Лабораторна робота №3

Класи та об’єкти

Приклад класу та методів

```
class Point:
     def __init__(self, x, y): # конструктор класу
          self.x = x
          self.y = y
     def setX(self, x):
          self.x = x
     def __str__(self):  # перетворення об'єкту на рядок
          return '(' + str(self.x) + ', ' + str(self.y) + ')'
     def __eq__(self, other): # перевантажений "=="
          if self.x == other.x and self.y == other.y: return
True
          return False
     def __add__(self, other):     # перевантажений "+"
          return Point(self.x + other.x, self.y + other.y)
     def __hash__(self):
          return 17*self.x + self.y

a = Point(1,2)
b = Point(3,1)
c = a + b

print a is b
print a == b
print c
```

Приклад класу, який реалізує структуру даних «множина» для цілих чисел - колекція елементів, яка містить унікальній значення.

```
class intSet(object):
    #An intSet is a set of integers
    def __init__(self):
        """Create an empty set of integers"""
        self.numBuckets = 47
        self.vals = []
        for i in range(self.numBuckets):
            self.vals.append([])

    def hashE(self, e):
        #Private function, should not be used outside of class
        return abs(e)%len(self.vals)

    def insert(self, e):
        """Assumes e is an integer and inserts e into self"""
        for i in self.vals[self.hashE(e)]:
            if i == e: return
        self.vals[self.hashE(e)].append(e)
```

```
def member(self, e):
    """Assumes e is an integer
       Returns True if e is in self, and False otherwise"""
    return e in self.vals[self.hashE(e)]

def __str__(self):
    """Returns a string representation of self"""
    elems = []
    for bucket in self.vals:
        for e in bucket: elems.append(e)
    elems.sort()
    result = ''
    for e in elems: result = result + str(e) + ','
    return '{' + result[:-1] + '}'
```

Завдання по варіантам

1) Створити class Комплексне число (Complex), який би мав:
   - конструктор, що приймає дійсну та уявну частину (за замовченням вони рівні 0)
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - перевантажений оператор «+» для додавання до поточного комплексного числа іншого комплексного числа
   - перевантажений оператор «-» для віднімання від поточного комплексного числа іншого комплексного числа
   - перевантажений оператор «*» для множення поточного комплексного числа на інше комплексне число
   - перевантажений оператор «/» для ділення поточного комплексного числа на інше комплексне число
   - метод для знаходження модуля комплексного числа

2) Створити class Комплексне число (Complex), який би мав:
   - конструктор, що задає значення числа у тригонометричній формі: через модуль $r = |z|$ і аргумент φ($x = r\cos\varphi$, $y = r\sin\varphi$)
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - перевантажений оператор «+» для додавання до поточного комплексного числа іншого комплексного числа
   - перевантажений оператор «-» для віднімання від поточного комплексного числа іншого комплексного числа
   - перевантажений оператор «*» для множення поточного комплексного числа на інше комплексне число
   - перевантажений оператор «/» для ділення поточного комплексного числа на інше комплексне число
   - метод для знаходження модуля комплексного числа

3) Реалізувати class Точка (Point) (задається двома координатами), і на її основі class Лінія (Line). Клас лінія з наступними методам:
   - конструктор, що задає координати лінії двома точками типу класу Point
   - метод *__str__* для перетворення на рядок для використанні у функції *print*

- метод для знаходження довжини лінії
- метод, який би повертав точку (Point), яка знаходиться на середині лінії
- метод, який повертає довжину проекції на вісь X
- метод, який повертає довжину проекції на вісь Y

4) Реалізувати class Точка (Point) (задається двома координатами), і на її основі class Трикутник (Triangle). Клас трикутник з наступними методам:
   - конструктор, що задає координати трикутника трьома точками типу класу Point
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - метод для знаходження площі трикутника
   - метод, який би повертав кортеж, де містяться довжини сторін трикутника

5) Реалізувати class Точка (Point) (задається двома координатами), і на її основі class Вектор (Vector). Клас вектор з наступними методам:
   - конструктор, що задає координати початку та кінця вектора двома точками типу класу Point
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - метод для знаходження довжини вектора
   - метод, який би додавав до даного вектора другий вектор та повертав би новий вектор
   - метод, який би віднімав від даного вектора другий вектор та повертав би новий вектор

6) Реалізувати class Точка (Point) (задається двома координатами), і на її основі class Вектор (Vector). Клас вектор з наступними методам:
   - конструктор, що задає координати початку та кінця вектора двома точками типу класу Point
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - метод для знаходження довжини вектора
   - метод, який би множив вектор на число
   - метод, який би виконував скалярне множення даного вектора на другий вектор

7) Створити class Коло (Circule), який би мав:
   - конструктор, що задає його радіус та координати центру
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - метод для обчислення площі
   - метод для обчислення довжини кола
   - метод, який би повертав діаметр
   - метод, який би обчислював площу сектора, який задається кутом
   - метод, який би обчислював довжину дуги, яка задається кутом
   - метод який би визначав, чи перетинається дане коло з іншим колом

8) Створити class Студент (Student), який би мав:
   - конструктор, що задає прізвище та ім’я, а також словник, де у якості ключа буде значитись семестр, значеннями буде список оцінок за цей семестр
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - метод для середнього балу за певний семестр
   - метод для обчислення середнього балу за всі роки навчання

- метод, який би задавав список оцінок за певний семестр
- метод, який би повертав список оцінок за певний семестр

9) Створити class Студент (Student), який би мав:
   - конструктор, що задає прізвище та ім'я, а також словник, де у якості ключа буде значитись семестр, значеннями буде список оцінок за цей семестр
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - метод для середнього балу за певний семестр
   - метод для обчислення середнього балу за всі роки навчання
   - метод, який би визначав за середнім балом, у якому з семестрів студент вчився краще за все
   - метод, який би визначав за середнім балом, у якому з семестрів студент вчився гірше за все

10) Створити class Викладач (Instructor), який би мав:
   - конструктор, що задає прізвище та ім'я, а також словник, де у якості ключа буде значитись семестр, значеннями буде список предметів на цей семестр
   - метод *__str__* для перетворення на рядок для використанні у функції *print*
   - метод для обчислення кількості предметів на певний семестр
   - метод для обчислення середньої кількості предметів за всі семестри
   - метод, який би визначав у якому з семестрів було більше за все предметів у викладача
   - методи, які б дозволяли додавати/виділяти предмети у семестрах

Література:

Программирование на Python: Часть 6. Классы
https://www.ibm.com/developerworks/ru/library/l-python_part_6/