C++ : Compte rendu de TP

COSIALLS Julien et BEZIER Charlène

3A ROB

Table des matières

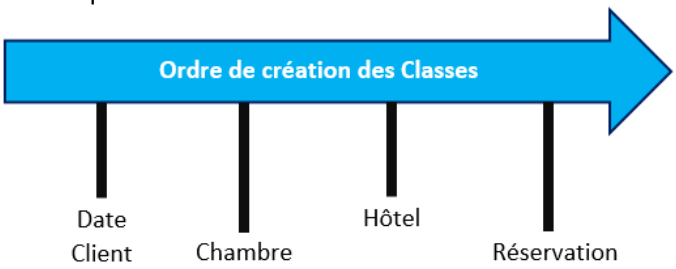
Introduction générale

Nous avons débuté les travaux pratiques par la création d'un dépôt git, puis d'un makefile pour nos deux TP. Durant ces travaux, nous avons utilisé les logiciels Visual Studio, Visual Code et Sublime Text.
Avant d'écrire la moindre ligne de code, nous réfléchissons à la manière dont il faut écrire les différentes classes : il faut commencer par coder les petites classes, car certaines classes ont besoin d'utiliser une ou plusieurs autres petites classes.
Ceci étant fait, il faut déterminer les éléments qui vont composer ces classes (entre autres, les variables).
Lors de l'écriture des différentes fonctions, il est nécessaire de s'assurer qu'elles soient fonctionnelles à l'aide de programmes de tests.
Attention, ce n'est pas parce qu'un objet est créé et que le code fonctionne, qu'il est forcément correct ! Il faut également s'assurer de la validité de celui-ci !
Il faut de plus s'assurer que les informations entrées correspondent bien à leurs types : c'est-à-dire que si l'on doit entrer par exemple une date, ce sont bien des nombres qui sont entrés et non des lettres.

Nous allons à présent réaliser une application de gestion de réservation d'un hôtel : nous allons voir la manière dont les classes sont organisées entre elles, ainsi que la manière dont nous allons interpréter les différents problèmes. Concernant la répartition du travail, nous avons travaillé en parallèle en même temps. Nous avons communiqué concernant la manière dont on pouvait résoudre les problèmes qui arrivaient d'un coup, en se mettant d'accord sur la manière de les résoudre après avoir fait la liste des méthodes de résolution.

TP Hôtel (Débutant)

L'objectif de ce TP est de mettre en place une application de réservation d'un hôtel. Durant ce travail, il faut hiérarchiser par ordre de priorité la création des différentes classes, car en effet, une grande classe telle que la classe Réservation a besoin d'utiliser plusieurs classes.
Voici la chronologie dans la laquelle nous avons créé les classes:



Les classes Date et Client peuvent être créées en même temps, car elles ne nécessitent pas l'utilisation d'autres classes. La classe Chambre avait besoin de Date (et de Réservation) et la classe Hôtel avait besoin de Chambre. Bien évidemment, des modifications ont été rajoutées dans certaines Classes au fur et à mesure.

Description:

- Classe Date: La classe date à 3 variables de classe qui sont : le jour, le mois et l'année. Il faut aussi s'assurer que chaque date créée existe. Pour cela, nous utilisons des fonctions qui s'assurent de la validité du jour (entre 28 et 31) et du mois (entre 1 et 12), mais pas de l'année, car nous estimons que n'importe quelle année peut fonctionner.

- Classe Client : Pour la classe client il faut : le nom et le prénom. Mais les deux seuls ne sont pas suffisants, d'où la nécessité d'un identifiant. Pour cela, il a été décidé que l'identifiant devait être saisi à la main lors de l'enregistrement du client.

- Classe Chambre : Il faut s'assurer que le prix des chambres ne soit pas négatif (on veut du profit !) et idem pour leur numéros. De plus, nous devons faire attention aux chambres disponibles : il sera impossible de réserver une chambre déjà occupée.

- Classe Hôtel : Pour identifier un hôtel, le nom, la ville et un identifiant sont suffisants. En effet, il n'y aura jamais d'Hôtels ayant exactement le même nom (à moins de faire partie d'un groupe d'hôtels, comme Kyriad). Il doit également être possible d'ajouter ou bien d'enlever des chambres en fonction de leur disponibilité.

- Classe Réservation : Beaucoup d'informations sont utilisées ici : en plus des variables présentes dans les autres classes, il faut en créer d'autres. La date de début et de fin ainsi que le prix total doivent apparaître, tout comme le nombre total de nuits. Il y a également plusieurs choses à vérifier : si le client existe, si la chambre souhaitée est disponible sur la période souhaitée et la possibilité de modifier une date de réservation.

Remarques : 1) Il y a un problème avec la chambre 20 (uniquement la chambre 20, qui est une suite). Lorsque l'on souhaite réserver une suite, seule la chambre 21 nous est proposée. En revanche, lorsque la chambre 21 est prise et que l'on souhaite tout de même réserver une suite, la chambre 20 n'est toujours pas proposée et il n'y a donc aucune possibilité de réserver pour une suite.

2) Nous avons remarqué que même l'identifiant n'était pas suffisant pour identifier un client, car l'id du client contient la première lettre du prénom + le nom. Or si 2 personnes se nomment pareil, il faudrait rajouter un chiffre en plus par exemple pour mieux les identifier. Mais là encore, si plusieurs personnes se nomment pareil, comment allons-nous reconnaître la personne si les chiffres sont aléatoires ? Il aurait certainement fallu une autre variable qui prendrait un chiffre à saisir (à ajouter à la fin de l'id client) et qui indiquerait qu'il faut changer de chiffre, si l'id client et le chiffre ont déjà été attribués à un autre client. Ici, nous supposons que ce cas n'arrivera pas.

3) Le code n'est pas très propre et nous pourrions l'améliorer, notamment en sortant les std::cout des fonctions et les placer dans le main.

Conclusion générale

A l'issue de ces travaux pratiques, nous avons constaté l'importance de la décomposition en sous-problèmes : en effet, au vu du nombre important d'informations, réaliser petit à petit les parties de codes est essentiel. Cela permet d'éviter les erreurs et de vérifier le bon fonctionnement des différents codes étapes par étapes.
De plus, il est important de bien hiérarchiser l'ordre dans lequel il faut réaliser les codes : typiquement dans ce TP, il faut d'abord créer les autres classes avant de faire la classe Réservation.
Nous nous sommes aperçus après coup que de nouveaux problèmes peuvent apparaître, même lorsque l'on pense qu'il n'y en a plus. Même si cela ne reflète pas forcément la réalité, nous nous sommes permis de faire abstraction de certains cas particuliers. La prochaine étape sera pour nous de pousser encore plus loin notre réflexion dans la recherche de problèmes, mêmes ceux qui nous semblent négligeables.
Nous avons choisi ce TP car, n'ayant peu ou pas du tout pratiqué de langage C++ dans nos cursus respectifs, nous souhaitions d'abord nous faire une idée de la manière dont il faut établir les démarches de résolution dans la programmation orientée objet. Un TP de niveau Débutant nous semblait donc approprié pour découvrir cela, d'autant plus que les étapes ont permis de bien nous guider pour ce premier sujet. Nous avons pu apprendre à créer des classes fonctionnelles qui interagissent entre elles. Nous avons aussi appris à surcharger des opérateurs, à utiliser des vecteurs et des itérateurs.