Exercise 1:

You are required to build the class diagram to manage some staffs and tasks. ***Each staff is managed by the manager***. The staff includes information: Staff ID, name, gender, basic salary, bonus salary (if the staff is a manager, he/she will be paid a bonus salary). ***Each task is performed by one or many staffs***. This system need to record ***working hours*** of each staff per a task. Information of task includes: id, title, begin date, end date, total hours.

1)OOA

Main noun: Staff ( ID,name, gender, basic salary)

Main noun: Manager(ID,name, gender, basic salary, bonus salary)

Main noun: Task (id, title, begin date, end date, total hours)

Working hours

2)OOD: UML

|  |
| --- |
| Staff  perform |
| ID: number name: text gender:text basicSalary: number |
| //methods  toString(){…} |

0..\*

|  |
| --- |
| Task |
| ID: number title: text beginDate:number endDate: number totalHours:number |
| //methods |

1..\*

Is performed

1..\*

Is managed

extends

1

Manage

|  |
| --- |
| Manager |
| bonusSalary:number |
| //methods |

3)OOP

OFB

OFA

Class Manager  
 id, name, gender, basicSalary, bonusSalary

Manager(){ super(); bonusSalary=.. }  
  
Manager(id, name, …){super(…); bonusSalary=……}  
  
getBonusSalary(){…}

setBonusSalary() {..}

outputStaff(){........}

~~outputManage~~r(){ outputStaff(); sout….}  
  
inputStaff()  
~~inputManager~~(){ inputStaff(); bonusSalary=..}

Class Staff  
 id, name, gender, basicSalary

Staff(){..}  
  
  
Staff(id, name, …){…}  
  
getId(){..}

setId(){..}  
  
outputStaff(){… warning()}

….

inputStaff(){..}

warning(){….}

Manager() 100  
Manager (id,..) 200  
Staff() 10  
Staff(id,..) 20  
getId() 30  
setId() 40   
outputStaff() 500  
inputStaff() 600

Warning() 70  
getBonusSalary() 300   
setBonusSalary() 400   
~~inputManager() 600~~   
~~outputManager () 500~~

Staff() 10  
Staff(id,..) 20  
getId() 30  
setId() 40  
outputStaff() 50  
 inputStaff() 60  
warning() 70

Java copy cai nay

10

20

30

40

100

200

50

300

60

70

400

500

600

Stack dynamic

x=1000

Id=1  
name=no  
gender=female  
basicSalary=100

Id=1  
name=tien  
gender=female  
basicSalary=100  
mng=~~null~~ 2000

obj=1000

Pointer=0FA

Id=2  
name=kiet  
gender=nam  
basicSalary=100  
mng=null

Bonus=90

Obj2=2000

Id=2  
name=pheo  
gender=male  
basicSalary=100  
bonusSalary=80

y=2000

Pointer=0FB

Id=2  
title=”aap2”  
...

T2=400

Id=1  
title=”aap1”  
...

T1=300

staff=~~null~~1000  
task=~~null~~ 400

Workinghour=10

Ts2=700

Ts1=600

staff=~~null~~1000  
task=~~null~~ 300

Workinghour=3

Exercise 2:

You are required to build the class diagram for a health hospital. This diagram is used to manage some ***owners,*** some ***pets*** and ***services***. The owner includes information: ID, name, address. Each owner has one or many pets. Information of pet includes: id, name, birthday, gender. When a pet uses any services the system needs to record them to charge money. Each service contains: Id, name, price

Services

-Id:int

-name:String

-price:int

//methods

Owner  
  
-ID:String

- name:String

- address:String

//methods

Pet

-Id:int

-name:String

-birthday:String

-gender:String

//methods

1..\*

1..\*

1..\*

1

id=1

name=”chim”

bd=”1/1/2000”

gender=”cai”

owner=~~null~~ 400

P1=1000

id=2

name=”meo”

bd=”1/1/2014”

gender=”cai”

owner=~~null~~ 400

P2=2000

id=”O123”

name=”Tien Minh”

address=”SG”

owner=400

id=1

name=”tam”

price=100000

S1=500

id=2

name=”cat mong”

price=200000

S2=700

X=800

pet=~~null~~ 1000  
service=~~null~~ 500

pet=~~null~~1000  
service=~~null~~ 700

y=900

pet=~~null~~ 2000  
service=~~null~~ 500

z=600

List1=100

List:~~null~~200  
MAX=100  
n=~~0~~ 1 2;

300  
  
400  
  
500

0

1

Id=1  
Name:”chim  
gender:…

2

99

Id=2

Name=”cho”  
gender=”…”

Idremove=2

Id:3

Name:”vit”

……

P=400

Exercise 3:

Build a class diagram to manage students and courses of FPTU. Each student includes: id, name, address, gender. A student can enroll in some courses. Each course contains: code, name, credits. Addition, each student belongs to the campus. Each campus includes: code, name, address.

1..\*

1..\*

Courses

Student

Campus

1..\*

1

Exercise 4:

Build a class diagram to manager Customers, Items, Orders, invoices. Each Item: id, name, price, status ( 1: đã bán, 0: chưa bán). Customer includes: id, name, address. Order contains: id, orderDate, shipDate, TotalOfQuantity. Each customer has one or many orders and an order belongs to the customer. Each ***Order has only one invoice.***

Customer

order

1..\*

1

Item

1..\*

1..\*

1

1

Invoice  
ma: Number  
date: String

Example:

Son  
  
  
  
Son(){}  
getter/setter()  
bu(){

Sout(“ha mieng”)

Sout(“ngam nhe”)

Sout(“nut nut nut”)

Sout(“no”)

}

~~Bu2(){…..}~~

Father  
  
  
  
Father(){}  
getter/setter()  
bu(){  
 sout(“xoa tay”);  
 sout(“bop nhe”);  
 sout(“phe”);  
}

Father f=new Father();   
f.bu();

OFB

Class Father  
 fields

Father(){...}  
getter(){..}  
  
bu(){... “phe”}

OFA

Class Son  
 fields

Son(){...}  
getter(){..}  
  
  
bu(){ .... “no”}

Son s=new Son();

s.bu();

Father() 10  
getter () 20  
...  
bu() 70

Father() 10  
getter () 20  
..  
Son() 100  
bu() 500

Father f2=new Son(); f2.bu();

~~((Son)F2).bu2()~~

10

100

70

500

fields

F2=400

Pointer: OFB

Pointer: OFA

fields

F=500

15db9742

x=15db9742

Id=1  
name=”abc”  
.....

<<abstract>

Car

serial:string  
  
Car(){...}

Car(String){...}  
getter/setter(){.....}  
brake(){.......}

Abstract void makeTailLight();

CarVN  
  
discount:int

CarVN(){...}

CarVN(String){...}  
getter/setter(){.....}  
void makeTailLight(){   
 sout(“dinh hoa vao duoi xe”);  
}

CarThaiLand  
  
price:int

CarThaiLand(){...}

CarThaidLan(String){...}  
getter/setter(){.....}  
void makeTailLight(){   
 sout(“dinh con voi vao duoi xe”);  
}

Plant:   
fields: name, year  
methods: useWater(), grow()

Animal:

Fields: name,gender  
Methods:useWater(), grow(), run()

Tester:  
LivingBeing x=new Plant();  
x.useWater();  
LivingBeing y=new Animal();  
y.useWater();  
~~((Plant)x).grow();  
((Animal)y).grow();~~x.grow();  
y.grow()

<<Abstract>>

LivingBeing

Name:text

LivingBeing()

LivingBeing(text)  
useWater(){  
 sout(“used water”);   
}

Abstract void grow();  
~~eatBug()~~

Exercise 5:

extends

implement

<<interface>>  
 Food

void eatBug();// abstract

Thằn lằn

eatBug(){   
 sout(“dung luoi”)

sout(“bo vao mieng”)  
 sout(“nhai”)  
 }

Venus(bat ruoi)  
  
  
eatBug(){   
 sout(“day nap am”)  
 sout(“tiet hoa chat nat con bo”)

Sout(“am chat dinh duong qua mang mang ong”)   
}

Animal

gender:text

Animal ()

Animal(text,text)  
grow(){  
 sout(“by food”); }

Cat

Orchid

Price:int

Plant

year:int

Plant()

Plant(text,int)  
grow(){  
 sout(“by light”); }