CE320-6-AU

UNIVERSITY OF ESSEX

Undergraduate Examinations 2013

LARGE SCALE SOFTWARE SYSTEMS AND EXTREME PROGRAMMING

Time allowed: **TWO** hours

Candidates must answer **ALL** questions**.**

The paper consists of **FOUR** questions.

The questions are **NOT** of equal weight.

The percentages shown in brackets provide an indication of the proportion of the total marks for the **PAPER** which will be allocated.

Please do not leave your seat unless you are given permission by an invigilator.
Do not communicate in any way with any other candidate in the examination room.
Do not open the question paper until told to do so.
All answers must be written in the answer book(s) provided.
All rough work must be written in the answer book(s) provided. A line should be drawn through any rough work to indicate to the examiner that it is not part of the work to be marked.
At the end of the examination, remain seated until your answer book(s) have been collected and you have been told you may leave.

Candidates must answer ALL questions

Question 1

Extreme programming (XP) and other forms of agile development have radically changed the way software is developed. Explain what are the key issues with more traditional forms of software development and describe how XP overcomes these issues. [25%]

Question 2

Unit testing is one of the key practices in XP. Explain why it is so useful and how it is related to other XP practices. [20%]

Question 3

You are part of an extreme programming team that has been asked to build the control software for a new type of vending machine. When fully developed this will be able to sell snacks, hot soup, hot drinks, ice creams, soft drinks, cigarettes, alcoholic drinks, blank DVDs and memory sticks. The machine has a touch screen which will allow interactions with its customers. Some products should not be sold to underage customers.

You are asked to sketch a release plan for the software that will control the machine. Given the specific nature of the project you can act both as a programmer and as a client (the company making the vending machine). The hardware for the fundamental functions of the machine (e.g., making hot drinks) will be provided to you.

In your release plan please specify:

(a) A list of the features identified for the release. If you require further special hardware to implement a feature, indicate so in the plan. [15%]

(b) A priority for each item in terms of value for the client (the company making the vending machine) using a point system where 3 = very valuable, 2 = valuable, 1 = optional. [5%]

(c) A list of the features that should be implemented in the first iteration. [5%]

Question 4

Clean code is an essential element of extreme programming. Someone has developed a simple car race simulator. The simulator class allows the selection of a particular circuit for a race, it lets cars register for the race, and then simulates the race (by progressively making the cars advance along the track by small random amounts). At the end of the race it also declares the winner.

The output of the simulator is purely textual. Something like the following:

```
Welcome to Monza
The race is about to start
5
4
3
2
1
Go!!
Ferrari --> 0.033659663137   McLaren --> 0.096624565228   Mercedes --> 0.0499968453216
Ferrari --> 0.122809172195   McLaren --> 0.101325361906   Mercedes --> 0.0994164776461
Ferrari --> 0.203239123046   McLaren --> 0.121865548628   Mercedes --> 0.105377357465
...
Ferrari --> 17.9656969015   McLaren --> 17.641724238   Mercedes --> 17.2805364256
Ferrari --> 18.0321602779   McLaren --> 17.6717052886   Mercedes --> 17.2816959402
And the winner is....
** Ferrari **
```

Below you will find two implementations of the simulator class: one in Java and one in Python. (The simulator uses two further classes, Car and Circuit, the implementation of which is omitted.)

Choose one implementation, study it, and the list the places (use line numbers) and ways in which the code is not clean code. [30%]

Question 4 ***continues...***

```java
package exam.ce320.essex.ac.uk;

import java.util.ArrayList;
import java.util.Random;

/**
 * This is a super important class that does something extremely useful so
 * we must remember who the author is:
 *
 * @author rpoli
 *
 */

public class RaceManager {
    private          ArrayList<Car>        m_carDatabase       = new ArrayList<Car>();
    private          Circuit               m_crct;
    private          Random                m_generator;
    private          ArrayList<String>     myLaundry           = new ArrayList<String>();

    // here we print the car positions, by doing a for loop and then invoking the
    // method printPosition for each car. Finally we print a newline
    private void printCarPositions() {
        for (Car car: m_carDatabase ) {
            car.printPosition();
        }
        System.out.println();
    } // end of printCarPositions

    private          double                l; // this is the number of laps
    private          boolean               winner;

    private void do_a_bang_bang_bang() {
        for ( int t = 5; t >= 1; t--) {
            System.out.println(t);
            zzz(1000);
        }
        System.out.println("Go!!");
    } // end of do_a_bang_bang_bang

    private void zzz(int millis) {
        try {
            Thread.sleep(millis);
        } catch (InterruptedException e) {
            Thread.currentThread().interrupt();
        }
    }

```

```
48    RaceManager(Circuit     circuit,int     laps) {
49        this.m_crct=circuit;
50        this.m_generator=new Random();
51        this.l= laps*circuit.         getMilesPerLap();
52        this.winner               = false;
53    }
54
55    public void registerCarForRace(Car car) {
56        m_carDatabase.add(car);
57
58    }
59    public void run(){
60        for (Car  car : m_carDatabase ) {
61
62
63            car.resetPosition();
64        }
65        greetSpectators();
66        do_a_bang_bang_bang();
67        moveThem();
68        System.out.println("And the winner is....");
69        zzz(2000);
70        System.out.println("** " + findWinner() + " **");
71    }
72
73    private void moveThem() {
74        while (!winner){
75            for (Car  car : m_carDatabase ) {
76                car.incrementPosition(0.1 * m_generator.nextDouble());
77                if (car.getPosition() > l )
78                    winner = true;
79            }
80            printCarPositions();
81            zzz(1);
82        }
83    }
84
85    private String findWinner() {
86        String theWinner = "";
87        double maxPosition = 0.0;
88        for (Car car: m_carDatabase ) {
89            if (car.getPosition()>maxPosition) {
90                maxPosition = car.getPosition();
91                theWinner = car.getName();
92            }
93        }
94        return theWinner;
```

```
95      }
96
97      private void greetSpectators() {
98          System.out.println("Welcome to " + m_crct.getName());
99          zzz(1000);
100         System.out.println("The race is about to start");
101         zzz(2000);
102     }
103
104     public static void main(String [] args) {
105         Circuit c = new Circuit("Monza", 3.6 );
106         RaceManager r = new RaceManager(c, 60);
107         r.registerCarForRace(new Car("Ferrari"));
108         r.registerCarForRace(new Car("McLaren"));
109         r.registerCarForRace(new Car("Mercedes"));
110         r.run();
111     }
112
113     public ArrayList<String> getMyLaundry() {
114         return myLaundry;
115     }
116
117     public void setMyLaundry(ArrayList<String> myLaundry) {
118         this.myLaundry = myLaundry;
119     }
120
121     public void addToMyLaundry(String item ) {
122         myLaundry.add(item);
123     }
124 }
```

```
1  from car import Car
2  from circuit import Circuit
3  from random import random
4
5  # This is a super important class that does something extremely useful so
6  # we must remember who the author is:
7  #
8  # Author: rpoli
9
10 class RaceManager:
11
12     # here we print the car positions, by doing a for loop and then invoking the
13     # method printPosition for each car. Finally we print a newline
14     def printCarPositions(self):
15         for car in self.m_carDatabase:
16             car.printPosition()
17         print
18     # end of printCarPositions
19
20     def do_a_bang_bang_bang(self):
21         for t in range(5,0,-1):
22             print t
23             self.zzz(1000)
24
25         print "Go!!"
26     # end of do_a_bang_bang_bang
27
28     def zzz(self, millis):
29         from time import sleep
30         sleep(millis/1000.0)
31
32     def __init__(self, circuit, laps):
33         self.m_carDatabase = []
34         self.crct = circuit
35
36         self.totalMilesForRace = laps * circuit.getMilesPerLap()
37         self.winner = False
38         self.myLaundry = []
39
40     def registerCarForRace(self, car):
41         self.m_carDatabase.append(car)
42     def run(self):
43         for car in self.m_carDatabase:
44
45
46             car.resetPosition()
47         self.greetSpectators()
48         self.do_a_bang_bang_bang()
49         self.moveThem()
50         print "And the winner is...."
```

```
        self.zzz(2000)
        print "** " + self.findWinner() + " **"

    def moveThem(self):
        while not self.winner:
            for car in self.m_carDatabase:
                car.incrementPosition(0.1 * random())
                if car.getPosition() > self.totalMilesForRace:
                    self.winner = True
            self.printCarPositions()
            self.zzz(1)

    def findWinner(self):
        theWinner = ""
        maxPosition = 0.0
        for car in self.m_carDatabase:
            if car.getPosition()>maxPosition:
                maxPosition = car.getPosition()
                theWinner = car.getName()
        return theWinner

    def greetSpectators(self):
        print "Welcome to " + self.crct.getName()
        self.zzz(1000)
        print "The race is about to start"
        self.zzz(2000)

    def getMyLaundry(self):
        return self.myLaundry

    def setMyLaundry(self, myLaundry):
        self.myLaundry = myLaundry

    def addToMyLaundry(self,item):
        self.myLaundry.append(item)

if __name__ == '__main__':
    c = Circuit("Monza", 3.6 )
    r = RaceManager(c, 5)
    r.registerCarForRace(Car("Ferrari"))
    r.registerCarForRace(Car("McLaren"))
    r.registerCarForRace(Car("Mercedes"))
    r.run()
```