R308 : Consolidation de la programmation

TPD3 : Les exceptions

1 Contrôle de saisie par exception :

Écrivez un programme qui demande à l'utilisateur de saisir un nombre compris entre 0 et 20

Si l'utilisateur ne saisit pas un nombre ou si la saisie n'est pas convenable, le programme doit indiquer à l'utilisateur qu'il a fait une erreur et lui redemander de saisir un nouveau nombre.

Lorsque la saisie est convenable, le programme affiche le nombre de tentatives.

Consignes : toutes des contraintes devront être gérées dans votre propre exception:

- Une variable statique mémorisera le nombre de tentatives, une méthode de classe permettra de lire cette variable.
- Chaque tentative sera un paramètre du constructeur.
- Chaque appel à votre exception retournera un message explicite qui permettra de mettre fin à la boucle de saisie.
- Une erreur de type "ValueError" devra faire un appel à votre exception.

2 Voici le début d'un jeu très simple consistant à arrêter une boucle sur un nombre précisé à l'avance :

```
#!/usr/bin/python
#coding:utf-8

import random
import time
import os

if __name__ == "__main__":
      MIN:int = 10
      MAX:int = 50
      tempo:float = 0.2
      nb_secret:float = random.randint(MIN,MAX)
      i:int
      fin:bool = False
      print ("Vous devez arreter le programme sur ",nb_secret)
      print ("Pour arreter le programme , il faut faire Ctrl+C ")
      input("Appuyer sur Entree pour commencer ...")
      i=0
      while(i<MAX):
            i+=1
            os.system("clear")
            print("Vous devez arreter le programme sur ",nb_secret)
            print(i);
            time.sleep(tempo)
```

En gérant l'exception KeyboardInterrupt, compléter le programme pour qu'il réponde aux spécifications.

3 Calcul de points d'une fonction :

Écrivez un programme permettant de calculer et d'afficher la courbe |sin(x)/x| ou x [-20,0 ; 20,0] avec un pas de 0,5.

Vous afficherez la courbe à l'écran en utilisant les fonction plot() et show() de la classe matplotlib.pyplot

Le programme devra gérer les exceptions.

Remarque : vous utiliserez la classe math pour les calcul (et non pas numpy)

4 Nombre magique

Écrire un jeu dans lequel votre programme va choisi aléatoirement un nombre entre 0 et 100, et essayer de trouver ce nombre en 10 étapes maximum.

Votre programme devra donner des précisions à chaque nouvelle tentative :

"trop grand", "trop petit", "trop tard" ou "bravo, vous avez trouvé en xx tentatives"

Pour cela, vous utiliserez le mécanisme des exceptions. Voici le diagramme UML de la classe :

Classe Devine hérite de Exception	
# Attributs __proposition : int **# Variable de classe** NB_TENTATVES: int = 0 NB_MAGIQUE: int = None FIN: bool = False	
# Méthodes __init__(self, proposition: int): __str__(self)-> str: @staticmethod set_nb_magique(): get_fin()-> bool:	 La proposition sera un paramètre du constructeur la méthode __str__() retournera le message en fonction de la proposition la méthode statique permettra de fixer le nombre à deviner. La méthode statique permettra de tester la fin du jeu

4.1 Le programme principal est donné :

```
if __name__ == "__main__":
	valeur: int = None
	Devine.set_nb_magique()
	while not Devine.get_fin():
		try:
			valeur = int(input("entre un nombre : "))
			raise Devine(valeur) # une exception est levée à chaque nouvelle proposition
		except Devine as mex:
			print(mex)
		except ValueError:
			print("erreur de saisie ... un nombre est attendu")
```

5 Ouverture de fichier :

Lorsque l'on essaie d'ouvrir un fichier qui n'existe pas avec la fonction open de Python, l'interpréteur nous renvoie une exception (mal définie en Python < 3 et IOError en Python3)

Écrire un programme qui demande à l'utilisateur le nom d'un fichier et essaie de le lire. Si le fichier n'existe pas, le programme affiche qu'il n'existe pas et redemande à l'utilisateur
un autre nom de fichier.

6 Fonctions récursives:

Voici une fonction récursive:

```
def fibonacci(n):
	int res
	if n==0 or n==1 :
		res = 1
	else :
		res = fibonacci(n-1)+ fibonacci(n-2)
	return res
```

Modifier le programme pour qu'il renvoie le nombre d'appel récursifs nécessaires au calcul de fibonacci(n).
Conseil : on pourra ajouter un argument à fibonacci.
Modifier le programme pour qu'il lève une exception lorsque le niveau de récursivité dépasse 100.

7 Addition en temps limité:

Écrire un programme qui demande à l'utilisateur de calculer la somme de deux nombres (pris aléatoirement entre 0 et 100).
En utilisant le module time, lever une exception lorsque l'utilisateur ne répond pas après 10 secondes.
Conseil : on utilisera une boucle while.