BÀI THỰC HÀNH 4
HƯỚNG ĐỐI TƯỢNG TRONG PYTHON

A. MỤC TIÊU:

- Hiểu được các nguyên lý lập trình hướng đối tượng trong Python: lớp và đối tượng, dữ liệu và hành vi thuộc lớp, tính thừa kế, lớp trừu tượng, tính đa hình.
- Vận dụng các nguyên lý lập trình hướng đối tượng trong các bài toán cụ thể.

B. NỘI DUNG THỰC HÀNH

1. Cơ sở lý thuyết

1.1. Kiến thức cần nhớ

1. Đối tượng

Tất cả mọi thứ trong Python đều là object (kể cả type).

Mỗi object là một instance của type. Ví dụ 123 là instance của int; "hello" là instance của string.

Mỗi objects – là trừu tượng/abstraction dữ liệu bao gồm:

- Data attribute (thuộc tính) để biểu diễn dữ liệu (primitive)
- Phương thức/hàm tương tác với object đó

```
print(type(123))  # <class 'int'>
print(type(int))  # <class 'type'>
```

1.1. Định nghĩa object

```
class ClassName(object):
    #Các thuộc tính và phương thức của lớp
```

Trong đó:

- Tiền tố __ (gạch dưới kép) để đánh dấu thành viên private.
- Tiền tố _ (gạch dưới đơn) để đánh dấu thành viên protected.

Ví dụ: định nghĩa lớp Student với phương thức khởi tạo và sayhello:

```
class Student:
    def __init__(self, name, dob):
        self.name = name
        self.dob = dob
    def sayhello(self):
        print("Hello, my name is " + self.name)
```

Trong đó: Phương thức *__init__()* là phương thức khởi tạo của Student.

Khai báo 1 thể hiện kiểu Student (khai báo đối tượng và khởi tạo đối tượng)

```
sv1 = Student("Thanh Long", "19-05-2002")
print(sv1.name)
```

```
sv1.sayhello()
```

Lưu ý: Tham số đầu tiên *self* là để trỏ đến chính đối tượng (instance) gọi hàm đó, và có thể dùng bất cứ tên gì thay thế. Không cần truyền giá trị cho *self* khi gọi phương thức, Python sẽ tự động điền giá trị này.

1.2. Biến class và biến instance (Class Side and Static Behaviour)

```
class HUFIStudent:
    uname = "HUFI"  #Biến tĩnh (Class Variable)
    def __init__(self, name, dob):
        self.name = name   #Instance Variable
        self.dob = dob     #Instance Variable
    def sayhello(self):
        print("Hello, my name is " + self.name)
#Objects of CSStudent class
sv007 = HUFIStudent("Long", "19-05-2002")
sv009 = HUFIStudent("Linh", "05-05-2002")
print(sv007.uname) #"HUFI"
print(sv009.uname) #"HUFI"
print(HUFIStudent.uname)  #"HUFI" : truy cập trực tiếp qua class name

sv007.uname = "HUS"
print(sv007.uname) #"HUS": thực tế 1 biến của object "uname"
print(sv009.uname) #"HUFI": biến của class vẫn không đổi
print(HUFIStudent.uname)  #"HUFI": truy cập trực tiếp qua class name

HUFIStudent.uname = "FTU"
print(sv007.uname) #"HUS": thực tế 1 biến của object "uname"
print(sv009.uname) #"FTU": biến của class vẫn không đổi
print(HUFIStudent.uname) #"FTU":truy cập trực tiếp qua class name
```

1.3. Các phép toán đặc biệt khác

Các phép toán được định nghĩ thông qua các hàm đặc biệt như sau:

```
__add__(self, other)   → self + other
__sub__(self, other)   → self - other
__eq__(self, other) → self == other
__lt__(self, other) →  self < other
__len__(self) →  len(self)
```

`__str__(self)` → `print self`

2. Kế thừa

2.1. Kế thừa

Lớp có thể chứa và sử dụng mọi thuộc tính, phương thức của lớp cha đúng như là chúng được định nghĩa tại lớp con. Trong python cho phép thiết lập đa kế thừa.

Cú pháp của Kế thừa:

```
class BaseClass:
   #Body of base class

class DerivedClass(BaseClass):
   #Body of derived class
```

Ví dụ:

```
#HUFIStudent-child class; Student-parent
class HUFIStudent(Student):
   def __init__(self, name, dob, startyear):
     super().__init__(name, dob)
     self.startyear = startyear
   def welcome(self, name, year):
     if year = self.startyear:
        print("Welcome: " + name)
```

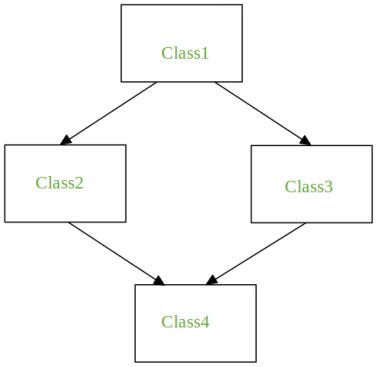
2.2. Đa kế thừa

Python cho phép một lớp kế thừa từ nhiều lớp cơ sở, theo cú pháp:

```
Class Base1:
        Body of the class

Class Base2:
     Body of the class

Class Derived(Base1, Base2):
     Body of the class
```



Vấn đề Diamond (con thoi) xảy ra khi một class con được thừa kế từ cả hai classes cha.

```
class Class1:
    def m(self):
        print("In Class1")

class Class2(Class1):
    def m(self):
        print("In Class2")

class Class3(Class1):
    def m(self):
        print("In Class3")

class Class4(Class2, Class3):
    pass

obj = Class4()
obj.m() #In Class2. Nếu gọi Class4(Class3, Class2) thì sẽ là "In Class 3"
```

2.3. Lớp trừu tượng

Lớp trừu tượng giúp xây dựng các kiểu dữ liệu chung và định nghĩa giao diện chung cho các lớp con. Có các đặc điểm sau:

- Lớp trừu tượng là một lớp mà chứa ít nhất một phương thức trừu tượng (abstract method).
- Phương thức trừu tượng là phương thức chỉ có khai báo mà không có cài đặt.
- Lớp trừu tượng không thể được khởi tạo trực tiếp, mà chỉ dùng để làm mẫu cho các lớp con.

- Để định nghĩa lớp trừu tượng, sử dụng từ khóa ABC và @abstractmethod.

```
#Abstract Base classes(ABC)
from abc import ABC, abstractmethod
#Khai báo lớp trừu tượng Polygon với phương thức trừu tượng noofsides
class Polygon(ABC):
    @abstractmethod
    def noofsides(self):
        pass
#Lớp Triangle là lớp dẫn xuất của lớp Polygon
class Triangle(Polygon):
    #overriding abstract method
    def noofsides(self):
        print("I have 3 sides")
#Lớp Pentagon là lớp dẫn xuất của lớp Polygon
class Pentagon(Polygon):
    #overriding abstract method
    def noofsides(self):
        print("I have 5 sides")

class Hexagon(Polygon):
    #overriding abstract method
    def noofsides(self):
        print("I have 6 sides")

class Quadrilateral(Polygon):
    #overriding abstract method
    def noofsides(self):
        print("I have 4 sides")
```

3. Đa hình: Đa hình trong Python là khái niệm quan trọng trong lập trình hướng đối tượng. Cho phép một đối tượng có thể thay đổi hình thức hoạt động dựa trên kiểu dữ liệu của đối tượng đó (*Sinh viên tham khảo).*

1.2. Giới thiệu bài tập mẫu:

Bài 1.

Thông tin nhân viên gồm:

+ Số lượng nhân viên: thông tin chung của lớp, kiểu số nguyên

+ Tên nhân viên: kiểu chuỗi, Lương: kiểu số nguyên

Các phương thức:

+ Phương thức khởi tạo

+ Phương thức hiển thị thông tin nhân viên

Yêu cầu:

- Khai báo 2 đối tượng a và b kiểu nhân viên
- Xuất thông tin từng nhân viên
- Xuất số lượng nhân viên
- Tạo danh sách gồm n nhân viên (n thông tin nhập từ phím)
- In danh sách nhân viên vừa nhập
- Tính tổng lương nhân viên
- Tìm nhân viên có lương cao nhất
- Sắp xếp danh sách nhân viên theo tên tăng dần

Hướng dẫn:

```
#Khai báo lớp NhanVien
class NhanVien(object):
   dem = 0
   def __init__(self, name, salary):
      self.name = name
      self.salary = salary
      NhanVien.dem += 1
   def hien_thi_so_luong(self):
      print ("Tổng số nhân viên được tạo: " + str(NhanVien.dem))
   def hien_thi_nhan_vien(self):
      print ("Tên: ",self.name,", Lương: ", self.salary)
   def __str__(self):
      return ("Tên: "+ str(self.name)+", Lương: "+str(self.salary))

#định nghĩa các hàm trên dsnv
def NhapDSNhanVien(lstNV):
   n = int (input("Nhap so phan tu:"))
   for i in range (n):
      print("Nhap thong tin thu: ",i)
      ten = input("Nhap ten:")
      tuoi = int(input("luong:"))
```

```
        Nv = NhanVien(ten,tuoi)
        lstNV.append(Nv)

def InDSNhanVien(lstNV):
    for x in lstNV:
        print(x)

#Thực thi
a = NhanVien("Hung", 5000)
b = NhanVien("Quan", 3500)
a.hien_thi_so_luong()
b.hien_thi_nhan_vien()

lst = []
NhapDSNhanVien(lst)
InDSNhanVien(lst)
```

Bài 2.

Thông tin lớp Person gồm:

- Tên: kiểu chuỗi
- Tuổi: số nguyên

Tạo lớp Person với hàm khởi tạo đầy đủ tham số và hàm in tất cả thông tin của lớp

Thông tin lớp Student gồm:

- Tên: kiểu chuỗi
- Tuổi: sô nguyên
- Mã sinh viên: kiểu chuỗi
- Điểm trung bình: kiểu số thực

Tạo lớp Student với yêu cầu sau:

- Phương thức khởi tạo đầy đủ tham số (có kế thừa từ lớp cha)
- Phương thức in tất cả thông tin của lớp (có kế thừa từ lớp cha)
- Phương thức tính tính điểm chữ (A, B, C, D, F)
- Phương thức tính kết quả (đậu/ rớt)

Yêu cầu:

- Tạo danh sách sinh viên, thông tin nhận từ file
- In danh sách sinh viên
- Đếm số lượng sinh viên đậu và in danh sách sinh viên đậu
- Thống kê số lượng sinh viên theo từng thang điểm

Hướng dẫn:

```
class Person:
   def __init__(self, name, age):
      self.name = name
      self.age = age
   def info(self):
      print(self.name + ",", self.age, "years old.")

#Student inherits from Person
class Student(Person):
   def __init__(self, name, age, id, score):
      super().__init__(name, age)
      self.id = id
      self.score = score
   def infoStudent(self):
      print(self.name + ",", self.age, "years old, id:",
self.id, "score:", self.score)
   def scoreAlpha(self):
      if self.score >=8.5:
           return 'A'
      elif self.score >=7:
           return 'B'
      elif self.score >=5.5:
           return 'C'
      elif self.score>=4:
           return 'D'
      else:
           return 'F'
     def resultStudent(self):
       if self.score >=4.0:
            return 'Đậu'
       else:
            return 'Rớt'
#Thực thi
minh = Person("Minh", 40)
minh.info()
```

```
son = Student("Son", 30, "0469191517",7.9)
son.infoStudent()
print("Kết quả môn học la:", son.resultStudent())
print("diem chu la:", son.scoreAlpha())

#định nghĩa hàm trên dssv
def NhapDSSinhVien(lstSV):
   n = int (input("Nhap so phan tu:"))
   for i in range (n):
      print("Nhap thong tin SV thu: ",i)
      ten = input("Nhap ten sinh vien:")
      tuoi = int(input("Nhap tuoi sinh vien:"))
      ma = input("Nhap ma so sinh vien:")
      diem = float(input("Nhap diem:"))
      sv = Student(ten,tuoi,ma,diem)
      lstSV.append(sv)

#định nghĩa hàm trên dssv
def DocFile(path,lstSV):
   file = open(path, mode='r', encoding='utf-8')
   lines = file.readlines()
   for line in lines:
      data = line.strip()
      arr = data.split('#')
      sv = Student(arr[0],int(arr[1]),arr[2],float(arr[3]))
      lstSV.append(sv)
      file.close()

def InDSSinhVien(lstSV):
   for x in lstSV:
      x.infoStudent()

#Thực thi các hàm trên dssv
lstSV = []
NhapDSSinhVien(lstSV)
```

```
InDSSinhVien(lstSV)
```

Bài 3.

Thông tin nhân viên phòng kế hoạch gồm:

- Thông tin chung nhân viên trong bài 1
- Thông tin riêng:
  + Số ngày công ghi nhận: kiểu số nguyên
- Phương thức khởi tạo của lớp nhân viên phòng kế hoạch
- Phương thức tính tiền thêm giờ cho nhân viên phòng kế hoạch theo công thức: số ngày ghi nhận >=21, mỗi ngày làm thêm sẽ được 100k
- Phương thức tính lương thực nhận cho nhân viên

Thông tin nhân viên phòng kinh doanh gồm:

- Thông tin chung nhân viên
- Thông tin riêng:
  + Số sản phẩm thực tế: kiểu số nguyên
- Phương thức khởi tạo của lớp nhân viên phòng kinh doanh
- Phương thức tính tiền thêm giờ cho nhân viên phòng kinh doanh theo công thức: số sản phẩm *50.
- Phương thức tính lương thực nhận cho nhân viên

Yêu cầu:

1. Nhập danh sách n nhân viên, với dữ liệu nhập từ file
2. In danh sách nhân viên
3. In tổng lương của tất cả nhân viên
4. In thông tin nhân viên có lương cao nhất
5. Sắp xếp danh sách theo tên tăng dần

Ghi chú: Nội dung file

```
1#Hoa # 5500 # 25
1#Van # 7500 # 22
1#Phat# 6000 # 18
2#Trang#3500 #25
2#Hong #1500 # 60
1#Nam # 6500 # 20
1#Khanh# 6500 # 18
2#Vinh #4500 # 60
```

Hướng dẫn:

```
from abc import ABC, abstractmethod
```

```
class NhanVien(ABC):
   dem = 0
   def __init__(self, ten, luongcoban):
         self.ten = ten
         self.luongcoban = luongcoban
         NhanVien.dem += 1
   def hien_thi_so_luong(self):
       print ("Tổng số nhân viên được tạo: " + str(NhanVien.dem))
   def hien_thi_nhan_vien(self):
       print ("Tên: ",self.ten,", Lương: ", self.luongcoban)
   def __str__(self):
       return ("Tên: "+ str(self.ten)+", Lương:"+str(self.luongcoban))
   def LuongThucNhan(self):
         return self.luongcoban + self.ThuNhapTangThem()
   @abstractmethod
   def ThuNhapTangThem(self):
           pass

class NhanVienKeHoach(NhanVien):
   def __init__(self, ten, luongcoban, songay):
      super().__init__(ten, luongcoban)
      self.songay = songay
   def ThuNhapTangThem(self):
     if self.songay>21:
          return (self.songay - 21)*100
     return 0
class NhanVienKinhDoanh(NhanVien):
   def __init__(self, ten, luongcoban, sosanpham):
      super().__init__(ten, luongcoban)
      self.sosanpham = sosanpham
   def ThuNhapTangThem(self):
      return self.sosanpham*50
```

2. Bài tập tại lớp

Bài 1.

Một nhân viên trong công ty ABC có các thông tin: mã nhân viên, tên nhân viên, phòng ban, chức vụ, hệ số lương, thâm niên công tác và số ngày làm việc trong tháng.

Chức vụ chỉ nhận 2 giá trị là "Lãnh đạo" và "Nhân viên". Lương của nhân viên hàng tháng được tính theo công thức:

Lương = hệ số lương * Lương cơ bản * hệ số thi đua +1100 + Phụ cấp

Trong đó lương cơ bản do nhà nước qui định và áp dụng cho tất cả các nhân viên có giá trị tại thời điểm hiện tại là 1210.

Hệ số thi đua được xác định như sau:

- Nếu chức vụ là "Lãnh đạo" thì hệ số là 1
- Nếu chức vụ là "Nhân viên" thì hệ số thi đua được xác định như sau:
  - Nếu số ngày đi làm lớn hơn 22 thì hệ số thi đua bằng 1.0
  - Nếu số ngày đi làm lớn hơn 20 thì hệ số thi đua bằng 0.8
  - Còn lại hệ số thi đua là 0.6
- Phụ cấp được xác định như sau:
  - Nếu chức vụ là "Lãnh đạo" thì phụ cấp chức vụ là 2000
  - Nhân viên không có phụ cấp chức vụ

Xây dựng lớp Nhân viên theo mô tả như trên.

Xây dựng lớp danh sách nhân viên để lưu các nhân viên của công ty với các yêu cầu quản lý như sau:

a) Thông tin nhân viên được nhập vào danh sách từ file XML do công ty qui định (sinh viên tự đặt ra qui định của file XML).
b) Xuất thông tin tất cả các nhân viên trong công ty.
c) Xuất ra thông tin nhân viên theo từng nhóm hệ số thi đua.
d) Xuất danh sách nhân viên theo phòng cho trước.
e) Xuất danh sách nhân viên có chức vụ là "Lãnh đạo"
f) Tính tổng lương công ty phải trả cho toàn bộ nhân viên.
g) Xóa các nhân viên có số ngày làm nhỏ hơn 10 trong danh sách.
h) Xuất danh sách những nhân viên không phải là lãnh đạo mà có số ngày làm việc lớn hơn 22.
i) Xuất danh sách các nhân viên có hệ số lương từ 4.34 trở lên và ở phòng "Tài vụ".

Bài 2.

Xây dựng ứng dụng quản lý danh sách các giao dịch trong một của hàng mua bán vàng bạc, đá quý. Hệ thống quản lý 2 loại giao dịch:

- **Giao dịch vàng:** Mã giao dịch, ngày giao dịch (ngày/tháng/năm), đơn giá, số lượng, loại vàng có 3 loại: 18k, 24k, 9999.

  Thành tiền bán vàng được tính như sau: thành tiền = số lượng * đơn giá.

- **Giao dịch tiền tệ:** Mã giao dịch, ngày giao dịch (ngày/tháng/năm), đơn giá), số lượng, loại tiền tệ có 3 loại: USD, EUR, AUD, loại giao dịch mua hoặc bán.

  Thành tiền mua/bán tiền tệ được tính như sau:

- Nếu loại giao dịch là "mua" thì: thành tiền = số lượng * đơn giá
- Nếu loại giao dịch là "bán" thì: thành tiền = (số lượng * đơn)* 1.05

Xây dựng các lớp theo mô tả. Lớp GiaoDich với các thuộc tính và phương thức chung. Giao dịch vàng hay giao dịch tiền tệ cũng là giao dịch với các thuộc tính và các phương thức riêng cần thiết.

Yêu cầu:

1. Nhập/ Đọc từ file, xuất danh sách các giao dịch.
2. Tính tổng thành tiền cho từng loại.
3. Xuất danh sách các giao dịch được thực hiện trong ngày: 27/02/2023
4. Xuất thông tin của giao dịch có thành tiền cao nhất.
5. Sắp xếp danh sách giao dịch theo từng loại. Trong cùng một loại sắp tăng dần theo thành tiền.