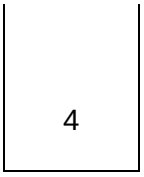
I2010 : langage C (15)

Les structures récursives

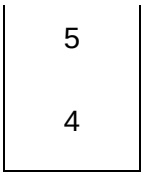
Ecrivez un programme `polonaiseInverse` pour évaluer le résultat d'une expression arithmétique écrite en notation polonaise inverse. Pareille évaluation se fait aisément en utilisant une pile. Il faut parcourir l'expression : lorsqu'un nombre est rencontré, il est mis sur la pile ; lorsqu'un opérateur est rencontré, il faut dépiler deux nombres, calculer le résultat de l'opération entre le premier nombre et le second et empiler le résultat.

Par exemple : **4 5 + 10 * 50 130 + /**

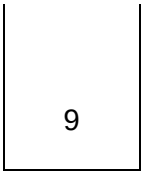
4 est mis sur la pile



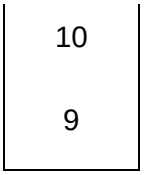
5 est ajouté au sommet de la pile



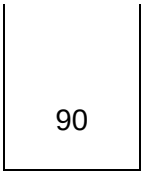
+ provoque le dépilage de 5 puis de 4 et la somme de 5 et 4 est mise sur la pile



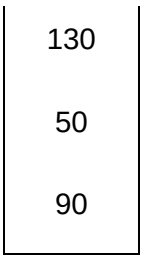
10 est ajouté au sommet de la pile



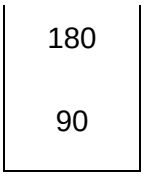
***** provoque le dépilage de 10 puis de 9 et le produit de 10 par 9 est mis sur la pile



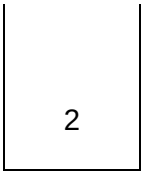
50 est ajouté au sommet de la pile, puis ***130***



+ provoque le dépilage de 130 puis de 50 et la somme de 130 et 50 est mise sur la pile



/ provoque le dépilage de 180 puis de 90 et le quotient, résultat de 180/90 est mis sur la pile



L'expression a été entièrement prise en compte, le résultat est le nombre qui reste dans la pile.

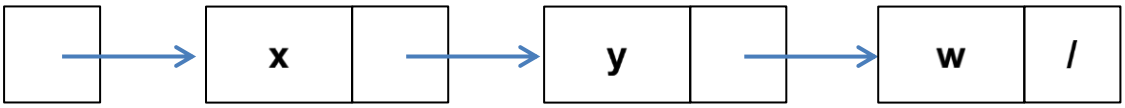
Voici l'**interface d'une pile** :

`Pile initPile ();`	qui renvoie une pile vide
`bool push (Pile*, int);`	qui renvoie vrai si l'entier a été placé sur la pile
`int pop (Pile*);`	qui retire l'entier du sommet de la pile et le renvoie
`bool pileVide (Pile);`	qui teste si la pile est vide
`void afficherPile (Pile);`	qui affiche le contenu de la pile

Schéma de la pile quand elle est vide :

Schéma de l'ajout d'un élément au début (sommet) de la pile:

La pile avant



L'élément à rajouter

L'ajout

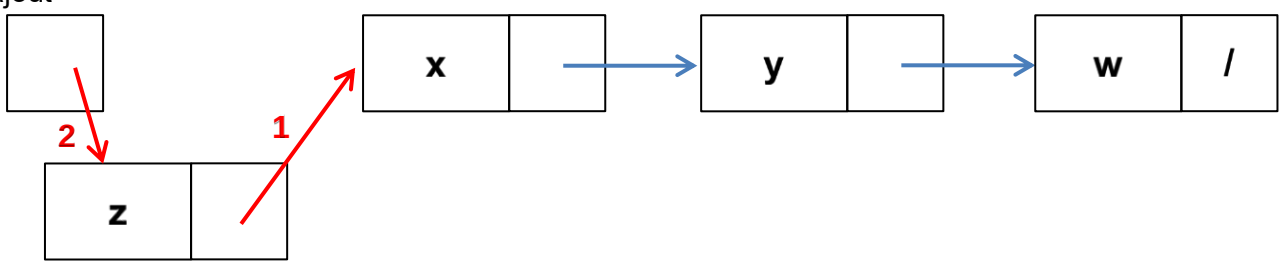
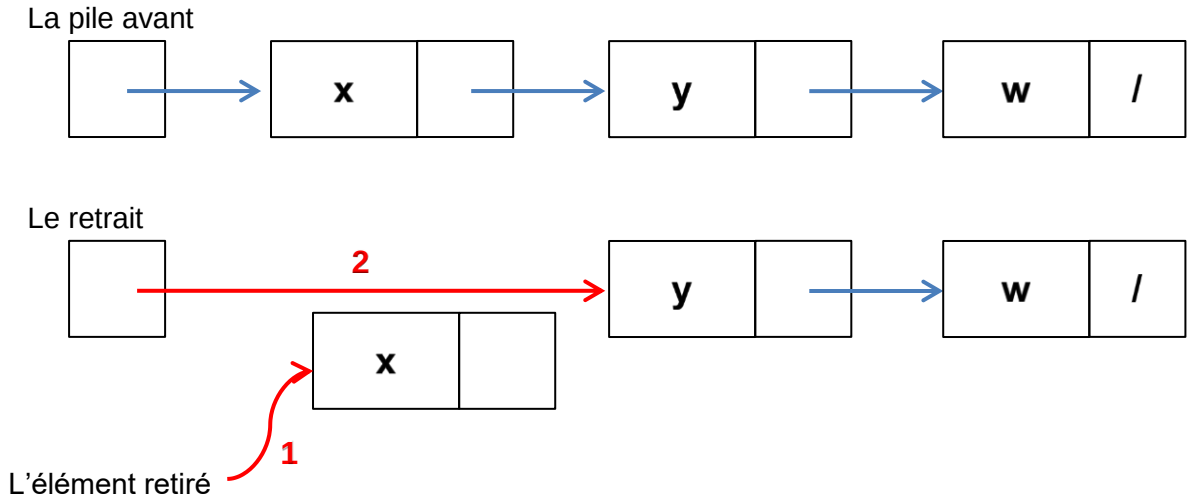


Schéma du retrait de l'élément du sommet de la pile:



1. Commencez par écrire un module *pile*. Celui-ci contiendra les fonctions nécessaires à l'implémentation d'une pile d'entiers à l'aide une liste chaînée, en respectant l'interface fournie.

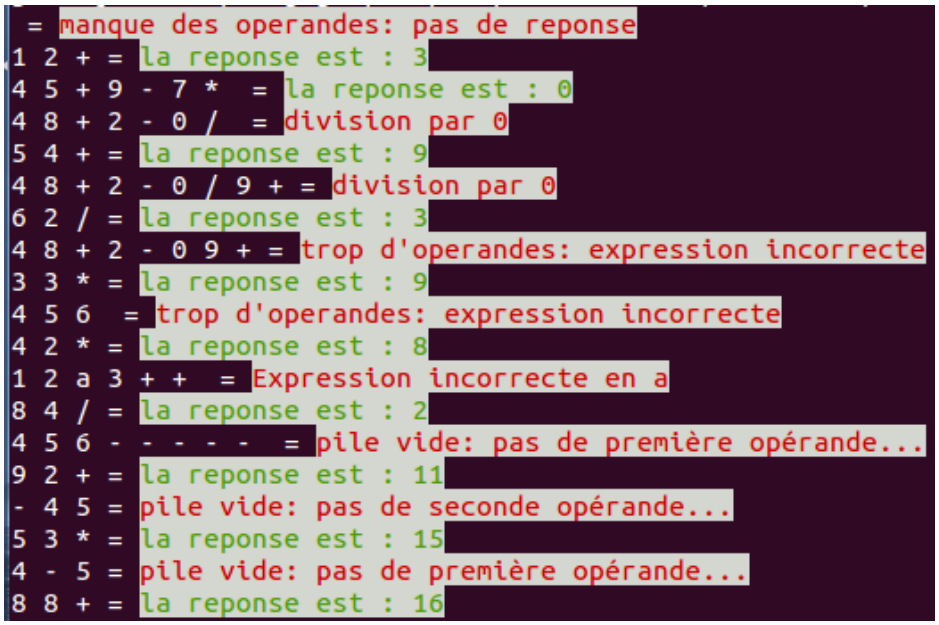
2. Testez votre implémentation avec un petit programme qui ajoute (*push*) et retire (*pop*) des éléments à une pile et affiche son contenu après chaque opération. Pour ce faire, écrivez un *makefile*.

3. Ecrivez ensuite un programme qui va implémenter l'interpréteur d'expressions en notation polonaise inverse. Ce programme doit :
   - lire à l'entrée standard des lignes d'au plus 256 caractères qui contiennent chacune une expression arithmétique en notation polonaise inverse ; pour simplifier le code, nous supposons que les nombres sont des entiers composés d'un seul chiffre et les opérateurs arithmétiques sont limités aux 4 suivants : +, -, *, /. Vous pouvez dans un premier temps vous limiter aux opérateurs d'addition et de soustraction.
   - vérifier que la ligne lue ne contient que les caractères autorisés grâce à la fonction `strspn`
   - évaluer l'expression lue
   - afficher le résultat.

   Testez votre programme avec le fichier `npi.txt` :

   ```
   npi < npi.txt
   ```

   Voici un exemple d'exécution avec ce fichier de test :


= manque des operandes: pas de reponse
1 2 + = la reponse est : 3
4 5 + 9 - 7 * = la reponse est : 0
4 8 + 2 - 0 / = division par 0
5 4 + = la reponse est : 9
4 8 + 2 - 0 / 9 + = division par 0
6 2 / = la reponse est : 3
4 8 + 2 - 0 9 + = trop d'operandes: expression incorrecte
3 3 * = la reponse est : 9
4 5 6 = trop d'operandes: expression incorrecte
4 2 * = la reponse est : 8
1 2 a 3 + + = Expression incorrecte en a
8 4 / = la reponse est : 2
4 5 6 - - - - - = pile vide: pas de première opérande...
9 2 + = la reponse est : 11
- 4 5 = pile vide: pas de seconde opérande...
5 3 * = la reponse est : 15
4 - 5 = pile vide: pas de première opérande...
8 8 + = la reponse est : 16