CE320-6-AU

UNIVERSITY OF ESSEX

Undergraduate Examinations 2019

LARGE SCALE SOFTWARE SYSTEMS AND EXTREME PROGRAMMING

Time allowed: **TWO** hours

Candidates are permitted to bring into the Examination Room:
Calculator – Casio FX-83GT Plus or Casio FX-85 GT Plus ONLY

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.

Question 1

Refactoring is an important practice in extreme programming (XP):

(a) What is refactoring? [5%]

(b) Why is refactoring easier/safer in XP than in traditional project development? [10%]

(c) Why does XP insist on refactoring regularly? [10%]

Question 2

Unit testing is one of the key practices in XP. Compare and contrast manual unit testing and junit/pyunit style of unit testing. [15%]

Question 3

You are part of an extreme programming team that has been asked to build a commercial Web service that will act as an automated letting agent.

You are asked to sketch a release plan for the software. Given the specific nature of the project you can act both as a programmer and as a client.

The system will have three types of users: 1) landlords, 2) current tenants and 3) new potential tenants. Examples of tasks that the system should simplify include: advertising properties, arranging viewings, paying deposits, organising inspection visits, reporting faults, etc.

In your release plan please specify:

(a) A list of the features and stories identified for the release. [20%]

(b) The value of each story for the customers using a point system where 3 = very valuable, 2 = valuable, 1 = optional. [5%]

(c) An estimate of the programming effort required by each story based on the following scale:

E = Easy (takes 0 to 2 working days to the team)

M = Medium (takes 3 to 5 working days to the team)

D = Difficult (2 weeks)

X = eXtremely difficult (3 or more weeks) [5%]

Question 4

An XP team has started developing the software for a TV game show. In the show contestants are given the choice of three doors. Behind one door is a car; behind the others, a goat and a bottle of wine. Clearly, the car is more valuable than the goat, and the goat is more valuable than the bottle of wine. The contestant picks a door, say No 1, which is then opened to reveal what's behind it. If the car is behind that door, the contestant wins it. Otherwise, the computer (which acts as the host) offers the option of picking another door. If the contestant accepts it, then whatever is behind that door is the final prize.

Presently, the software implements the basic functionality of the game. Here is an example of a run of the system (text in boldface has been typed in by a contestant):

```
Please, select door
1
Behind door 1 is a...wine bottle
Pick another door? (y/n)
y
Please, select door
3
Behind door 3 is a...goat
You won it!!!
```

At present the implementation is very simple. It includes one class, **ThreeDoorGame,** with approximately 10 methods. Below you can find two implementations, one in Java and one in Python.

Choose one implementation (Python or Java), study it, then indicate in which ways the code *does not* respect the practices of XP (ie, it is not clean) and *what should be done* to rectify these problems, referring to line numbers in the code listings. Please consider:

(a) names, [7%]

(b) methods, [10%]

(c) comments, and [5%]

(d) formatting. [8%]

Java Version

```java
import java.util.Random;
import java.util.Scanner;

public class ThreeDoorGame {
    // This is a three door game
    String[] doorPrizes      =     new String[]{"car", "wine bottle", "goat"};
    private Random random    =     new Random();
    private Scanner input    =     new Scanner(System.in);

    ThreeDoorGame() {
        randomizeDoorPrizes();
    }

    boolean userWantsToPickAnotherDoor() {
        System.out.println("Pick another door? (y/n)");
        return input.next().equals("y");
    }

    void startTheMerryGoRound() {
        int firstDoor = openDoor();
        if (doorPrizes[firstDoor-1].equals("car")) // checking if top prize
            System.out.println("You won the car!!!");
        else
            if (userWantsToPickAnotherDoor())
                pickAnotherDoorAndSeeWhatWon(firstDoor);
            else
                System.out.println("You won it!!!");
     }

    private void pickAnotherDoorAndSeeWhatWon(int firstDoor) {
        int secondDoor = openDoorDisallowing(firstDoor);
        if (doorPrizes[secondDoor-1].equals("car"))
            System.out.println("You won the car!!!");
        else
            System.out.println("You won it!!!");
    }

    int openDoor() {
        int door = getUserToChooseDoor();
        peekaboo(door);
        return door;
    }

    int openDoorDisallowing(int door) {
        int differentDoor = door;
        while (differentDoor == door)
            differentDoor = getUserToChooseDoor();
        peekaboo(differentDoor);
        return differentDoor;
    }

    void peekaboo(int firstDoor) {
        System.out.println("Behind door "+ firstDoor + " is a..." + doorPrizes[firstDoor-1])
    }

    int getUserToChooseDoor() {
        System.out.println("Please, select door");
        return input.nextInt();
    }

    void randomizeDoorPrizes() {
        for (int door = 0; door < doorPrizes.length; door++) {
            int randomDoor = random.nextInt(doorPrizes.length);
            String prize = doorPrizes[door];
            doorPrizes[door] = doorPrizes[randomDoor];
            doorPrizes[randomDoor] = prize;
        }
    }

    public static void main(String[] args) {
        ThreeDoorGame game = new ThreeDoorGame();
        game.startTheMerryGoRound();
    }
}
```

Python Version

```python
import random

class ThreeDoorGame:
    # This is a three door game
    door_prizes        =     ["car", "wine bottle", "goat"]
    get_keyboard_input =     input

    def __init__(self):
        self.randomize_door_prizes()

    def user_wants_to_pick_another_door(self):
        print("Pick another door? (y/n)")
        return self.get_keyboard_input() == "y"

    def start_the_merry_go_round(self):
        first_door = self.open_door()
        if self.door_prizes[first_door-1] == "car": # checking if top prize
            print("You won the car!!!")
        else:
            if self.user_wants_to_pick_another_door():
                self.pick_another_door_and_see_what_won(first_door)
            else:
                print("You won it!!!")

    def pick_another_door_and_see_what_won(self, first_door):
        second_door = self.open_door_disallowing(first_door)
        if self.door_prizes[second_door-1] == "car":
            print("You won the car!!!")
        else:
            print("You won it!!!")

    def open_door(self):
        door = self.get_user_to_choose_door()
        self.peekaboo(door)
        return door

    def open_door_disallowing(self, door):
        different_door = door
        while different_door == door:
            different_door = self.get_user_to_choose_door()
        self.peekaboo(different_door)
        return different_door

    def peekaboo(self, door):
        print("Behind door %s is a...%s" % (door, self.door_prizes[door-1]))

    def get_user_to_choose_door(self):
        print("Please, select door")
        return int(self.get_keyboard_input())

    def randomize_door_prizes(self):
        for door in range(len(self.door_prizes)):
            random_door = random.randrange(len(self.door_prizes))
            prize = self.door_prizes[door]
            self.door_prizes[door] = self.door_prizes[random_door]
            self.door_prizes[random_door] = prize

if __name__ == "__main__":
    game = ThreeDoorGame()
    game.start_the_merry_go_round()
```

END OF EXAM PAPER CE320-6-AU