TP.1

[Mise en œuvre d'une classe]

L'objectif de ce TP consiste à écrire le programme d'une classe `MessageCode`. Cette classe contiendra trois champs privés : un champs `message` (une chaîne de caractère - objet `string`[1]) dont le contenu ne sera jamais accessible à l'utilisateur, un champs `messageCode` (une chaîne de caractère - objet `string`) qui correspondra à une version codé du message (accessible à tous) ainsi qu'un champs `clef` qui sera un entier permettant de coder un `message` et/ou de décoder un `messageCode`.

I . Squelette du programme et création de la classe

1. Créer un projet qui contiendra trois fichiers `main.cpp`, `messageCode.hpp` et `messageCode.cpp`.
2. On devra définir la classe `MessageCode`. Dans quel fichier cela devra-t-il être fait ? On devra définir les fonctions membres dans quel fichier cela devra être fait ? On réalise un programme. Dans quel fichier devra-t-on écrire les instructions qui seront réalisées par l'exécutable ? [Ici le programme principal, fonction `main`, servira simplement à tester/vérifier que la classe (et ses fonctions membres) sont implémentées correctement. Il devra donc faire appel à chacune des fonctions publiques de la classe.]
3. Ici, on va définir la classe `MessageCode`. Cette classe contiend un constructeur par défaut qui initialise les trois champs d'un objet (sans recevoir d'argument). [Les deux chaînes de caractères (`string`) seront vides. Le champs clef sera initialisé à une valeur aléatoire entre 1 et 25.]
   (a) Réaliser ce constructeur. [Vérifier autant que possible qu'il exécute correctement sa tache.]

   Cette classe contiend un destructeur. Que fait-il ?
   (b) Réaliser ce constructeur. [Vérifier qu'il est bien appelé.]

Pour accéder à chacun des champs, on va créer deux fonctions membres (`setNomDuChamps` et `getNomDuChamps`).

4. Pour chacune des champs de la classe, expliquer pourquoi ces fonctions sont publiques (accessibles en dehors de la classe) ou privée (propre à la classe).
5. Réaliser ces six fonctions membres.

Pour se simplifier la vie (et pouvoir vérifier ce que l'on fait), on va créer une fonction `afficheMessageCode` qui permet d'afficher à l'écran les valeurs des membres d'un objet.

6. Expliquer pourquoi cette fonctions sera publique (accessibles en dehors de la classe) ou privée (propre à la classe).
7. Réaliser cette fonction. [Vérifier autant que possible qu'elle s'exécute correctement.]

II . Création de quelques constructeurs supplémentaires

En langage C la chaîne de caractère n'existe par en tant que telle. Elle correspond à un tableau de caractère se terminant par un caractère spécial. En C++, il existe une classe `string` permettant de gérer les chaînes de caractères aisément. La taille de la chaîne peut évoluer dynamiquement. Elle est associée avec un certain nombre de fonctions prédéfinies qui rendent son usage simple.
Dans cette partie, nous allons définir deux constructeurs l'un basé sur un tableau de caractère constant (connu à la compilation) alors que l'autre, reposant sur la classe `string`, pour lequel l'objet pourra être évalué dynamiquement lors de l'exécution du programme. [Dans cette partie, on laissera pour l'instant le champs `messageCode` vide. On devra revenir sur cette partie pour la compléter une fois que la fonction de codage aura été réalisée.]

8. Création d'un constructeur à partir d'un objet constant de type `string`. [Vérifier autant que possible qu'il exécute correctement sa tache.]
9. Création d'un constructeur à partir d'une chaîne de caractère constante (de type `char[]`). [Vérifier autant que possible qu'il exécute correctement sa tache.]

III . Création du message codé

Ici on veut déterminer le message codé, pour cela on devra implémenter un algorithme de chiffrement par décallage[2] (aussi appelé code ou chiffre de césar. Il s'agir d'une méthode de chiffrement très simple.
Le texte chiffré s'obtient en remplaçant chaque lettre du texte clair original par une lettre à distance fixe, toujours du même côté, dans l'ordre de l'alphabet. Pour les dernières lettres (dans le cas d'un décalage à droite), on reprend au début. Par exemple avec un décalage de 3 vers la droite, A est remplacé par D, B devient E, et ainsi jusqu'à W qui devient Z, puis X devient A etc. Il s'agit d'une permutation circulaire de l'alphabet. La longueur du décalage, 3 dans l'exemple évoqué, constitue la clé du chiffrement qu'il suffit de transmettre au destinataire - s'il sait déjà qu'il s'agit d'un chiffrement de César - pour que celui-ci puisse déchiffrer le message. Dans le cas de l'alphabet latin, le chiffre de César n'a que 26 clés possibles (y compris la clé nulle, qui ne modifie pas le texte). [Ici on supposera que le message est une lettre de l'alphabet.]

1. http://www.cplusplus.com/reference/string/string/
2. https://fr.wikipedia.org/wiki/Chiffrement_par_d%C3%A9calage

1. Pourquoi a-t-on demandé précédemment que la clef soit un entier compris entre `1` et `25` ? [Expliquer].
2. Réaliser une fonction membre `determinerLeMessageCode` qui permette, à partir des champs `message` et `clef`, de déterminer le champs `messageCode`. [Vérifier autant que possible qu'elle exécute correctement sa tache.]
3. Reprendre le code existant pour que les constructeurs prennent bien en compte cette modification.

Jusqu'ici on ne s'est pas intéressé au message. On a même supposé que le message était composé uniquement de lettres de l'alphabet. Dans les faits, cela ne sera pas nécessairement le cas. Il va donc falloir, au moment du codage, détecter les lettres de l'alphabet qui seront codées des autres caractères qui seront laissés inchangées.

4. Modifier la fonction membre `determinerLeMessageCode` qui devra distinguer les lettres de l'alphabet des autres caractères. [Vérifier.]

IV . Création d'un message à partir d'un fichier

On veut remplir un message à partir d'un fichier (`romeoAndJuliet_1112.txt`). Pour cela, on va devoir lire les données à partir d'un fichier. Pour cette partir on s'inspirera de ce qui est proposé sur le lien suivant http://www.cplusplus.com/doc/tutorial/files/.

5. Proposer une fonction membre `chargerAPartirDUnFichier`. Le message résultant devra être affiché à l'écran (et plus ou moins mis en forme). [Vérifier.]