IPEIN
المعهد التحضيري للدراسات الهندسية بنابل
Institut Préparatoire aux Etudes d'Ingénieurs de Nabeul

EPREUVE D'INFORMATIQUE
EXAMEN DU 1ER SEMESTRE
NIVEAU : 2EME ANNEE

Correction

Filières : SM, SP, ST — Durée : 2h

Exercice 1 : (3pts) Réponse

1. Parmi les avantages de la programmation orientée objet par rapport à la programmation procédurale : **[1]**
   - Réutilisation du code ; écrire une fois, utiliser plusieurs fois.
   - Permet la création des nouveaux types.
   - Permet d'étendre les fonctionnalités d'un programme grâce à l'héritage entre les classes
   - Code plus lisible, plus claire, plus modulaire.
   - Protection des données grâce aux modificateurs : publiques, privés et protégés
   - …

2. **[1]**
   - **Problème 1 :** Enregistrement des pages web visités par un navigateur web dans une pile. Lorsque l'internaute visite des pages web lors de son navigation, elles seront enregistrées (empilées) dans l'ordre de visite chronologique. Si l'internaute veut réafficher une page déjà visitée, il doit cliquer plusieurs fois sur le bouton précédent (dépiler) jusqu'à ce qu'il arrive à la page voulue.
   - **Problème 2 :** Enregistrement des résultats des appels récursifs dans une pile d'exécution lors de calcul du factoriel d'un entier n quelconque.

1 * Factoriel (0)
…
n-1 * Factoriel (n-2)
n * Factoriel (n-1)
Factoriel (n)

- **Problème 3 :** Utiliser une file d'attente pour enregistrer temporairement des documents en attente d'impression.

Fichier1.doc	Exercice1.txt	…	Test.pdf
En cours d'impression	fichier en attente d'impression		Dernier document ajouté
File d'attente			

- **Autres Problèmes** …

3. Pratiquement, la contrainte majeure de ne pas avoir une pile à capacité infinie est **la taille limitée de la mémoire** de la machine sur laquelle on implémente notre pile. **[1]**

Exercice 2 : (3pts)

Question	Réponse	Justification	
1	e) AttributeError	L'instance enfant n'a aucun attribut 'v1'. self.v1 n'a jamais été créé en tant que variable puisque le parent. __init__ n'était pas explicitement appelé.	**[1]**
2	c) 123	L'instanciation de classe appelle automatiquement la méthode __init__ et transmet 123 en tant que paramètre. 123 est affecté à l'attribut de données de l'objet appelé id. La valeur 100 n'est pas conservée dans l'objet car elle n'est pas affectée à un attribut de données de la classe / de l'objet.	**[1]**
3	a) La classe B hérite de A, mais le champ de données "i" dans A n'est pas hérité.		**[1]**

Exercice 3 : (6pts)

Question	Réponse	

1)	`class MaPile () :`
	`  def __init__(self, c = 0 ) :`
	`  self.pile = [None] * (c+1)`
	`  self.pile[0] = 1`
		[1]

2)	`  def est_vide (self) :` `  return self.pile[0] == 1`	**[0.5]**
3)	`  def est_pleine (self) :` `  return self.pile[0] == len(self.pile)`	**[0.5]**

4) **[0.5]**

```
def sommet (self) :
    if not self.est_vide():
      i = self.pile[0]
      return self.pile[i]
```

5) **[1.5]**

```
def empiler(self, e) :
    if not self.est_pleine():
      i = self.pile[0]
      self.pile[i] = e
      self.pile[0] += 1
    else:
      print("Débordement de pile, arrêt du programme")
      raise(IndexError)
```

6) **[1.5]**

```
def depiler(self) :
    if self.est_vide():
      print("Débordement négatif de pile : arrêt du programme")
      raise(IndexError)
    else:
      self.pile[0] -= 1
      i = self.pile[0]
      return self.pile[i]
```

7) **[0.5]**

```
def __str__(self):
    #cette method doit retourner une chaine de caractères
    ch = str(self.pile)
    return ch
```

Exercice 4 : (8 pts)

Question **Réponse**

1) **[1]**

```
class Vect2d :
   def __init__(self, x=0, y=0):
      self.x = x
      self.y = y
```

2) **[1]**

```
def add(self,autre):
  v = Vect2d()
  v.x = self.x + autre.x
  v.y = self.y + autre.y
  return v
```

3) **[1]**

```
def mul_ext(self, k):
  v = Vect2d()
  v.x = self.x * k
  v.y = self.y * k
  return v
```

4) **[1]**

```
def zoom (self, k):
  self.x *= k
  self.y *= k
```

5) **[1]**

```
def prodscal (self,autre):
  x = self.x * autre.x
  y = self.y * autre.y
  return x+y
```

6) **[0.5]**

```
def norme (self):
  from math import sqrt
  return sqrt(self.x **2 + self.y **2)
```

7) **[1]**

```
#En déhors de la classe
def add(v1,v2):
   v = Vect2d()
   v.x = v1.x + v2.x
   v.y = v1.y + v2.y
   return v
```

8) **[1.5]**

[0.25]

```
#Programme principal
#a)
u = Vect2d(3,-2)
v = Vect2d(4,1)
```

[0.25]

```
#b)
w = u.add(v)
print(w.x, w.y)
```

[0.25]

```
#c)
ps1 = w.add(u).prodscal(v)
ps2 = u.prodscal(v) + w.prodscal(v)
print(ps1 == ps2 )
```

[0.25]

```
#d)
from copy import copy
u1 = copy(u)
u1.zoom(5) #5u
ps1 = u1.prodscal(v)

ps2 = u.prodscal(v)
ps2 *=5

print(ps1 == ps2 )
```

[0.5]

```
#e)
a = u.prodscal(v) /( u.norme() * v.norme() )
from math import acos
alpha = acos(a)
print(alpha)
```

••• Fin de correction •••