Introduction à la programmation – TP6

Exercice 1 – Maximum de 4 nombres

Écrire une fonction `maximum` qui prend en paramètres deux entiers `a` et `b` et qui renvoie le maximum entre `a` et `b`.

En utilisant **obligatoirement** la fonction précédente, écrire une fonction `maximum4` qui prend en paramètres quatre entiers `a`, `b`, `c` et `d`, et qui renvoie le maximum des quatre nombres.

Dans le programme principal, vous testerez vos fonctions en affichant le résultat de `maximum(12,1)` et de `maximum4(15,3,17,15)`.

Exercice 2 – retour sur des mots

Une institutrice distribue des mots à ses élèves pour qu'ils apprennent à les écrire. . Chaque mot sera une chaîne de caractères (on ne prendra ni articles ni mots composés). Ces mots vont être mis dans des tuples pour avoir un ensemble de mots. Par exemple t=('pomme', 'poire', 'ananas', 'banane', 'citron', 'carambole', 'kiwi', 'pastèque')

1. Comparaisons : Deux enfants veulent comparer leurs listes de mots respectives.
   a. Ecrire la fonction compare_nb(l1,l2) qui renvoie le nombre de mots communs aux deux listes.

```
>>> compare_nb(("ananas", "pomme", "poire", "kiwi", "banane"), ("kiwi", "pomme", "poire", "fraise","
3
```

   b. Ecrire la fonction compare(l1,l2) qui renvoie le tuple des mots communs aux deux listes.

```
>>> compare(("ananas", "pomme", "poire", "kiwi", "banane"), ("kiwi", "pomme", "poire", "fraise","abr
('pomme', 'poire', 'kiwi')
```

2. Ecrire la fonction fusion(l1,l2) qui étant données deux tuples de mots renvoie le tuple des mots obtenus en prenant les mots étant dans l'une ou l'autre des listes mais sans répétitions.

```
>>> fusion(("ananas", "pomme", "poire", "kiwi", "banane"), ("kiwi", "pomme", "poire", "fraise", "abr
('ananas', 'pomme', 'poire', 'kiwi', 'banane', 'fraise', 'abricot')
```

3. mots longs
   Ecrire la fonction motpluslong(listemots) qui étant donnée un tuple de mots listemots, renvoie un des mots qui a le plus de lettres dans le tuple (si les mots les plus longs de la liste comportent 8 lettres, on renverra n'importe quel mot de 8 lettres)

```
>>>motpluslong(("ananas", "pomme", "poire", "carambole", "banane"))
'carambole'
>>>motpluslong(("ananas", "pomme", "poire", "kiwi", "banane"))
'ananas'
```

Exercice 3 – les dominos

On va travailler avec un jeu de dominos : on va représenter un domino par un tuple de deux entiers compris entre 0 et 6 (inclus) par exemple (2,4). Une main (jeu d'un joueur) sera représentée par un tuple de dominos de même que le plateau de jeu (c'est à dire les dominos déjà posés).

1. A la fin du jeu, pour compter les points, on fait la somme de tous les chiffres inscrits sur les dominos.
   — Ecrire la fonction somme(d) qui fait la somme des deux chiffres du domino d
   — Ecrire la fonction comptePoints(jeu) qui étant donné un jeu va renvoyer le nombre de points correspondant (on utilisera somme) »>somme(((1, 4)) 5 »>ComptePoints(((1, 4), (4, 5), (0, 5), (0, 0), (0, 1))) 20
2. Il peut être stratégique de se débarrasser du domino qui coûterait le plus cher si on le gardait à la fin du jeu. Ecrire donc la fonction pluscher(jeu) qui renvoie le domino du jeu qui coûte le plus de points. Si plusieurs dominos réalisent le même nombre de points, on n'en renverra qu'un seul (n'importe lequel).

```
>>> main=((1, 4),(0, 5),(5, 5), (0, 0), (4, 6), (2, 6))
>>> pluscher(main)
(5,5)
```

3. On va maintenant s'occuper du choix d'un domino à jouer pour l'ordinateur.
   — Ecrire une fonction qui étant donné un domino et un plateau, teste si le domino est jouable. Attention à ne pas oublier le cas où le plateau est vide.

```
>>> plateau= ((3, 6), (6, 5), (5, 5))
>>>jouable((1,5), plateau)
True
>>>jouable((1,6), plateau)
False
```

4. En déduire la fonction dominosJouables(jeu, plateau) qui renvoie le tuple de tous les dominos jouables du tuple jeu par rapport au plateau. Si aucun domino n'est jouable on renverra le tuple vide.

```
>>> plateau=((3, 6), (6, 5), (5, 5))
>>> mainOrdi
((1, 4), (4, 5), (0, 5), (0, 0), (0, 1), (2, 6), (3, 4), (5, 5))
>>> dominosJouables(mainOrdi, plateau)
((4, 5), (0, 5), (3, 4), (5, 5))
```

5. On va faire jouer à l'ordinateur l'un des dominos qui compte le plus de points parmi les dominos jouables. Ecrire la fonction choixOrdi(main, plateau) qui renvoie False si l'ordinateur ne peut pas jouer et sinon le domino à jouer par l'ordinateur.

```
>>> plateau=((3, 6), (6, 5), (5, 5))
>>> mainOrdi
((1, 4), (4, 5), (0, 5), (0, 0), (0, 1), (2, 6), (3, 4), (5, 5))
>>> choixOrdi(main, plateau)
(5, 5)

>>> mainOrdi
((1, 4), (4, 4), (0, 0), (0, 1), (2, 6)))
>>> dominosJouables(mainOrdi, plateau)
        ()
>>> choixOrdi(main, plateau)
False
```

6. On considère un plateau résultant d'un début de jeu effectué sans tests, et on voudrait savoir si les dominos ont bien été posés. Ecrire une fonction qui teste si un plateau est "correct" c'est à dire si les dominos s'enchaînent bien. On considèrera là que le plateau n'est pas vide.

```
>>> plateau
((3, 6), (6, 5), (5, 5), (5, 2), (2, 2), (2, 4), (4, 6), (6, 6), (6, 1))
>>> correct(plateau)
True
>>> plateau2
((3, 6), (6, 5), (5, 5), (5, 2), (2, 4), (4, 6), (6, 6), (6, 1), (4,1)))
>>> correct(plateau2)
False
```

7. Il existe 28 dominos différents. On ne va pas les écrire à la main!!! Ecrire la fonction creationJeu, sans paramètre, qui renvoie le tuple des 28 dominos.

```
>>> creationJeu()
((0, 0), (0, 1), (0, 2), (0, 3), (0, 4), (0, 5), (0, 6), (1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1,
```

8. Et pour les plus rapide,commencer à écrire le programme pour jouer!