TP N° : 8

Exercice 1:

I- On appellera *dimension* d'une pile le nombre maximal d'éléments qu'elle peut contenir et *taille* d'une pile le nombre d'éléments qu'elle contient réellement. Le tableau représentant la pile est donc indexé de 0 (bas de la pile) à *taille-1* (haut de la pile). L'élément que l'on peut dépiler est donc dans la case d'indice *taille-1*.
Ecrire le fichier Pile.cpp correspondant au fichier Pile.h suivant :

```
#ifndef Pile H
#define Pile H
class Pile
{ public :
        Pile(int t = 10); //Constructeur qui construit une pile de dimension t (10 par defaut)
        Pile(const Pile &); //Constructeur par recopie
        ~Pile(); //Destructeur
        void empile(int); //empile n en haut de la pile
        void depile(); //depile le sommet de la pile
        int donnetaille() const; //renvoie la taille de la pile
       Pile & operator=(const Pile &);//surcharge de l'opérateur d'affectation
        bool operator==(const Pile &);//surcharge de l'opérateur ==

private :
        int dim;
        int taille;
        int *adr;
};
#endif
```

II- Remplacer la fonction membre « empile » par l'opérateur < et la fonction membre « depile » par l'opérateur --.

p < n ajoute la valeur n sur la pile p
--p supprime la valeur du haut de la pile p.

III- Ajouter à la classe Pile la surdéfinition de l'opérateur [], de sorte que la notation p[i] ait un sens et retourne l'élément d'emplacement i de la pile p.
Utiliser ce nouvel opérateur pour écrire les fonctions affiche et saisie
On créera donc une fonction membre de prototype int &Pile::operator[](int i);
IV- En utilisant les fonctions amies, donner une signification aux opérations :

p+x et x+p où p désigne une pile et x un entier.(la somme d'une pile et un entier est obtenue en ajoutant à chaque élément de la pile l'entier x).

V- Ajouter à la classe pile la surcharge des opérateurs << et >>.

Exercice 2:

Définir une classe chaine permettant de créer et de manipuler une chaîne de caractères:

données:

- adresse d'une zône allouée dynamiquement (inutile d'y ranger la constante \0)

méthodes:

- constructeur chaine() initialise une chaîne vide
- constructeur chaine(char *texte) initialise avec la chaîne passée en argument
- constructeur par recopie chaine(const chaine &ch)
- opérateurs affectation (=), comparaison (==), concaténation (+), accès à un caractère de rang donné ([]), << et >>.

Exercice 3:

En vue de la gestion d'une bibliothèque on vous demande d'écrire une application pour traiter des livres. Chaque Livre possède un code (un entier), un titre (char*) et un auteur (char*).

1) Définissez la classe Livre, attributs et méthodes : constructeurs, destructeur, constructeur par recopie, opérateur d'affectation, méthode de saisie, méthode d'affichage, méthode de modification.

2) Définissez les attributs de la classe BIBLIO (Bibliothèque) : un tableau dynamique de livres, leur nombre, et le nombre maximal de livres pouvant y être rangés.

3) Définissez les méthodes suivantes :

1- Ajoutez le constructeur à la classe Biblio avec un paramètre le nombre maximal de livres pouvant y être rangés. Par défaut, la bibliothèque ne contient aucun livre.

2- Ajoutez le constructeur par recopie et surcharger l'opérateur d'affectation.

2- Ajoutez une méthode donnant le contenu de la bibliothèque.

3- Ajoutez une méthode ajouter qui ajoute à la bibliothèque le livre donné en argument et renvoie true (false en cas d'échec).

4- Ajoutez une méthode Rechercher qui parcourt la bibliothèque à la recherche du livre dont le code est donné.

5- Ajouter une méthode Supprimer qui supprime un livre de la bibliothèque dont le code est donné en argument.

6- Ajoutez une méthode Rechercher_Aut qui recherche tous les livres de la bibliothèque ayant le même auteur donné en argument.

7- Ajoutez un destructeur à la classe BIBLIO.