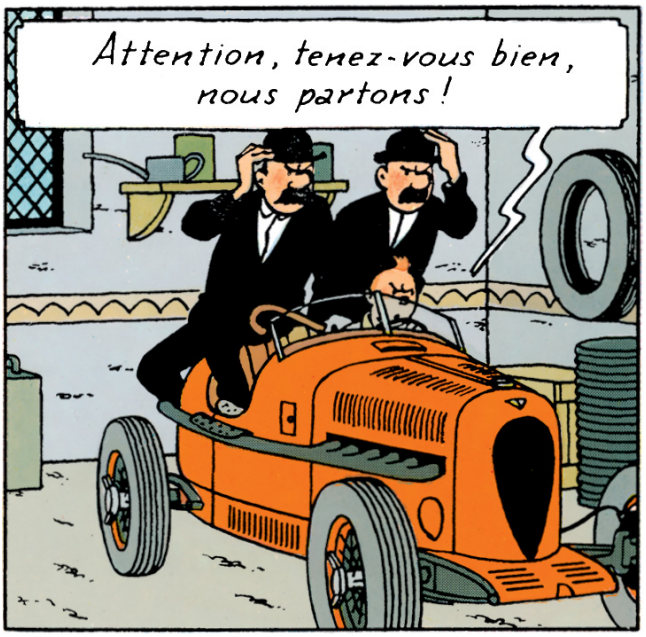
Structures de Données 2024 - TP1

B. Pulfer
Brian.Pulfer@unige.ch

A. Freeman
Arthur.Freeman@etu.unige.ch

E. Arm
Ethan.Arm@etu.unige.ch

Février 2024



Règles

Note: Les travaux ne sont pas notés. Cependant, nous vous encourageons vivement à les faire afin de vous familiariser avec les concepts traités en classe. **Date limite**: Nous contribuerons à votre projet personnel avec des cas de test supplémentaires dans deux semaines, d'où le **12.03.2024**. Essayez de terminer le travail avant cette date pour voir si votre implémentation est correcte même à ce moment-là.

1 Le garagiste

1.1 Problème

Un garagiste souhaiterait mieux gérer son parc de voitures et sa clientèle. Jusqu'à maintenant, il notait tout dans de gros registres mais bien souvent il ne s'y retrouve plus. Il voudrait avoir un programme informatique pour gérer tout cela.

- Ce programme doit permettre d'obtenir le nom et adresse d'un client particulier pour pouvoir le facturer.
- Il doit aussi permettre d'avoir les info des voitures présentes au garage: marque, modèle et immatriculation, ainsi que leur propriétaire.
- Le programme doit aussi permettre de lister tous les clients et voitures a partir d'un registre.

1.2 Modélisation

Veuillez répondre et expliquer votre réflexion sur les questions suivantes:

1. Identifier les noms pouvant être regroupés ensemble. Pourquoi ce choix ?
2. Construire les composants correspondant à ces concepts avec leurs fiches techniques.
3. Représenter les dépendances entre les composants avec UML. Vous pouvez utiliser `https://nomnoml.com/` pour cela.

1.3 Implementation

Modifier les fichiers **lib/objects/objects.h** et **lib/objects/objects.c** pour créer les structures de données nécessaires à la résolution de l'exercice. Certaines parties du code sont déjà données et ne doivent pas être modifiées. Vous pouvez voir ce que nous attendons du code dans le fichier de test **test/test_objects.c**. Veuillez suivre le flux de travail.

1.4 Flux de travail

- Modifier les fichiers **lib/objects/objects.h** et **lib/objects/objects.c**
- Exécute **make test_objects** à partir d'un terminal (Si vous êtes sous Windows, exécutez ceci à partir du terminal **mingw64**. Si cela ne fonctionne pas, installez make avec **pacman −S make**).
- Si tous les tests semblent réussis:
  - git pull
  - git add lib/objects/
  - git commit −m "Mon message de commit."
  - git push
  - Vérifiez que le test correspondant a réussi sur votre repo GitHub, dans la section "actions". Si elles prennent trop de temps (ou ne sont pas exécutées du tout), envisagez de passer à GitHub Pro gratuitement avec votre adresse électronique de l'université `https://education.github.com/pack`.
  - Si le test n'a pas reussi, revenez à la première étape. Si le test a réussi, écrivez un fichier principal dans src/tp1/main.c tout en utilisant les structures de données nouvellement créées!
  - add, commit, push le fichier src/tp1/main.c.

Après la date limite, nous allons ensuite "contribuer" à votre repo en y insérant un nouveau fichier de test avec des cas de test différents qui devraient toujours fonctionner. Nous discuterons également de la solution en classe.