

TP n°2

Classes

Note : ces exercices nécessitent la lecture des supports des quatre premiers cours que l'on peut retrouver à l'adresse <http://www.liafa.univ-paris-diderot.fr/~yunes/cours/cpp/>

Exercice 1 (Une première classe)

Écrire une classe `Date` qui représente une date avec constructeurs adéquats et une méthode `void afficher()` qui affiche la date à l'écran au format `jj/mm/aaaa` (ou équivalent). Écrire une méthode `Date Date::demain()` qui renvoie la date du lendemain. Ces `Dates` pourront être manipulées diversement, par exemple en employant une des méthodes :

- `ajouteJours(unsigned int n);`
- `ajouteMois(unsigned int n);`
- `ajouteAnnees(unsigned int n);`
- et leurs équivalents `enleve*(unsigned int n)`.

Note : inutile de prendre en compte les années bissextiles et autres dérivations temporelles. On considèrera donc que février est un mois de 28 jours.

Testez votre classe avec un programme de test de votre cru.

Exercice 2 (Une seconde classe, des constantes et des statiques...)

Écrire une classe `Duree` qui représente une durée exprimée en jours avec constructeurs adéquats et également une méthode `afficher` qui affiche la durée en années et jours (pas les mois). Écrire une méthode `Duree Date::dureeJusquA(Date d)` qui renvoie la durée jusqu'à la date `d` (attention les durées peuvent donc être « négatives »). Définir des méthodes permettant de modifier une durée y ajoutant ou retranchant des jours, mois ou années. Définir des méthodes permettant d'appliquer un coefficient multiplicateur à une durée de sorte que l'on puisse obtenir une durée, par exemple, 3 fois plus longue qu'une autre.

Dans la fonction `main` calculer la durée jusqu'à Noël de la façon suivante :

```
Date noel(24, 12, 2012);
Date aujourd'hui(15, 10, 2012);
Duree d = aujourd'hui.dureeJusquA(noel);
d.afficher(); // ça m'affiche : 0 ans, 70 jours
```

Modifier la classe `Date` de sorte que l'on puisse définir des dates constantes (Noël, la fin du monde Maya 12/12/2012, etc.). Vérifier que l'on ne peut pas modifier une date constante. Faire en sorte que vos classes `Date` et `Duree` permettent d'exécuter le bout de code suivant :

```
const Date eow(12,12,2012);
Date d(5/5/2005);
Duree duree = d.dureeJusquA(eow);
d.ajouter(5);
duree.multiplierPar(2);
d.ajouter(duree);
d.afficher();
```

Modifier vos classes de sorte que l'on ait à disposition dès le départ quelques constantes utiles : une durée de 30 jours de nom `TRENTE_JOURS`, le jour de l'an `NEW_YEAR_S_DAY`, etc. Les utiliser pour obtenir la date situé 60 jours après le premier de l'an 2013.

Exercice 3 (Construction par copie)

Soit le programme suivant :

```
#include <iostream>
class Test {
private:
    int *ptr;
public:
    Test(int n);
    ~Test();
    void affiche();
};
Test::Test(int n=100) {
    ptr = new int[n];
    for (int i = 0; i<100; i++) { *(ptr + i) = i; }
}
Test::~~Test() {
    delete[] ptr;
}
void Test::affiche() {
    std::cout << "valeur:_" << *(ptr+5) << std::endl;
}
void uneFonction(Test t) {
    t.affiche();
}
int main() {
    Test x;
    uneFonction(x);
    int *p = new int[10];
    p[5] = 555;
    x.affiche();
    delete p;
    return 0;
}
```

1. Compilez et exécutez le programme. Expliquez ce qui se produit en analysant soigneusement le code. Qu'y a-t-il d'anormal ?
2. Peut-t-on corriger sans modifier la classe `Test` ? Songez aux références ou au pointeurs... Cette correction est-elle raisonnable ?
3. Rajoutez un constructeur de copie à la classe `Test`. Testez que la classe a un comportement normal. Attention : pour fonctionner correctement la classe nécessite quelques améliorations/corrections.

Exercice 4 (Modélisation d'une banque)

Une banque compte plusieurs agences réparties sur le territoire français. Une banque est caractérisée par son nom commercial, le nom de son directeur général, son capital global et de

l'adresse de son siège social. Le directeur général est identifié par son nom, son prénom et son revenu. Une agence a un numéro d'agence et une adresse. Chaque agence emploie plusieurs employés, qui se caractérisent par leurs nom, prénom et date d'embauche. Les employés peuvent demander leur mutation d'une agence à une autre, mais un employé ne peut travailler que dans une seule agence. Les employés d'une agence ne font que gérer des clients. Un client ne peut avoir des comptes que dans une seule agence de la banque. Chaque nouveau client se voit systématiquement attribuer un employé de l'agence (conseiller). Les clients ont un nom, un prénom et une adresse. Les comptes sont de nature différente selon qu'ils soient rémunérés ou non (comptes courants). Les comptes rémunérés ont un taux d'intérêt et rapportent des intérêts versés annuellement.

Une relation particulière lie l'agence, le client, l'employé et le compte. De quelle relation s'agit-il ?

1. Donnez un diagramme de classes pour la modéliser.
2. Créer les classes suivantes :
Banque, Directeur, Agence, Employé, Client, Compte Courant, Compte d'épargne
3. Testez votre programme.