h e g | Haute école de gestion de Genève

TP03

L'objectif de ce TP est de vous faire pratiquer l'utilisation des fonctions et affiner vos connaissances des boucles `for` et `while`. Attention à bien récupérer le projet fourni avec le TP !

1) Nombre porte-bonheur

L'objectif du programme est de tester si un nombre entier positif plus grand que 10 est un nombre porte-bonheur. Un nombre porte-bonheur est un nombre entier qui, lorsque l'on ajoute les carrés de chacun de ses chiffres, puis les carrés des chiffres de ce résultat et ainsi de suite jusqu'à l'obtention du résultat 1.
Le calcul s'arrête lorsque le chiffre devient <= 10.

Exemple de sortie :

```
Veuillez entrer un nombre plus grand que 10 : 913
9^2=81
1^2=1
3^2=9
Nouveau nombre : 81+1+9=91
9^2=81
1^2=1
Nouveau nombre : 81+1=82
8^2=64
2^2=4
Nouveau nombre : 64+4=68
6^2=36
8^2=64
Nouveau nombre : 36+64=100
1^2=1
0^2=0
0^2=0
Nouveau nombre : 1+0+0=1
Le nombre 913 est un nombre porte bonheur !
```

```
Veuillez entrer un nombre plus grand que 10 : 97
9^2=81
7^2=49
Nouveau nombre : 81+49=130
1^2=1
3^2=9
0^2=0
Nouveau nombre : 1+9+0=10
Le nombre 97 n'est pas un nombre porte bonheur
```

2) Jeu du pendu

Vous devez programmer un jeu du pendu. Pour rappel, le jeu du pendu est un jeu ou l'objectif est de découvrir un mot caché en choisissant une lettre. La longueur du mot caché est connue car il est affiché sous cette forme :

Pour le mot « Bonjour » : _ _ _ _ _ _ _

Si le mot caché contient cette lettre, l'indice est mis à jour avec cette lettre.

Choix de la lettre « o » : _ o _ _ o _ _

Sinon le joueur perd un essai. Si le joueur atteint un nombre prédéfini d'essais manqués, le joueur perds la partie. S'il trouve le mot avant, il gagne la partie. Le joueur possède **10 essais** et le mot caché est sélectionné aléatoirement dans une liste de mots. Le mot caché se trouve dans la constante `MOT` mise à votre disposition.

Pour ce faire vous devrez séparer votre code en plusieurs fonctions.

a. La fonction `affichage_mot_cache` qui prends en paramètres le mot caché, les lettres déjà tirées et le nombre d'erreurs restant
   i. Cette fonction affichera le mot caché et les lettres déjà trouvées sous la forme vu ci-dessus
   ii. Elle affichera également le nombre d'erreurs restantes (voir exemple de sortie)
b. La fonction `tirage_lettre` qui prend en paramètres les lettres déjà tirées. Celle-ci retournera les lettres déjà tirées avec la nouvelle lettre venant d'être choisie
   i. Cette fonction demandera à l'utilisateur de saisir une lettre et reposera la question tant que la lettre entrée a déjà été choisie
c. La fonction `verification_mot` qui prend en paramètres le mot et les lettres déjà tirées. Celle-ci retournera un booléen si le mot a été trouvé ou non
   i. Cette fonction vérifiera si le mot a effectivement été trouvé.
d. La fonction `diminuer_erreurs` qui prend en paramètre le nombre d'erreurs restants et retourne le nombre d'erreurs restant diminué de 1.

Exemple de sortie :

```
_ _ _ _              (10 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : e
_ _ _ _              (9 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : i
_ _ _ _              (8 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : u
_ _ _ _              (7 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : a
_ _ _ _              (6 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : o
_ o _ _              (6 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : r
_ o _ _              (5 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : t
_ o _ _              (4 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : s
_ o _ _              (3 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : m
_ o _ _              (2 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : n
_ o n _              (2 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : s
Veuillez entrer une lettre que vous n'avez pas choisie avant : o
Veuillez entrer une lettre que vous n'avez pas choisie avant : p
_ o n _              (1 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : q
Vous avez perdu ! Le mot était : long


_ _ _ _ _ _ _ _ _            (10 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : e
_ _ _ _ _ _ e _              (10 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : a
_ _ _ _ _ _ e _              (9 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : u
_ _ _ _ _ _ e _              (8 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : i
i _ _ _ _ _ e _              (8 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : o
i _ _ o _ _ e _              (8 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : r
i _ _ o r _ e r              (8 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : t
i _ _ o r t e r              (8 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : m
i m _ o r t e r              (8 erreurs restantes)
Veuillez entrer une lettre que vous n'avez pas choisie avant : p
Bravo vous avez trouvé le mot ! importer
```

2) BlackJack

Programme simulant un jeu de BlackJack avec des lancés de dés. L'objectif du jeu est d'arriver au plus proche de 21 sans dépasser 21. Pour se faire l'utilisateur peut lancer un nombre de dés de son choix. Le programme simule un lancer de dés (en générant aléatoirement des valeurs entre 1 et 6) et obtiens une somme. L'utilisateur peut décider de continuer de lancer des dés supplémentaires ou de s'arrêter (entrer 0 lorsque l'on demande le nombre de dés). Si l'utilisateur dépasse 21, sa partie se termine.
L'ordinateur joue également en parallèle avec sa propre somme et son score est affiché également. Le jeu se termine lorsque le joueur **ET** l'ordinateur ont terminé de jouer.

Pour faire cela, il faudra développer 4 fonctions :

a. La fonction `jeu` qui ne prends pas de paramètres
   i. Cette fonction sera le programme principal permettant le jeu du BlackJack. C'est ici que l'on appellera les autres fonctions et que l'on va gérer si la partie est en cours ou non.
b. La fonction `jeu_joueur` qui prend en paramètres la somme du joueur (son score) et l'objectif (le score à atteindre (21)). Celle-ci retournera le nouveau score du joueur après le lancer de dés et le nombre de dés lancés à ce tour par celui-ci.
   i. Cette fonction demandera à l'utilisateur de saisir un nombre de dés à lancer et affichera son score.
c. La fonction `jeu_ordinateur` qui prend en paramètres la somme de l'ordinateur (son score) et l'objectif (le score à atteindre (21)). Celle-ci retournera le nouveau score de l'ordinateur après le lancer de dés et le nombre de dés lancés à ce tour par celui-ci.
   i. Cette fonction affichera le nombre de dé choisi par l'ordinateur et son nouveau score.
d. La fonction `affichage_resultat` qui prend en paramètre le score du joueur, le score de l'ordinateur et l'objectif (21). Cette fonction va afficher un message selon les scores des participants.
   i. Si les deux dépassent 21
      ```
      Vous avez perdu car vous avez dépassé 21 (41)
      Rassurez-vous l'ordinateur n'a pas fait mieux (22).
      ```
   ii. Si le joueur dépasse 21 mais pas l'ordinateur ou que le joueur a un score plus petit que l'ordinateur
      ```
      Vous avez perdu (23) contre l'ordinateur (20)
      ```
   iii. Si le joueur ne dépasse pas 21 et gagne contre l'ordinateur (qu'il dépasse 21 ou non)
      ```
      Vous avez gagné (21) contre l'ordinateur (20)
      ```
   iv. En cas d'égalité
      ```
      Égalité ! Pas de gagnant ! (21)
      ```

Indications :

- Si le joueur entre 0 comme nombre de dés à lancer ou que le score du joueur dépasse 21, cela signifie qu'il arrête de lancer plus de dés et sa partie se termine
- Voici le détail sur la stratégie de jeu de l'ordinateur :
  - Si la somme de l'ordinateur est inférieure à 6, il demande 3 dés
  - Si la somme de l'ordinateur est supérieure ou égale à 6 et inférieure à 12, il demande 2 dés
  - Si la somme de l'ordinateur est supérieure ou égale à 12 et inférieure à 18, il demande 1 dés
  - Si la somme de l'ordinateur est supérieure ou égale à 18, il s'arrête de jouer

Exemple de sortie

```
Combien de dés souhaitez-vous lancer ? 4
Vous avez un score de  13

L'ordinateur choisi 3 dés
L'ordinateur a un score de  7

Combien de dés souhaitez-vous lancer ? 2
Vous avez un score de  24

L'ordinateur choisi 2 dés
L'ordinateur a un score de  15

L'ordinateur choisi 1 dés
L'ordinateur a un score de  18

Vous avez perdu (24) contre l'ordinateur (18)
>>>
```

```
Combien de dés souhaitez-vous lancer ? 4
Vous avez un score de  12

L'ordinateur choisi 3 dés
L'ordinateur a un score de  13

Combien de dés souhaitez-vous lancer ? 2
Vous avez un score de  19

L'ordinateur choisi 1 dés
L'ordinateur a un score de  14

Combien de dés souhaitez-vous lancer ? 1
Vous avez un score de  21

L'ordinateur choisi 1 dés
L'ordinateur a un score de  19

Combien de dés souhaitez-vous lancer ? 0
Vous avez un score de  21

Vous avez gagné (21) contre l'ordinateur (19)
>>>
```