

1 Introduction

Le but de ce TP est de vous initier :

- à la gestion de version via git
- aux bonnes pratiques de programmation en demandant de réécrire du code mal écrit.

Lien du TP : https://classroom.github.com/a/RspiyPx_

2 Gestion de versions

2.1 Quelques explications sur git

Ce projet est l'occasion d'apprendre à utiliser des outils de développement professionnel. En plus de l'IDE Pycharm, vous aurez à utiliser un outil de gestion de versions, en l'occurrence `git`. Un tel outil permet d'enregistrer à distance votre projet et de permettre à plusieurs personnes de travailler simultanément, sans devoir échanger les fichiers par courriel, et sans se marcher sur les pieds (par exemple modification sans le savoir du même fichier).

Le principe de `git` est d'avoir un *dépôt* distant : une version de votre projet stockée sur un serveur sur Internet (en l'occurrence par `github`). Vous disposez en plus de dépôts locaux sur les ordinateurs sur lesquels vous travaillez. Vous faites vos modifications sur votre ordinateur, et lorsque vous avez accompli une amélioration qui fonctionne bien, vous pouvez la faire enregistrer (`commit`) par `git`. Ces enregistrements sont locaux à votre ordinateur, et vous pouvez en faire autant que vous le souhaitez. Si vous voulez partager votre travail avec votre équipe, il vous faut l'envoyer vers le dépôt distant (`push`). À l'inverse, si vous souhaitez récupérer le travail fait par vos coéquipiers, il faut ramener ces modifications depuis le dépôt distant (`pull`). Pycharm est capable de gérer `git` ; vous trouverez dans le menu VCS l'option `Commit`, et l'option `Git` qui contient `push` et `pull`.

2.2 Méthodologie de travail sur git

Vous travaillerez de la façon suivante :

- Si vous ne possédez pas de compte Github, aller sur `github` et créez-vous un compte gratuit.
- Allez sur le lien du TP pour accepter le devoir. Vous devez pour cela être connecté à votre compte `github`.
- Au tout début, vous lancerez Pycharm, puis commencerez un nouveau projet depuis un dépôt `git` (menu `File, New, Project from version control, Git`). Renseignez l'adresse du dépôt `git` fournie par `github`, vous la trouverez en cliquant sur le bouton vert `clone or download` sur la page `github` du projet. Pycharm va initialiser votre dépôt local, en vous demandant dans quel répertoire le placer.
- Modifiez le fichier `README.md`. Mettez votre nom. Faites un `commit` avec pour message "inscription d'un membre de l'équipe", suivi d'un `push`.
- Après avoir fini une modification de code, faites un `commit` en sélectionnant les fichiers modifiés. Chaque `commit` doit contenir un message précisant la nature des modifications effectuées.
- À chaque tâche terminée, faites un `push` de votre travail.
- Ceci est le minimum. Vous pouvez faire plus de `commit` et plus de `push`, ainsi que des `pull` pour récupérer le travail de votre coéquipier.

Si vous ne souhaitez pas gérer `git` avec Pycharm, vous pouvez aussi le faire en ligne de commande. Vous trouverez un tutoriel au lien suivant : <https://git-scm.com/docs/gittutorial>

3 Tâche 1 : modifier le code existant du mastermind

3.1 Le jeu du mastermind

Normalement vous devriez avoir un fichier `mastermind.py` contenant le code d'un jeu de mastermind.

On rappelle le principe de ce jeu. Un joueur (ici l'ordinateur) choisit une séquence secrète de 5 chiffres parmi les 10 chiffres disponibles (qu'on représentera avec les entiers 0, 1, 2, 3, 4, 5, 6, 7, 8 et 9). On représentera une séquence par une liste. Un chiffre peut apparaître plusieurs fois dans une séquence. L'adversaire (ici le joueur humain) de ce joueur a 15 tentatives pour deviner cette séquence sachant qu'après chaque proposition, il bénéficie d'une indication : l'ordinateur examine sa proposition et lui dit combien de chiffres sont bien placés et combien sont présents dans la séquence secrète, mais mal placés. La saisie se fait chiffre par chiffre (en appuyant sur entrée entre chaque chiffre). Ainsi dans l'exemple suivant :

- séquence secrète 42605
- séquence proposée 16405

le joueur qui devine se verra répondre "2 chiffre(s) bien placé(s) et 2 chiffre(s) mal placé(s)", correspondant aux chiffres 0 et 5 pour les bien placés, et aux chiffres 6 et 4 pour les mal placés.

Voici un deuxième exemple pour bien fixer les idées :

- séquence secrète 11603
- séquence proposée 16411

le joueur qui devine se verra répondre "1 chiffre(s) bien placé(s) et 2 chiffre(s) mal placé(s)", correspondant au chiffre 1 en position 0 bien placé et les chiffres 1 (position 3 ou 4) et 6 (position 1).

3.2 Le code existant

Dans le fichier vous trouverez le code fonctionnel du mastermind tel que décrit au-dessus. Le code a été écrit par un élève n'ayant jamais entendu parlé des bonnes pratiques de programmation :

- nommage précis et rigoureux des éléments du code : les variables ont des noms très courts dont la plupart sont durs à déchiffrer ;
- utilisation de petites fonctions : il y a qu'une seule fonction faisant une centaine de lignes en incluant les commentaires.

Question : Quels autres problèmes au code voyez-vous en plus des deux que l'on vient d'indiquer (nommage imprécis et fonction trop longue) ?

Question : Pensez-vous qu'il est facile de changer le code pour modifier des fonctionnalités comme le nombre de chiffres autorisés dans les combinaisons ?

Question : Réécrivez le code en utilisant les outils de refactoring de Pycharm (clic droit puis `refactor` dans le menu) ou raccourcis indiqués ci-dessous :

- Shift + F6 **Rename** : permet de renommer partout dans le code un symbole (fonction, variable, constante).
- Ctrl + F6 (CMD + F6) **Change Signature** : permet de changer la signature (les paramètres et leur ordre) d'une fonction.
- Ctrl + Alt + M (CMD + OPT + M pour mac) **Extract Method** : extrait le code sélectionné dans une fonction que vous nommez.
- Ctrl + Alt + V (CMD + OPT + V pour mac) **Extract Variable** : extrait l'expression sélectionnée dans une variable que vous nommez.
- Ctrl + Alt + C (CMD + OPT + C pour mac) **Extract Constant** : extrait la valeur sélectionnée dans une constante que vous nommez.

Votre but est d'obtenir un code :

- **lisible** : qui peut se lire facilement et qui n'a pas besoin de commentaire pour être compris.
- **modifiable** : le code est facile à modifier car il est modulaire (en autre petites fonctions réutilisables).

N'oubliez pas de faire un commit après chaque modification du code avec un commentaire décrivant la modification et un push quand vous avez fini de refactorer le code.

3.3 Pour ceux voulant aller plus loin

Pycharm possède d'autres fonctionnalités que vous pouvez trouver au liens suivants :

- Documentation Pycharm
- Raccourcis windows et linux
- Raccourcis mac

4 Tâche 2 : rajouter du code pour de nouvelles fonctionnalités

Question : Rajouter des fonctionnalités au code existant du mastermind :

- paramétrage de la partie : permettre la modification des paramètres suivants :
 - taille des combinaisons : plus ou moins que 5 chiffres.
 - chiffres autorisés :
 - moins de chiffres (facile).
 - plus de chiffres et donc la possibilité d'avoir de lettres : A, B , ...
 - nombre de tentative autorisée (facile).
- Ajouter des affichages/saisie pour que cela soit l'utilisateur qui détermine au début les différents paramètres de la partie.
- Changement de la saisie des combinaisons :
 - Rajouter du code pour prendre en compte la possibilité que l'utilisateur ne rentre pas un chiffre et lui demander par exemple de redonner une nouvelle valeur.
 - Permettre à l'utilisateur de saisir la combinaison en une fois (en tapant par exemple 12467 puis entrée).
 - Permettre à l'utilisateur d'abandonner la partie en cours.
- interface graphique (pour ceux qui sont à l'aise en Python)

5 Projet

5.1 Démarrage de votre projet

Lien du projet : <https://classroom.github.com/g/rL2zgf5j>

- Allez sur le lien du projet pour accepter de créer un dépôt pour votre projet. Vous devez pour cela être connecté à votre compte github.
- La différence avec le TP est que c'est un travail en équipe. Vous pouvez créer une équipe (première personne du groupe) ou bien rejoindre une équipe existante (pour les personnes suivantes du groupe). Mettez-vous d'accord avant sur la composition des équipes et essayer de donner un nom cohérent à votre équipe.
- Démarrez votre projet en suivant les instructions dans le `README.md`.