Université de Carthage Institut Préparatoire aux Etudes d'Ingénieurs de Nabeul Département Mathématique-Informatique	IPEIN المعهد التحضيري للدراسات الهندسية بنابل Institut Préparatoire aux Etudes d'Ingénieurs de Nabeul	Année universitaire : 2024/2025 Filière : SM, SP et ST Niveau d'étude : 2ème année Semestre : 1 Nombre de pages : 3 Date : 24/10/2024 Durée : 1H30

Devoir Surveillé n°1 d'informatique
Corrigé

Exercice1 : (10 points)

```
#1- Fonction saisie
def Saisie():
    while True:
        try:
            n = int(input("Donner n >= 0:"))
            if n >= 0: return n
        except: print("Erreur de saisie...")

#2- Ecrire « factIter(n) » une fonction itérative
def factIter(n):
    f = 1
    for i in range(1,n+1): f *= i
    return f
#3- fonction récursive « factRec(n) »
def factRec(n):
    assert n >= 0,f"Erreur1: {n=} < 0 !"
    if not n: return 1
    else: return n * factRec(n-1)
#4- schéma d'exécution de la fonction récursive
"""
<-120-
      |
    FR(5) <-24--
       |        |
        ----> FR(4) <-6--
                |        |
                 ----> FR(3) <-2--
                         |        |
                          ----> FR(2) <-1--
                                  |        |
                                   ----> FR(1) <-1--
                                           |        |
                                            ----> FR(0)
* La complexité spatiale est linéaire O(n):
Coût = n appels = n * (1 test >= + 1 multiplication)
     = 2n d'où O(n)
* La complexité temporelle est linéaire O(n):
coût = n appels soit n sauvegarde de paramètre n dans
la pile d'où un coût = n case mémoires soit O(n)
"""

```

```
#5- Fonction récursive rapide « puiRecRap(x,n) »
def puiRapRec(x,n):
    if n == 0: return 1
    elif n == 1: return x
    r = puiRapRec(x,n//2)
    if n%2: return x*r*r
    else: return r*r

#6- Fonction récursive « expRec(x,n) »
def expRec(x,n):
    assert n >= 0,f"Erreur3: {n=} < 0 !"
    if n == 0: return 1
    else: return puiRapRec(x,n)/factIter(n) + expRec(x,n-1)

#7- Fonction itérative « expEps(x,eps=0.001) »
def expEps(x,eps=0.001):
    n,s = 0,0
    while 1:
        #print(s,n)
        delta = puiRapRec(x,n)/factIter(n)
        if abs(delta) <= eps: return s,n
        s += delta
        n += 1

#8- Programme principal « main() »
#if __name__ == '__main__':
#ou def main():
#a) Ouverture de fichier en lecture
with open("in.txt") as f:
    #Chargement de fichier dans « Lx »
    Lx = []
    for ligne in f:
        Lx += ligne.strip().split(';')
    #print(Lx)
    Lx = [float(e) for e in Lx]
#b) Calculer les exponentielles des toutes les réels
    print("Donner n :")
    n = Saisie()
    d = dict()
    for x in Lx:
        d[x] = expRec(x,n)
    #print(d)
#c) Sauver le contenu de « d » dans un fichier
    with open("out.txt","w") as f:
        f.write("   x  \t  exp(x) \n")
        for k,v in d.items():
            f.write(f"{k:7.3f} -> {v:7.3f}\n")
```

Exercice2 : (10 points)

```
1.  #1. Définition de la classe Point
2.  class Point:
3.      def __init__(self,x,y):
4.          self.x = x
5.          self.y = y
6.  #2. Méthode __str__
7.      def __str__(self):
8.          return f"P({self.x},{self.y})"
9.  #3. Méthode distance
10.     def distance(self):
11.         return (self.x**2 + self.y**2)**0.5
12. #4. Méthode « symétrie »
13.     def symetrie(self):
14.         return Point(-self.x,-self.y)
15. #5. Classe Rectangle
16. class Rectangle:
17.     def __init__(self,centre,largeur,hauteur):
18.         assert type(centre) == Point,'Erreur ...'
19.         self.C = centre
20.         self.L = largeur
21.         self.H = hauteur
22. #6. Méthode perimSurf
23.     def perimSurf(self):
24.         return {'Perimetre': 2*(self.L + self.H),
25.                 'Surface': self.L * self.H}
26. #7. Méthode coins
27.     def coins(self):
28.         c1 = Point(self.C.x-self.L/2,self.C.y+self.H/2)
29.         c2 = Point(self.C.x+self.L/2,self.C.y+self.H/2)
30.         c3 = Point(self.C.x+self.L/2,self.C.y-self.H/2)
31.         c4 = Point(self.C.x-self.L/2,self.C.y-self.H/2)
32.         return [c1,c2,c3,c4]
33. #8. Méthode affiche
34.     def affiche(self):
35.         d = self.perimSurf()
36.         cs = self.coins()
37.         ch = f"""Rectangle :
38. \tcentre = {str(self.C)}
39. \tLargeur = {self.L}
40. \tHauteur = {self.H}
41. \tPérimetre = {d['Perimetre']:.3f}
42. \tSurface = {d['Surface']:.3f}
43. \tCoins = {str(cs[0])},{str(cs[1])},{str(cs[2])},{str(cs[3])}
44. """
45.         print(ch)
46.
```

```
47.  #9. programme principal
48.  #if __name__ == '__main__':
49.  #ou def main():
50.  #ou
51.  #a- Saisie de deux réels « x et y »
52.  while True:
53.      try:
54.          x = float(input("Donner x:"))
55.          break
56.      except: print("Erreur de saisie...")
57.  while True:
58.      try:
59.          y = float(input("Donner y:"))
60.          break
61.      except: continue
62.  #b- Saisie de la largeur et la hauteur
63.  while True:
64.      try:
65.          L = float(input("Donner L:"))
66.          if L >0: break
67.      except: continue
68.  while True:
69.      try:
70.          H = float(input("Donner H:"))
71.          if H > 0: break
72.      except: continue
73.  #c- Création de l'objet « c » de type Point
74.  c = Point(x,y)
75.  #d- Création d'un objet « rect »
76.  rect = Rectangle(c,L,H)
77.  #e- Affichage des caractéristiques
78.  rect.affiche()
```