véhicules de luxe terminé sont stockés avant envoi dans des longs hangars étroits cul à cul.
   Sachant que ces hangars ne possèdent qu'une seule porte d'entrée à véhicule, quelle est la structure de données à privilégier pour gérer l'entrepôt de ces véhicules ?

4. L'entreprise travaille avec de nombreux sous-traitants européens.
   Quelle structure de données est à privilégier pour gérer l'ensemble des noms sous-traitants européens ?

5. Pour communiquer avec une entreprise sensible se trouvant aussi dans l'Union Européenne, l'entreprise vérifie toutes les minutes le routage des paquets entre elle et cette entreprise afin de s'assurer de ne passer que par des routeurs au sein de l'Europe.
   Quelle structure de données est à privilégier afin de gérer chaque minute l'ensemble des routeurs par lesquels semblent transiter les paquets transmis ?

Exercice 8

La Notation Polonaise Inversée (NPI) permet d'écrire des opérations arithmétiques, sans utiliser de parenthèses. Ici, nous nous limiterons à des nombres entiers naturels et aux opérations + , - , * et / sur eux. Dans cette notation, les opérateurs sont écrits après les opérandes. Par exemple, l'expression $13*(3+2)$ donne en NPI :

$32+13*$¿.

On écrit et on exécute les opérations dans le sens des priorités mathématiques.

Dans cette notation, on réalise :

- L'addition entre 3 et 2 (3 2 +)
- La multiplication entre le précédent résultat et 13 (13 *)

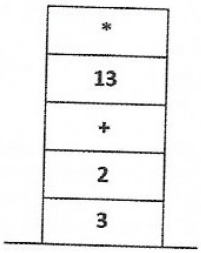
Ce qui donne le résultat.

1. Donner la File correspondant à la saisie NPI de l'exemple. Faire de même avec la Pile.

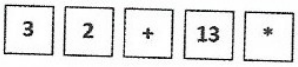
2. Quelle est la structure adaptée à la résolution de l'expression ? (Note : on remarquera qu'on doit toujours avoir deux opérandes pour un opérateur. Il faut stocker le résultat intermédiaire dans la structure pour effectuer la suite des calculs).

3. En utilisant les opérations du type abstrait Pile, proposer une fonction permettant d'afficher le résultat d'une expression en NPI. (On pourra considérer qu'on a déjà la fonction INVERSER8PILE(P) qui est dans l'ordre inverse de celle donnée en argument).

1. Après saisie, la pile peut être représentée par

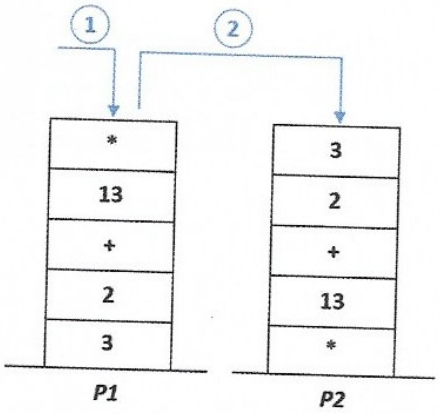


De même la file :



2. La File donne l'apparence que nos éléments peuvent sortir dans le bon ordre (par la gauche), cependant on ne pourra pas stocker dans la file, au bon endroit, le résultat intermédiaire (vu que l'enfilage se fait par la droite !). Ce n'est donc pas la structure adaptée.

Dans le cas de la Pile, on s'aperçoit qu'il faut inverser les éléments pour qu'ils puissent sortir dans le bon ordre. La saisie correspond à un empilage successif dans P1, pour inverser, il faut dépiler toute la pile P1 dans P2.



Précisions

Il faudra prendre les 2 premiers éléments de la pile (opérandes) et le troisième (opérateur) pour empiler le résultat intermédiaire et répéter jusqu'au résultat final.

3. En utilisant les opérations du type abstrait Pile et la fonction INVERSER(), on peut proposer la fonction suivante :

```
Fonction RESULTAT(P) :
    N = INVERSER(P)
    Resultat_trouve = faux
    Resultat = 0
    Tant que Resultat_trouve == Faux
        A=DEPILER(N)
        Si EST_VIDE(N)== Vrai :
                Resultat_trouve = Vrai
                Resultat = A
        Sinon :
                B= DEPILER(N)
                Op = DEPILER(N)
                Si Op == '+' :
                    EMPILER(N, A+B)
                Sinon :
                    Si Op == '-' :
                        EMPILER(N, A-B)
                    Sinon :
                        Si Op == '*' :
                            EMPILER(N, A*B)
                        Sinon :
                            EMPILER(N, A/B)
    Retourner Resultat
```