TP 2 C++

Le but de ce TP est de réaliser une classe structure de pile traitant des données de type `char`. La structure de pile sera implantée sous forme d'une classe, appelée PiledeChar.

- Créer trois fichiers : `tp2ex1.cpp`, `piledechar.h` et `piledechar.cpp`. Les fichiers `piledechar.h` et `piledechar.cpp` contiennent respectivement la déclaration de la classe `PiledeChar`, et les définitions des méthodes de la classe. Le fichier `tp2ex1.cpp` contient la fonction `main()` (chaque fichier `.cpp` effectue l'inclusion du fichier `.h` afin d'apporter au compilateur les informations relatives au contenu de la classe).
  Ecrire la déclaration de classe de `PiledeChar` dans le fichier `piledechar.h`, en tenant compte des informations suivantes. La classe contient trois données membres privées : deux entiers strictement positifs, nommés `mMax` et `mSommet`, et un pointeur sur `char mPile`. La donnée membre `mMax` contient la taille de la pile créée. La donnée membre `mSommet` indique le numéro de la case vide dans laquelle on pourra empiler le prochain caractère. Le pointeur sur un caractère `mPile` désigne le tableau de caractères, alloué dynamiquement.
  Déclarer et écrire les différents constructeurs de la classe. La taille max de la pile sera par défaut 100. Définir aussi le constructeur de copie.
- Ecrire le destructeur, chargé de libérer les ressources allouées par chaque instance de classe.
- Ecrire un méthode `ComputerElements()` qui donne retourne un entier positif qui correspond au nombre d'éléments actuellement présents dans la pile.
  Ecrire la méthode `AfficherPile()` qui affiche entre des '[' et ']' les éléments actuellement présents dans la pile.
- Ecrire une méthode `EmpilerElem()` qui prend un caractère en paramètre et le place sur le dessus de la pile.
  Ecrire une méthode `DesempilerElem()` qui retourne le caractère du dessus de la pile.
  Ecrire dans le fichier `tp2ex1.cpp` la fonction `main()`. Déclarer un tableau de caractère et demandez à l'utilisateur d'entrer un mot au clavier. On utilisera la méthode `get()` associée à l'instruction `cin` sans oublier les trois paramètres suivants : Le tampon à remplir, le nombre maximal de caractères à lire et le délimiteur désignant la fin de la saisie. Par défaut le délimiteur de saisie est le caractère newline `'\n'`. Le `cin` rajoute automatiquement le caractère `'\0'` à la fin de la chaîne de caractère. Les lignes suivantes illustrent le processus :

  ```
  char buffer [80];
  cin.get(buffer,79);
  cout<<"chaine : "<<buffer<<endl;
  ```

  Attention, le caractère délimiteur n'est pas récupéré. Il reste donc dans le flux et sera retiré lors d'un autre appel ultérieur à `cin`, provoquant des effets indésirables. On peut éviter cela en utilisant la méthode `cin.getline(buffer, 79)` à la place de `cin.get(buffer, 79)`.
  Créer une instance de la classe `PiledeChar`. Le programme lit l'entrée clavier et ensuite découpe la chaîne de caractère lettre par lettre et les place dans la pile. Remarque : le caractère de retour chariot `'\n'` qui permet à l'utilisateur de valider la ligne reste présent dans le chaîne de caractère. Si un caractère de fin est rencontré avant le nombre maximal de caractères autorisés, un caractère fin de chaîne est rajouté.
- Pour surveiller l'évolution de la pile à chaque empilage, afficher le contenu de la pile après avoir empilé chaque lettre.
  Ecrire une fonction `afficheinverse()` recevant en paramètre une pile et qui dépile les lettres qui y sont contenues en les écrivant à l'écran au fur et à mesure. Cette fonction ne doit pas modifier le contenu de la pile déclarée dans le `main()`. Utiliser cette fonction pour écrire à l'envers le message

tapé par l'utilisateur.
Ecrire une fonction `inversemajuscule()` qui reçoit une instance de la classe `PiledeChar` en paramètre, qui crée une nouvelle pile `nPile`, dépile les lettres une par une pour ensuite les rempiler en majuscule dans `nPile`.
Afin de vérifier le programme, ajouter dans chaque constructeur et destructeur, l'affichage d'un message indiquant que la construction/destruction a lieu.