TP N° 6
Exception

Objectifs

- *Réviser la notion d'exception, ainsi que la manière dont on l'appelle (throw-try-catch).*

0. Introduction

Ecrire la classe Erreur vue dans le cours et la tester

Ecrire un programme qui gère la saisie d'un entier avec des exceptions

Ecrire un programme qui gère la division par zéro avec des exceptions

1. Une simple division par zéro

Nous allons reprendre l'exercice sur les nombres rationnels. Maintenant que l'on sait comment les manipuler, il faut prévoir ce qui se passe au cas où on voudrait diviser un nombre rationnel par un autre nombre rationnel nul.

Modifier l'opérateur de division pour qu'il lève une exception en cas de division par zéro.
Essayer plusieurs types d'exceptions (type primitif, classe...).

Modifier le programme principal pour qu'il intercepte les exceptions générées par une mauvaise division.

4. Le coursier au sac dynamique

Nous allons reprendre l'exercice du coursier et modifier sa classe Coursier, afin de lui conférer un sac de volume toujours limité, mais de quantité cette fois illimitée. Pour ce faire, nous allons utiliser le conteneur *vector*. Le sac demeure donc un tableau de courriers, pour lequel on doit pouvoir être renseigné à tout moment sur la quantité et le volume de courriers qu'il contient. A la création, le sac est vide.

On doit pouvoir calculer l'affranchissement total du sac, afficher son contenu, retourner le courrier d'indice donné, et ajouter un courrier à la fin du sac, à condition qu'il y ait encore de la place. Pour l'avant-dernière méthode en particulier, on lèvera une exception au cas où l'entrée demandée est trop grande par rapport à la quantité courante du sac. Et pour la méthode d'ajout, on lèvera des exceptions au cas où un tel ajout dépasserait la capacité du sac (en quantité comme en volume).

Ecrire un programme principal qui permet de tester cette classe, et surtout d'intercepter les exceptions générées par une mauvaise demande de renseignement, ou un mauvais ajout.