TD ADS 1125– Applications Piles&Files

1. Expressions Postfixée

a. Lecture d'une nombre entier tapé au clavier

Proposez un pseudo code qui transforme une suite de caractères numériques tapés au clavier et appartenant à ['0' - '9'] en nombre entier. Il ne faut pas faire une fonction mais simplement traiter ce problème. L'utilisateur ne reçoit pas de message, il tape juste des caractères.

Vous disposez de la fonction car <- **lireCar**() qui permet de lire un caractère tapé au clavier.

L'interprétation des chiffres tapés au clavier s'arrête quand le caractère n'est pas un chiffre comme par exemple un espace (code ASCII = 32) ou la touche entrée (caractère '**\r**' équivalent du code ASCII=13).

La fonction int **atoi**(char *mot) est fournie et fera la conversion d'une chaine en valeur entière.

b. Calcul d'une expression postfixée

Une expression arithmétique dite **postfixée** est une expression pour laquelle on écrit **d'abord** les opérandes **puis** l'opérateur. Ainsi, l'expression postfixée `12 30 +` équivaut à l'expression **infixée** `(12 + 30)` et vaut 42.

Remarque: 12 et 30 sont ici appelés les opérandes et + est l'opérateur.

L'expression postfixée `2 3 + 2 *` équivaut à l'expression **infixée** `((2 + 3) * 2)` et vaut 10.

L'expression postfixée `12 3 + 3 / 1 - 3 *` équivaut à l'expression infixée `(((12 + 3) / 3) – 1 ) * 3)` et vaut 12.

L'expression infixée est la manière courante d'écrire une expression arithmétique, les priorités d'opérations sont gérées avec des parenthèses. L'expression postfixée est plus connue sous le nom de « notation polonaise inversée » et ne nécessite pas l'usage des parenthèses. C'est l'ordre d'écriture des opérandes et des opérateurs qui fixe les priorités.

On choisit de résoudre le calcul d'une expression arithmétique **postfixée** à l'aide d'une **pile**. En effet, on observe qu'un calcul partiel d'une expression postfixée peut être effectué quand apparait un opérateur. Dès lors, il faut retrouver les deux opérandes saisis auparavant et faire l'opération. Le résultat ainsi obtenu est alors mis en attente de l'arrivée d'un autre opérateur.

L'idée est donc d'utiliser une pile afin de stocker les opérandes en attendant la survenue d'un opérateur, de récupérer les opérandes quand un opérateur est saisi, de faire l'opération et d'empiler le résultat qui sera utilisé comme un nouvel opérande pour une future opération.

L'expression contient soit des caractères opérateurs « **+** , **-** , **/** , ***** » soit des nombres entiers contenant des chiffres de 0 à 9. Le caractère **espace** est utilisé comme séparateur entre les nombres.

Proposer un programme en **langage C** afin d'effectuer le calcul d'une expression arithmétique **postfixée** entrée au clavier caractère par caractère. La saisie du caractère spécial CR (code ASCII =13) indique que l'expression a été complètement saisie et qu'on peut finaliser le calcul. En pseudocode, utiliser la fonction **readChar**() pour lire un caractère taper au clavier. La fonction `char readChar()` est disponible dans l'outil ConsoleTools fourni. Cette fonction renvoie le caractère alphanumérique tapé au clavier sans qu'on soit obligé de valider la saisie

2. Expression parenthésée en expression postfixée

On souhaite transformer une expression arithmétique parenthésée en expression dite postfixée : Par exemple, l'expression parenthésée ((((2 + 3) * 5) * 2) * 5)

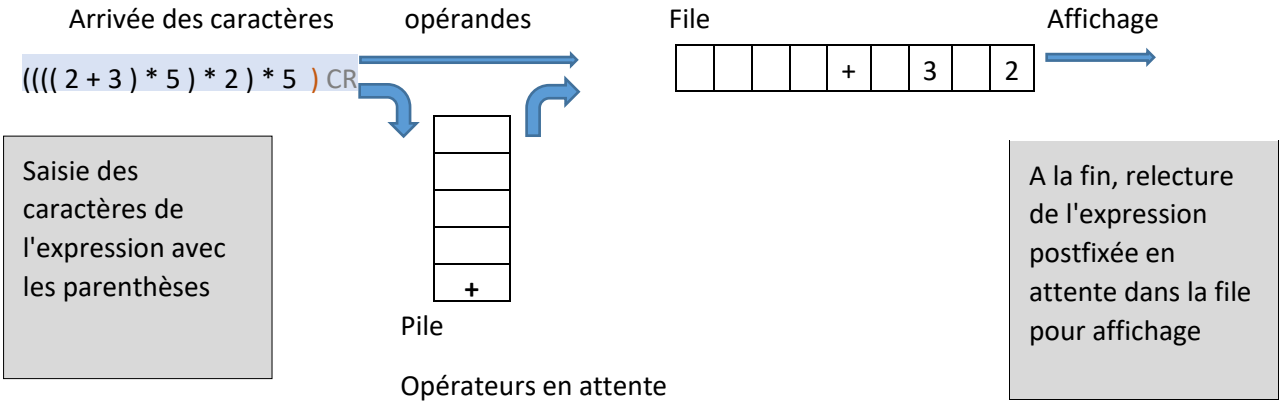
correspond à l'expression postifixée 2 3 + 5 * 2 * 5 * dans laquelle les opérateurs sont placés après les opérandes.

Pour cela on a besoin d'une pile et d'une file. La file sert à reconstituer l'expression postfixée. On y pousse les chiffres des opérandes dans l'ordre d'arrivée. Les opérateurs sont mis en attente dans la pile. Quand une parenthèse fermante est tapée au clavier on récupère un opérateur au sommet de la pile pour le pousser dans la file. Quand il n'y a plus d'entrée au clavier on termine de dépiler les opérateurs s'il en reste, pour les mettre dans la file.

De plus, à chaque fois qu'apparait une parenthèse fermante ou un opérateur on place un caractère espace dans la file.

La lecture des caractères se fait à l'aide de la fonction car = lireCar(); La lecture s'arrête quand on tape 'entrée', ce qui correspond au caractère spéciale CR.

Le code des fonctions qui servent à créer et manipuler piles et files ne doit pas être détaillé.



Question :

- Exprimer en pseudo-langage l'ensemble de ce traitement.
- Programmer et tester en langage C cet algorithme.