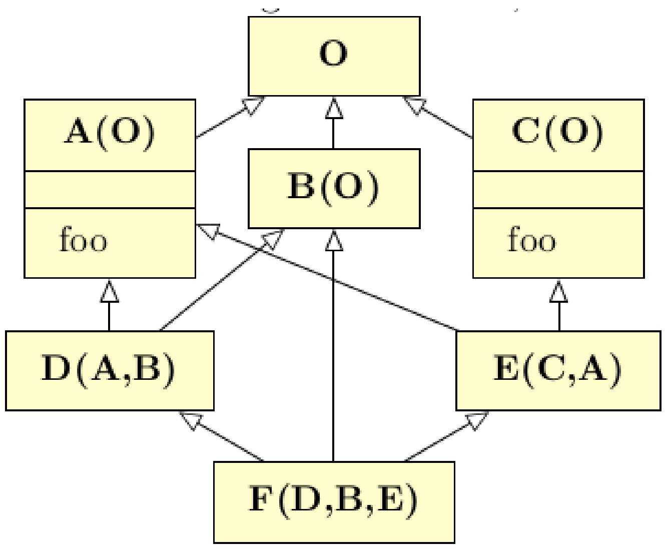
Tutorial 4 : Object Oriented Programming

Thang Huu Thang
thang.nguyen99@hcmut.edu.vn

Question 1

Given the following class schema, (left to right order)



(a) Apply method resolution order of Python 3 to find the class search path of each class in the above class schema? If there are errors in the class search path, please change the order of super classes of some class to make all the class search path success.
Answers:
L(O) = [O]
L(A) = [A] + merge(L(O), [O]) = [A, O]
L(B) = [B] + merge(L(O), [O]) = [B, O]
L(C) = [C] + merge(L(O), [O]) = [C, O]
L(D) = [D] + merge(L(A), L(B), [A, B]) = [D, A, B, O]
L(E) = [E] + merge(L(C), L(A), [C, A]) = [E, C, A, O]
L(F) = [F] + merge(L(D), L(B), L(E), [D, B, E]) = **?!?!**
Fix:
F(D, E, B) $\Rightarrow$ L(F) = F + merge(L(D), L(E), L(B), [D, E, B]) = [F, D, E, C, A, B, O]

(b) If x contains an object of F, which will method foo be called by x.foo() using your successful class schema?
Answers: Method **foo** of class C will called

Question 2

(a) Write class Rational using Python.

(b) Make sure that when creating a Rational object without any argument (Rational()), object Rational whose n is 0 and d is 1 is created.

(c) Rewrite method + so that it can accepts parameter **that** in type **int**. Make sure that the new code calls recursively to the old code.

```
class Rational:
def __init__(self, n=0, d=1):
    if d == 0:
        raise SystemExit("d == 0")
    else:
        self.n = n
        self.d = d
        self.__g = self.__gcd(n, d)
        self.numer = n // self.__g
        self.denom = d // self.__g

def __gcd(self, a, b):
    if b == 0:
        return a
    return self.__gcd(b, a % b)

def __add__(self, that):
    if isinstance(that, Rational):
        return Rational(self.numer * that.denom + that.numer * self.denom,
        self.denom * that.denom)
    elif isinstance(that, int):
        return self + Rational(that)
    else:
        return None

def __str__(self):
    return str(self.numer) + "/" + str(self.denom)
```

Question 3

Redefine the example on Case class using Python

(a) Write method print for class **Number** to print the value of field **num** in **Number**.

(b) Make an object that represents the expression **(x + 0.2) * 3** and assign it to variable *t*.

(c) Write method eval that can evaluate an expression return a **Number** object. The operators which may be appeared in an expression are "+", "-", "*", "/". Assume that the value of all variables is 1. For example, t.eval().print() => 3.6

```
from abc import ABC

class Expr(ABC):
    pass

class Var(Expr):
    def __init__(self, name):
        self.name = str(name)

class Number(Expr):
    def __init__(self, num):
        self.num = num

    def print(self):
        print(self.num)

class UpOp(Expr):
    def __init__(self, operator, arg):
        self.operator = operator
        self.arg = arg

class BinOp(Expr):
    def __init__(self, operator, left, right):
        self.operator = str(operator)
        self.left = left
        self.right = right

    def eval(self):
        if self.operator == '*':
            return Number(self.left * self.right)
        if self.operator == '/':
            return Number(self.left / self.right)
        if self.operator == '+':
            return Number(self.left + self.right)
        if self.operator == '-':
            return Number(self.left - self.right)
```