Description: Waik_OnScreen_RGB_H_1Black

2014 A SEMESTER EXAMINATION

DEPARTMENT Computer Science

PAPER TITLE Programming Usable Systems

TIME ALLOWED Three Hours

NUMBER OF QUESTIONS Eight

IN PAPER

NUMBER OF QUESTIONS Eight

TO BE ANSWERED

VALUE OF EACH QUESTION All questions are worth TEN MARKS each.

GENERAL INSTRUCTIONS Answer ALL EIGHT questions.

SPECIAL INSTRUCTIONS This examination paper is the Answer Booklet. Write your answers in the spaces provided.

CALCULATORS PERMITTED No

THIS EXAMINATION PAPER MUST BE HANDED IN

The following code is the definition of a class. The comments on the right hand side of the listing are line numbers. Study this code carefully before attempting Questions 1 and 2.

class Watch {

public int hour, min; // L1

public boolean isAM; // L2

public Watch(int hour, int min) { // L3

this.hour = hour; // L4

this.min = min; // L5

isAM = true; // L6

}

public String toString() { // L7

String result = pad(hour)+":"+pad(min)+" "; // L8

if (isAM) result += "AM"; // L9

else result += "PM"; // L10

return result; // L11

}

public void incrementMin() { // L12

min++; // L13

if (min>59) { // L14

min=0; // L15

incrementHour(); // L16

}

}

public void incrementHour() { // L17

hour++; // L18

if (hour>12) { // L19

hour=1; // L20

isAM = !isAM; // L21

}

}

protected String pad(int value) { // L22

if (value<10) return "0"+value; // L23

else return ""+value; // L24

}

}

1. Study the Watch class before attempting the following questions.

(a) Write down the line numbers in which a constructor for the Watch class is defined.

L3-6

[2 marks]

(b) Write down the line numbers in which a the method incrementMin() for the Watch class is defined.

L17-21

[2 marks]

(c) What is the output of this fragment of code?

Watch watch = new Watch(5,45);

println( watch );

05:45 AM

[2 marks]

(d) What is the output of this fragment of code?

Watch anotherWatch = new Watch(10,0);

anotherWatch.isAM=false;

anotherWatch.min=59;

println( anotherWatch );

anotherWatch.incrementMin();

println( anotherWatch );

10:59 PM

11:00 PM

[2 marks]

(e) What is the output of this fragment of code?

Watch first, second;

first = new Watch(11,21);

second = first;

second.incrementMin();

println( first );

println( second );

11:22 PM

11:22 PM

[2 marks]

2. You will now write down the code for a subclass of Watch.

A normal clock displays hours between 01 and 12, with “AM” indicating the morning and “PM” indicating the afternoon, e.g. “05:43 AM”, “11:21 PM”.

A 24-hour clock, on the other hand, displays hours between 00 (midnight) and 23 (an hour before midnight) and there is no “AM” or “PM” indicator, e.g. “00:12”, “05:43”, “23:31”.

The Watch class can be extended using inheritance to define a 24-hour watch class called TwentyFourHourWatch.

Assume that a TwentyFourHourWatch class exists. Here is some test code that creates both Watch and TwentyFourHourWatch objects:

Watch watch1 = new Watch(9, 58);

Watch watch2 = new TwentyFourHourWatch(9, 58);

println(watch1+"\t"+watch2);

for (int index=0; index<3; index++) {

watch1.incrementHour();

watch2.incrementHour();

}

println(watch1+"\t"+watch2);

watch1.incrementHour();

watch2.incrementHour();

println(watch1+"\t"+watch2);

for (int index=0; index<5; index++) {

watch1.incrementMin();

watch2.incrementMin();

}

println(watch1+"\t"+watch2);

The output of this test code is:

09:58 AM 09:58

12:58 AM 12:58

01:58 PM 13:58

02:03 PM 14:03

Implement the TwentyFourHourWatch class by overriding a *single* method, toString(), from the Watch class.

A template for the class is given for you on the next page.

Note: the subclass does not need any additional properties or methods, and you do not need to override any other methods!

class TwentyFourHourWatch extends Watch {

public TwentyFourHourWatch(int hour, int min) {

super(hour, min);

}

public String toString() {

// \*\*\*\*\*\*\* FILL IN THIS METHOD!!!

int tmpHour = hour;

if (!isAM) tmpHour += 12;

String result = pad(tmpHour)+":"+pad(min)+" ";

return result;

// \*\*\*\*\*\*\* DONE!!!

}

}

[10 marks]

3. Consider the following brief:

*A library system needs to keep track of books and library patrons. Books have a title, author, publisher and publication date. Library patrons have a name, address and patron ID number. All library employees are automatically patrons, and have a library branch that they work at. Patrons can check out books from the library (up to a maximum of ten books), and the system needs to keep track of which patron has checked out which book. Every time a book is issued, an issuing and a due date should be stored.*

Draw a class diagram on this page reflecting this situation as accurately as possible. Your diagram should have

* class names
* property names
* association and inheritance relationships
* multiplicities

Types for the properties, and methods for the classes, are *not* required.

A:



[10 marks]

4. Consider the following collection of classes:

abstract class Animal {

public String name="animal name";

abstract public void speak();

}

abstract class Canine extends Animal {

}

abstract class Feline extends Animal {

}

class Labrador extends Canine {

public String job="unknown";

public void speak() {

println(name+": Woof!");

}

}

class Chihuahua extends Canine {

public void speak() {

println(name+": Squeak");

}

}

class Lion extends Feline {

public void speak() {

println(name+": Roar!");

}

public void attack(){

println(name+" attacks!");

}

}

class Balinese extends Feline {

public void speak() {

println(name+": Meow");

}

}

(a) Translate this code into a UML diagram as accurately as you can.

[5 marks]

(b) Write down the output (or state that there will be an error) of each of these code fragments. If there is a compilation error, circle the line that would cause the error and explain why the error occurs.

Animal animal = new Balinese();

animal.name="Frank";

animal.speak();

Outcome:

“Frank: Meow”

[1 mark]

Balinese animal = new Animal();

animal.name="Frank";

animal.speak();

Outcome:

Error on first line because Animal is not a Balinese and/or Animal is abstract and cannot be instantiated.

[1 mark]

Animal animal;

Lion lion = new Lion();

lion.name = "Eve";

animal = lion;

animal.speak();

Outcome:

“Eve: Roar!”

[1 mark]

Animal animal;

Canine myDog = new Labrador();

myDog.name = "Eve";

myDog.job = "Tracker";

println("My dog’s name is"+myDog.name);

println("My dog’s job is"+myDog.job);

Outcome:

Error on line 4 because the myDog is a reference of type Canine, which does not have a job property.

[2 mark]

5.

6.

7.

8.