Pile

Exercice 1 : Pile de caractères

Une pile est une structure de données contenant des données de même type, ici caractère. Le principe d'une pile est que le dernier élément ajouté est le premier à en être retiré. Si on considère une pile d'assiettes, on pose une nouvelle assiette au sommet de la pile. Si on prend une assiette, c'est naturellement celle du sommet. On dit que la pile est une structure de données de type LIFO (Last In, First Out : dernier entré, premier sorti). Les opérations sur une pile sont les suivantes :

- Empiler (ajouter) un nouvel élément dans une pile.
- Dépiler une pile (supprimer le dernier élément ajouté). La pile ne doit pas être vide.
- Accéder à l'élément en sommet de pile. La pile ne doit pas être vide.
- Savoir si une pile est vide ou non.

Bien sûr , il faut aussi pouvoir initialiser une pile. La pile est alors vide. Une variable de type Pile ne peut être utilisée que si elle a été initialisée.

1. *Architecture.* Cette notion de pile pourra être utilisée dans différents programmes. Indiquer quel est le concept qui permet de réutiliser cette pile dans différents programmes. Rappeler ses principales caractéristiques.

2. *Spécifier la pile.* Écrire la spécification de la pile décrite ci-dessus.

3. *Tester la pile.* Écrire un programme de test de la pile qui :

- initialise une pile,
- empile successivement les caractères 'o', 'k', puis ' ?',
- vérifie que le sommet est ' ?',
- dépile 3 fois,
- vérifie que la pile est vide.

4. *Afficher un entier naturel.* Les opérations de la pile étant spécifiées, on peut écrire des programmes les utilisant même si l'implantation de ces opérations n'est pas encore définie.

4.1. Pour afficher un entier, on peut utiliser le sous-programme qui affiche un caractère et une pile de caractères pour conserver les caractères correspondant aux chiffres de l'entier. Écrire un tel sous-programme.

4.2. Aurait-il été possible d'écrire cette application sans utiliser la pile (ni une autre structure de données : liste, tableau...) ?

4.3. Comment faire si on veut utiliser ce sous-programme dans différents programmes ?

5. *Implanter la pile.* On choisit de représenter la pile par un enregistrement composé d'un tableau de 20 caractères (les éléments mis dans la pile) et un entier (le nombre d'éléments dans la pile). Le i^e élément ajouté dans la pile est à la i^e position du tableau.

Écrire l'implantation de la pile.

Exercice 2 : Généralisation de la pile

Dans l'exercice 1, nous avons défini une pile de caractères de capacité 20. En réalité, c'est l'utilisateur de la pile qui est le mieux placé pour décider de la capacité de la pile. Ce même utilisateur pourrait aussi souhaiter avoir une pile d'entiers, de réels, de dates, etc.

1. Expliquer comment faire pour permettre à l'utilisateur de la pile de choisir le type de ses éléments et sa capacité.

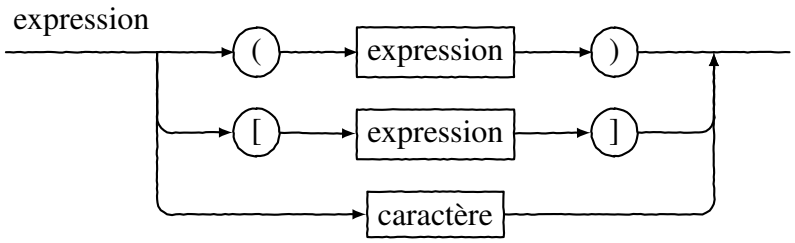
2. Adapter en conséquence le module ainsi que les programmes qui l'utilisent.

3. On souhaite disposer d'une nouvelle opération sur une pile : l'afficher sur le terminal. Ajouter cette opération et l'utiliser.

Exercice 3 : Expressions bien parenthésées

l'objectif de cet exercice est de vérifier que des expressions sont bien parenthésées par rapport à l'imbrication des parenthèses et des crochets.

Une expression est bien parenthésée si elle respecte le diagramme syntaxique suivant :



Sachant que « caractère » est n'importe quel caractère, à l'exception des parenthèses et des crochets bien sûr.

Voici quelques exemples d'expressions bien parenthésées :

```
2 * (x + y)
occurrences[chr(ord('0') + (entiers[i] mod 4))]
[ ( ( [ ] ) [ ( ) ( ) ] ) ]
```

et d'expressions mal parenthésées :

```
2 * (x + y))
occurrences[chr(ord('0')] + (entiers[i] mod 4))
[[[[
]
( ( ] )
( [ ) ]
```

Pour vérifier qu'une expression est bien parenthésée, on peut utiliser une pile. Les symboles ouvrant sont empilés et lorsque l'on rencontre un symbole fermant, on vérifie que le sommet de la pile contient bien le symbole ouvrant correspondant. Si ce n'est pas le cas, c'est que l'expression est mal formée. De plus, en fin d'analyse de l'expression, la pile doit être vide ce qui garantit que tous les caractères ouvrants ont été fermés.

Écrire un sous-programme qui vérifie si une chaîne de caractères est bien parenthésée et qui indique, si la chaîne est mal parenthésée, l'indice que caractère qui n'est pas appairé.