คณะวิทยาศาสตร์

มหาวิทยาลัยขอนแก่น

SC311003 Object-Oriented Programming ประจำปี 2563 ภาคปลาย

วันที่สอบ 20 เม.ย. 2564 เวลา 13.00-16.00 น.

อาจารย์ประจำวิชา รศ. ดร. ปัญญาพล หอระตะ

ชื่อ ปรีชานนท์ สกุล ชาติไทย รหัสประจำตัว 633020404-3

Section 2

คำสั่ง

ข้อสอบมีทั้งหมด 16 คะแนนเต็ม 95 คะแนน ให้ทำทุกข้อ

1. เขียนคำตอบลงในไฟล์ .docx หรือ pdf ตั้งชื่อไฟล์เป็น รหัสนักศึกษา\_answer.docx หรือ .pdf

2. ให้ตั้งชื่อโปรเจ็คชื่อ exam\_รหัสนักศึกษา แบ่งแต่ละข้อเอาไว้ในแต่ละ package เช่น

exam\_01 หมายถึงข้อที่ 1 แล้ว export project ทั้งหมดที่ทำข้อสอบให้อยู่ในรูปของไฟล์ .zip ชื่อเดียวกับโปรเจ็ค

3. คำตอบทุกคำตอบนักศึกษาต้องเขียนด้วยคำพูดของตัวเองห้ามคัดลอกมาจากที่ใดๆ ถ้าเหมือนหรือใกล้เคียงจะถือว่าทุจริต ทั้งนี้จะขึ้นอยู่ที่ดุลพินิจของอาจารย์ถือว่าเป็นอันสิ้นสุด

สิ่งที่ส่ง:

1.ไฟล์คำตอบ

2. ไฟล์ .zip

**กำหนดให้**

public class Person{

private String id;

protected String name;

protected int age;

public String address;

public Person(){ }

public Persion(String id ,String name, int age, String address){

this.id=id;

this.name=name;

this.age =age;

this.address = address;

}

public String getId(){

return id;

}

public String getName(){ return name;}

public String toString(){  
 return id + “:” + name + “ age:”+age + “ address:”+address;

}

private String note(){

return “my secrete”;

}

}

1. **กำหนดให้ id แทนเลขบัตร name แทนชื่อ age แทนอายุ address แทนที่อยู่จากคลาส Person สมมุติว่ามีคลาส นักเรียน (Student) และ คนงาน (Worker) ซึ่งต้องการสืบทอดจากคลาส Person โดยที่ข้อมูลพื้นฐานของนักเรียนคือ เลขบัตร รหัสประจำตัวนักศึกษา ชื่อ อายุ ที่อยู่ เกรดเฉลี่ย ขณะที่ คนงานมีข้อมูล คือ เลขบัตร รหัสประจำตัวคนงาน ชื่อ อายุ ที่อยู่ อัตราค่าแรงต่อวัน(baht/day) จำนวนวันที่ทำงาน (days) จงเขียนคลาสเพื่อสร้างคลาส Student และคลาส Worker ด้วยภาษาจาวาเพื่อสืบทอดจากคลาส Person (10 คะแนน)**

**กรณีทดสอบ:**

Student s = new Student(“34099110”,”6301”,”John”,19,”123 KKU”);

Worker w = new Worker(“34099220”,”W01”,”Giant”,40,”255 KKU”);

s.setGPA(3.5);

w.setRate(300.0);

w.setDays(20);

System.out.println(s);

System.out.println(w.toString());

**จะได้ผลลัพธ์ดังนี้**

**PID:34099110 SID:6301 Name: John Age:19 Address:123 KKU GPA:3.5**

**PID: 34099220 WID:W01 Name: Giant Age:40 Address:255 KKU Rate:300 baht/day Days:20**

คำตอบ:

**รูปภาพประกอบด้วย ข้อความ

คำอธิบายที่สร้างโดยอัตโนมัติ**

**package** exam\_01;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

Student s = **new** Student("34099110","6301","John",19,"123 KKU");

Worker w = **new** Worker("34099220","W01","Giant",40,"255 KKU");

s.setGPA(3.5);

w.setRate(300.0);

w.setDays(20);

System.***out***.println(s);

System.***out***.println(w.toString());

}

}

**package** exam\_01;

**public** **class** Person {

**private** String id;

**protected** String name;

**protected** **int** age;

**public** String address;

**public** Person(){ }

**public** Person(String id ,String name, **int** age, String address){

**this**.id=id;

**this**.name=name;

**this**.age =age;

**this**.address = address;

}

**public** String getId(){

**return** id;

}

**public** String getName(){ **return** name;}

**public** String toString(){

**return** id + ":" + name + " age:"+age + " address:"+address;

}

**private** String note(){

**return** "my secrete";

}

}

**package** exam\_01;

**public** **class** Student **extends** Person {

**private** **double** GPA;

**private** String Sid;

**public** Student(String id ,String Sid,String name, **int** age, String address) {

**super**(id,name,age,address);

**this**.Sid = Sid;

}

@Override

**public** String toString(){

**return** "PID :"+**super**.getId() + "SID:"+Sid +" Name :"+ **super**.getName() +

" age:"+**super**.age + " super.address:"+address + "GPA"+GPA;

}

**public** **void** setGPA(**double** GPA) {

**this**.GPA = GPA;

}

}

**package** exam\_01;

**public** **class** Worker **extends** Person {

**private** **double** rate;

**private** **int** day;

**private** String Sid;

**public** Worker(String id ,String Sid,String name, **int** age, String address) {

**super**(id,name,age,address);

**this**.Sid = Sid;

}

@Override

**public** String toString(){

String address;

**return** "PID :" + **super**.getId()

+ "SID:" + Sid +

" Name :" + **super**.getName() +

" age:" + **super**.age +

"address:" + **super**.address +

"Rate:" + rate + " baht/day "+

"Days:" + day;

}

**public** **void** setRate(**double** rate) {

**this**.rate = rate;

}

**public** **void** setDays(**int** day) {

**this**.day = day;

}

}

1. **จากข้อ 1 จงบอกฟิลด์ และเมธอดที่คลาส Student และ คลาส Worker จะได้รับการสืบทอดจากคลาส Person(5 คะแนน)**

คำตอบ:

**protected** String name;

**protected** **int** age;

**public** String address;

**public** String getId()

**public** String getName()

**public** String toString()

1. **จากข้อ 1 จงอธิบายว่า (5 คะแนน)**

super( pid, sid, sname,sage,saddress);

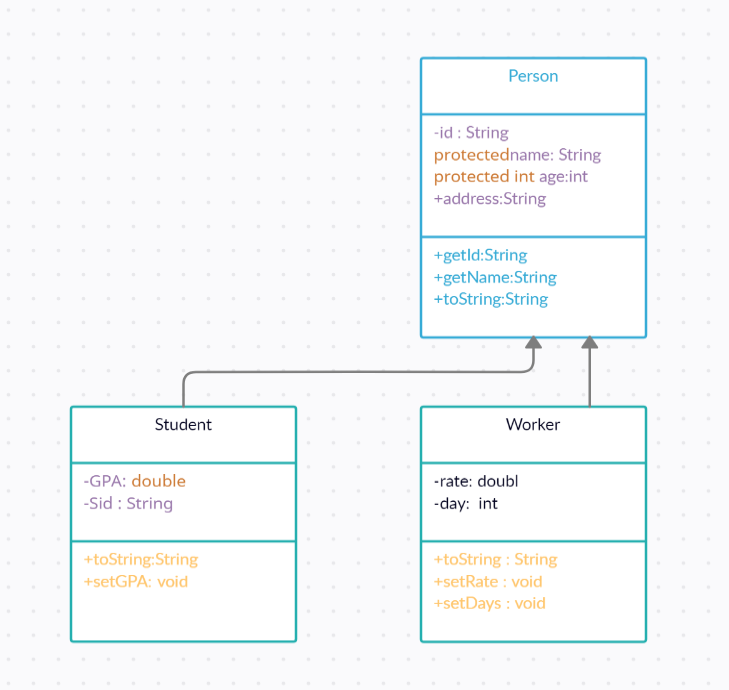
**คำสั่งนี้หมายถึงอะไร และควรเขียนในส่วนใดของคลาสลูก**

คำตอบ:

**super() เป็นการส่งข้อมูลเข้า constructor ของ Class แม่ ใช้เพื่อเรียก ตัวแปร หรือ method ของ Class แม่ที่เราสืบทอดมานั้นเอง**

1. **จากข้อ 1 จงเขียนเขียนแผนภาพ UML แสดงความสัมพันธ์ระหว่างคลาสแม่และคลาสลูก(5 คะแนน)**

คำตอบ:

****

**กำหนดให้ (ใช้ตอบคำถามในข้อ 5,6,7)**

interface IAddition{

public double add(double x, double y);

}

1. **จงเขียนคลาส MyComputing เพื่อให้ทำในสิ่งต่อไปนี้ได้(5 คะแนน)**

IAddition iobj = new MyComputing();

double result = iobj.add(5.0,6.0);

System.out.println(“result=”+result);

คำตอบ:



**package** exam\_05;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

IAddition iobj = **new** MyComputing();

**double** result = iobj.add(5.0,6.0);

System.***out***.println("result="+result);

}

}

**package** exam\_05;

**public** **interface** IAddition {

**public** **double** add(**double** x, **double** y);

}

**package** exam\_05;

**public** **class** MyComputing **implements** IAddition {

@Override

**public** **double** add(**double** x, **double** y) {

**return** x+y;

}

}

1. **จงแก้ไขคลาส MyComputing ใหม่ และเพิ่มอินเตอร์เฟซ ISubtraction เพื่อให้สามารถประมวลผลผ่าน คำสั่งต่อไปนี้(5 คะแนน)**

MyComputing iobj = new MyComputing();

double resultAdd = iobj.add(5.0,6.0);

double resultSub = iobj.subtract(10.5,5.5);

System.out.println(“result of addition=”+resultAdd);

System.out.println(“result of subtraction=”+resultSub);

ผลลัพธ์ :

result of addition=11.0

result of subtraction=5.0

คำตอบ:

รูปภาพประกอบด้วย ข้อความ

คำอธิบายที่สร้างโดยอัตโนมัติ

**package** exam\_06;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

MyComputing iobj = **new** MyComputing();

**double** resultAdd = iobj.add(5.0,6.0);

**double** resultSub = iobj.subtract(10.5,5.5);

System.***out***.println("result of addition="+resultAdd);

System.***out***.println("result of subtraction="+resultSub);

}

}

**package** exam\_06;

**public** **interface** IAddition {

**public** **double** add(**double** x, **double** y);

}

**package** exam\_06;

**public** **interface** ISubtraction {

**public** **double** subtract(**double** x, **double** y);

}

**package** exam\_06;

**public** **class** MyComputing **implements** IAddition,ISubtraction{

@Override

**public** **double** add(**double** x, **double** y) {

**return** x+y;

}

@Override

**public** **double** subtract(**double** x, **double** y) {

**return** x-y;

}

}

1. **จากข้อ 5 จงคำสั่งเพื่อสร้าง Lambda expression และการเรียกใช้เพื่อให้ได้ผลลัพธ์เหมือนกับในข้อ 5(5 คะแนน)**

คำตอบ:

**รูปภาพประกอบด้วย โต๊ะ

คำอธิบายที่สร้างโดยอัตโนมัติ**

**package** exam\_07;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

IAddition ia =(a,b)->(a+b);

System.***out***.println(ia.add(15.5,4.5));

ISubtraction is =(a,b)->(a-b);

System.***out***.println(is.subtract(15.5,5.5));

}

}

**package** exam\_07;

**public** **interface** IAddition {

**public** **double** add(**double** x, **double** y);

}

**package** exam\_07;

**public** **interface** ISubtraction {

**public** **double** subtract(**double** x, **double** y);

}

**package** exam\_07;

**public** **class** MyComputing **implements** IAddition,ISubtraction{

@Override

**public** **double** add(**double** x, **double** y) {

**return** x+y;

}

@Override

**public** **double** subtract(**double** x, **double** y) {

**return** x-y;

}

}

1. **จงประกาศ interface ชื่อ IWorkerComparator เพื่อบรรจุฟังก์ชัน compare( , ) แล้วคืนค่า int เพื่อเปรียบเทียบของวัตถุของคลาส Worker สองคน แล้วคืนค่า int แล้วสร้างคลาส WorkerComparator เพื่ออิมพลีเมนต์ ฟังก์ชัน compare( , )ของ IWorkerComparator ซึ่งคืนเป็นไปได้ 3 ค่า ดังนี้ (10 คะแนน)**

* **ถ้าอายุของคนแรกน้อยกว่าอายุของคนที่สองจะคืน -1**
* **ถ้าอายุของคนแรกเท่ากับอายุของคนที่สองจะคืน 0**
* **ถ้าอายุของคนแรกมากกว่าอายุของคนที่สองจะคืน 1**

คำตอบ:

****

**package** exam\_08;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

WorkerComparator sa = **new** WorkerComparator();

System.***out***.println(sa.compare(40,50));

System.***out***.println(sa.compare(10,10));

System.***out***.println(sa.compare(35,22));

}

}

**package** exam\_08;

**public** **interface** IWorkerComparator {

**int** compare(**int** a, **int** b);

}

**package** exam\_08;

**public** **class** WorkerComparator **implements** IWorkerComparator{

@Override

**public** **int** compare(**int** a, **int** b) {

**if**((a-b)<0)

{**return** -1;}

**else** **if** ((a-b)==0)

{**return** 0;}

**return** 1;

}

}

1. **จากข้อ 8 จงเขียน Anonymous class ที่สืบทอดจาก WorkerComparator เพื่อเปรียบเทียบโดยใช้เงื่อนไขอัตราค่าแรงต่อวัน (5 คะแนน)**

คำตอบ:



**package** exam\_09;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

Anonymous anno = **new** Anonymous(){

**public** **int** compare(**double** a,**double** b){

**if**((a-b)<0)

{**return** -1;}

**else** **if** ((a-b)==0)

{**return** 0;}

**return** 1;

}

};

System.***out***.println(anno.compare(5000,2000));

}

}

**package** exam\_09;

**public** **class** WorkerComparator **implements** IWorkerComparator{

@Override

**public** **int** compare(**int** a, **int** b) {

**if**((a-b)<0)

{**return** -1;}

**else** **if** ((a-b)==0)

{**return** 0;}

**return** 1;

}

}

**package** exam\_09;

**public** **interface** IWorkerComparator {

**int** compare(**int** a, **int** b);

}

**package** exam\_09;

**public** **class** Anonymous **extends** WorkerComparator {

}

1. **จากข้อ 8 จงเขียน Lambda expression แทนการใช้คลาส WorkerComparator**

**เพื่อเปรียบเทียบเพื่อเปรียบอายุของสองวัตถุต่อไปนี้(5 คะแนน)**

**Person w1 = new Worker(“5010”,”W011”,”Bian”,35, “44 Khon Kaen”);**

**Person w2 = new Worker(“64010”,”W031”,”Ken”,45, “64 Khon Kaen”);**

คำตอบ:

**รูปภาพประกอบด้วย ข้อความ

คำอธิบายที่สร้างโดยอัตโนมัติ**

**package** exam\_10;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

IWorkerComparator iw =(a,b)-> {**if**((a-b)<0)

{**return** -1;}

**else** **if** ((a-b)==0)

{**return** 0;}

**return** 1; };

Person w1 = **new** Worker("5010","W011","Bian",35, "44 Khon Kaen");

Person w2 = **new** Worker("64010","W031","Ken",45, "64 Khon Kaen");

System.***out***.println(iw.compare(w1.age, w2.age));

System.***out***.println(iw.compare(w2.age, w1.age));

}

}

**package** exam\_10;

**public** **interface** IWorkerComparator {

**int** compare(**int** a, **int** b);

}

**package** exam\_10;

**public** **class** Person {

**private** String id;

**protected** String name;

**protected** **int** age;

**public** String address;

**public** Person(){ }

**public** Person(String id ,String name, **int** age, String address){

**this**.id=id;

**this**.name=name;

**this**.age =age;

**this**.address = address;

}

**public** String getId(){

**return** id;

}

**public** String getName(){ **return** name;}

**public** String toString(){

**return** id + ":" + name + " age:"+age + " address:"+address;

}

**private** String note(){

**return** "my secrete";

}

}

**package** exam\_10;

**public** **class** Worker **extends** Person {

**private** **double** rate;

**private** **int** day;

**private** String Sid;

**public** Worker(String id ,String Sid,String name, **int** age, String address) {

**super**(id,name,age,address);

**this**.Sid = Sid;

}

@Override

**public** String toString(){

String address;

**return** "PID :" + **super**.getId()

+ "SID:" + Sid +

" Name :" + **super**.getName() +

" age:" + **super**.age +

"address:" + **super**.address +

"Rate:" + rate + " baht/day "+

"Days:" + day;

}

**public** **void** setRate(**double** rate) {

**this**.rate = rate;

}

**public** **void** setDays(**int** day) {

**this**.day = day;

}

}

**package** exam\_10;

**public** **class** WorkerComparator **implements** IWorkerComparator{

@Override

**public** **int** compare(**int** a, **int** b) {

**if**((a-b)<0)

{**return** -1;}

**else** **if** ((a-b)==0)

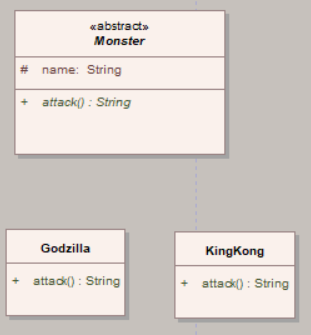
{**return** 0;}

**return** 1;

}

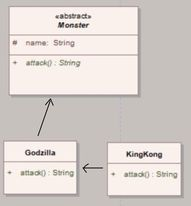
}

**กำหนดให้**



1. **จากแผนภาพคลาสข้างต้น จงเขียนเติมแผนภาพคลาสให้สมบูรณ์ พร้อมทั้งอธิบายว่าคลาสเหล่านี้มีคุณสมบัติอย่างไร (5คะแนน)**

คำตอบ:



ในคลาส Godzilla จะมี method Hit เพื่อทำการรับและส่งดาเมจจาก Monster ทั้ง2ตัว และมี method decreaseHealth เป็นการลดเลือดลงไปอีก KingKong สืบทอดมาจากGodzilla และGodzilla สืบทอดมาจาก Monster

1. **จงอธิบาย overriding คืออะไรแตกต่างจาก Overloading อย่างไร โดยยกตัวอย่างจากการเขียนคำสั่งเพื่อสร้างอะเรย์หรือลิสต์โดยใช้ List<> หรือArrayList<> เพื่อเพิ่มสมาชิก จำนวน 3 ตัว ตัวแรกเป็น Godzilla ตัวที่สองเป็น Godzilla และตัวสุดท้ายเป็น KingKong แล้ววนลูปการใช้ attack() ของ list (5คะแนน)**

คำตอบ:

Overriding บังคับเลยว่ามันจะต้องมาพร้อมกับการสืบทอด Inheritance เพราะว่า Override คือการ implement เมธอดใหม่

Overloading คือการตั้งชื่อ Method ที่เหมือนกันแต่จำนวนการรับ parameter แตกต่างกัน

**เช่น**

**Overloading**

Class Sum {

Int add(int a){

}

Int add(int a,int b)….

}

**overriding**

Class item {

Public int pice(){

Return 20;}

Class Pen extends item{

Public int pice(){

Return 50;}

1. **จากข้อ 12 จงเขียนคลาส Scene เพื่อบรรจุ สัตว์ประหลาด (5คะแนน)**

Scene s = new Scene();

s.addMonster(new Godzilla());

s.addMonster(new KingKong());

s.addMonster(new Godzilla());

s.attackAll();

**ผลลัพธ์:**

Fire!!!

Hit!!!

Fire!!!

คำตอบ:

รูปภาพประกอบด้วย ข้อความ

คำอธิบายที่สร้างโดยอัตโนมัติ

**package** exam\_13;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

Scene s = **new** Scene();

s.addMonster(**new** Godzilla());

s.addMonster(**new** KingKong());

s.addMonster(**new** Godzilla());

s.attackAll();

}

}

**package** exam\_13;

**public** **class** KingKong **extends** Monster{

String attack(){

**return** "Hit!!";

}

}

**package** exam\_13;

**public** **class** Godzilla **extends** Monster{

String attack(){

**return** "Fire!!!";

}

}

**package** exam\_13;

**public** **abstract** **class** Monster {

**abstract** String attack();

}

**package** exam\_13;

**import** java.util.ArrayList;

**import** java.util.List;

**public** **class** Scene {

List<Monster> Mon = **new** ArrayList<Monster>();

**public** **void** addMonster(Monster mon){

Mon.add(mon);

}

**public** **void** attackAll() {

**for** (Monster mon : Mon) {

System.***out***.println(mon.attack());

}

}

}

1. **จาก 13 จงอธิบายหลักการแทนที่ของ Liskov ทำได้อย่างไรและอะไรคือประโยชน์ของการแทนที่ (5คะแนน)**

คำตอบ:

เป็นการแทนที่สำหรับคลาสหนึ่งไปอีกยังคลาส ดังนั้นการเรียกใช้modthodใดMedthodหนึ่งจะเป็นการแทนที่การเรียกใช้นั้นๆได้

1. **จงอธิบายความเหมือนและแตกต่างระหว่าง Abstract class กับ interface ในJava (5 คะแนน)**

คำตอบ:

interface class จะมีแต่ส่วนของการเรียกใช้ฟังก์ชันแต่Abstract class จะมีส่วนของ body ในส่วนของหน้าที่การทำงาน

Class ลูก สามารถ implements Interface Class ได้มากกว่า 1 คลาส

Class ลูก สามารถ extends Abstract Class ได้เพียงคลาสเดียว

1. **จงเขียนโปรแกรมเพื่อแสดงการใช้ try catch, finally, thrown, throws เพื่ออ่านและเขียนไฟล์(15**

**คะแนน)**

**สมมุติว่าไฟล์นำเข้าคือ score.txt**

**50**

**30**

**70**

**95**

**-99**

**ถ้ามีให้เลือกสองเกณฑ์ต่อไปนี้**

**0-49 เกณฑ์ poor**

**50-69 เกณฑ์ pass**

**70-79 เกณฑ์ Good**

**>=80 เกณฑ์ Very Good**

**หรือใช้เกณฑ์ต่อไปนี้**

**0-49 เกณฑ์ D**

**50-69 เกณฑ์ C**

**70-79 เกณฑ์ B**

**>=80 เกณฑ์ A**

**ในกรณี อ่านข้อมูลเป็น -99 ถือว่าข้อมูลไม่มีค่าให้โยน Exception ที่มีข้อความว่า “Have some a missing value” ไปที่ catch แล้วพิมพ์ว่า “Have some a missing value”**

**ผลลัพธ์ที่ได้อาจเป็นดังนี้**

50 pass

30 poor

70 Good

90 Very Good

java.lang.Exception: Missing value

หรือ

50 C

30 D

70 B

90 A

java.lang.Exception: Missing value

เมื่อมีการเรียกใช้ฟังก์ชัน

**public** **static** **void** readScoreGrading(String fileName, IMeasure g)

คำตอบ:

**รูปภาพประกอบด้วย ข้อความ

คำอธิบายที่สร้างโดยอัตโนมัติ**

**package** exam\_16;

**import** java.io.File;

**import** java.util.Scanner;

**public** **class** Main {

**public** **static** **void** readScoreGrading(String a) **throws** Exception {

**try** {

File myObj = **new** File(a);

Scanner myReader = **new** Scanner(myObj);

**while** (myReader.hasNextLine()) {

String data = myReader.nextLine();

**int** i =Integer.*parseInt*(data);

**if**(i ==-99) {

**throw** **new** Exception("Missing value");

}**else** {

**if** (i>= 80) {

System.***out***.println("A");

}

**else** **if**(i>=70) {

System.***out***.println("B");

}

**else** **if**(i>=50) {

System.***out***.println("C");

}

**else**

System.***out***.println("D");

}

}

myReader.close();

} **catch** (NullPointerException ex) {

System.***out***.println("Have some a missing value");

**throw** ex;

}

}

**public** **static** **void** main(String[] args) **throws** Exception {

*readScoreGrading*("score.txt");

}

} }