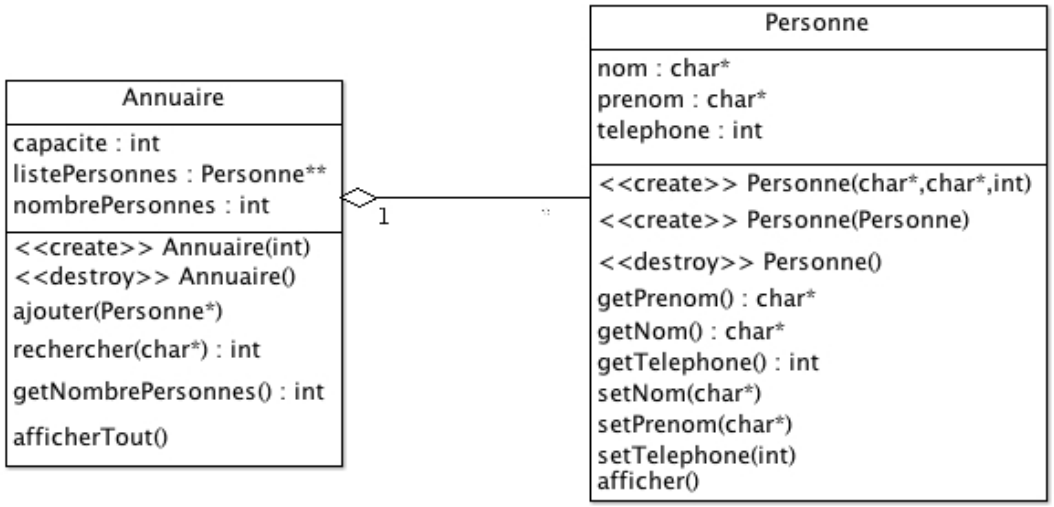
TP 1-2

Partie 1

1 Introduction

Le but de ce TP est de savoir écrire, compiler et exécuter un programme simple en C++ utilisant quelques classes.
Nous allons construire un annuaire de personnes. Pour cela, nous allons d'abord créer une classe **Personne**, puis une classe **Annuaire**, qui contiendra un tableau de Personne, et une méthode d'ajout, d'affichage et de recherche. Le programme est résumé par le diagramme UML suivant.



Exercice 1

- écrire les fichiers Personne.h et Personne.cpp permettant de définir la classe Personne.
- écrire le fichier main.cpp où l'on créera un objet Personne, et l'on affichera son nom et son prénom.

compilez puis exécutez.

Aide

- Vous disposez déjà du fichier main.cpp, Personne.h, et de l'ossature du fichier Personne.cpp. Vous n'avez qu'à remplir les fonctions du fichier Personne.cpp
- Pour copier un char* dans un autre char*, utilisez la fonction **strcpy** de la bibliothèque **string.h** (pour l'utilisation, référez vous au manuel : **man strcpy**)
- pour l'affichage, vous pouvez utiliser **cout**.
  Exemple : **cout « "valeur de la variable i : " « i « endl;**

Exercice 2

- écrire les fichiers Annuaire.h et Annuaire.cpp permettant de définir la classe Annuaire.
- créer un Annuaire de capacité 100 et tester les méthodes de la classe :
  - ajouter la personne créée précédemment
  - ajouter une seconde personne
  - afficher l'annuaire
  - rechercher une personne

Aide

- **listePersonnes** est un tableau de pointeurs de Personne. Dans le constructeur, on peut donc allouer dynamiquement un tableau de pointeurs de Personne de taille **capacite**.
- ne pas oublier d'initialiser le compteur (variable **nombrePersonnes**)
- dans le destructeur, on ne fait rien car les objets utilisés peuvent continuer à vivre à l'extérieur de la classe
- dans la fonction **ajouter(Personne*)**, on ajoute la nouvelle personne à la dernière case du tableau (ne pas oublier d'incrémenter le compteur...)
- pour la méthode **rechercher(char*)** :
  - on recherche une personne à partir de son **nom**, on retourne son téléphone s'il existe, ou -1 s'il n'existe pas

  - on peut par exemple définir une méthode privée qui prend en entrée un nom et retourne le numéro de la case du tableau qui contient cette personne
  - il suffit de parcourir le tableau **listePersonnes** (boucle pour)
  - pour comparer deux char*, on peut utiliser la fonction **strcmp** (pour l'utilisation, se référer au manuel : **man strcmp**)

- dans la fonction **afficherTout()**, on peut utiliser la fonction **afficher()** de la classe Personne

Bonus

- Plutôt que d'utiliser les fonctions **afficher** et **afficherTout**, redéfinir l'opérateur « pour les classes Personne et Annuaire

Partie 2

Exercice 3

- Créer deux classes Ami et Pro **héritant** toutes les deux de Personne. Ces classes ont un ou plusieurs attributs en plus, par exemple un Ami a un *surnom*, et un Pro a un *numBureau* (voire plus selon imagination...)

- Ajouter une méthode *Contacter()* à la classe Personne qui pourra par exemple afficher *"untel a été contacté"*

- **Redéfinir** cette méthode dans les classes Ami et Pro (afficher par exemple *"toto a été contacté de manière amicale"*, *"toto a été contacté de manière professionnelle"*, ou utilisez votre imagination...!)

- ajouter une méthode contacter à la classe Annuaire, qui prendra en entrée un nom, et contactera la personne si elle existe, ou affiche un message d'erreur sinon.

- modifier le main pour créer des objets de type Ami et Pro, les ajouter dans l'annuaire, et tester la méthode contacter de l'annuaire (vérifier que la bonne fonction est appelée).