CE320-6-AU

UNIVERSITY OF ESSEX

Undergraduate Examinations 2017

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.

Question 1

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

(a) What is refactoring? [7%]

(b) Why is it important for XP? [13%]

Question 2

Extreme Programming (XP) addresses particularly well some issues that have traditionally affected software development. Describe how XP addresses each of the following issues:

(a) Staff turnover. [9%]

(b) Schedule slips [8%]

(c) High defect rate and accumulation of problems [8%]

Question 3

You are part of an extreme programming team that has been asked to build some software that behaves similarly to a screensaver for the computer but actually rather than showing nice pictures it plays *soundscapes*. A soundscape is essentially a set of continuously (programmatically) generated sounds which are mixtures of natural sounds (such as bird singing sounds, rain and water noises, wind, waves on a beach, etc.) and are therefore pleasing to the ear and relaxing.

The program will monitor the computer and the environment around it to decide when it is appropriate to start the soundsaver and when it should be turned off.

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.

In your release plan please specify:

(a) A list the features and stories identified for the release. [15%]

(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

Your XP team has started developing a number-guessing game.

In the software the computer generates a random target number within a given range and a user needs to guess the number in the smallest number of tries possible. After every guess, the computer tells the user where the guess is smaller or bigger than the target number, until the guess matches the target number, in which case the program terminates.

Here is an example of a run of the system (text in boldface has been typed in by a user):

```
Enter your guess: 400
smaller
Enter your guess: 800
bigger
Enter your guess: 700
smaller
Enter your guess: 750
smaller
Enter your guess: 780
smaller
Enter your guess: 790
smaller
Enter your guess: 792
same
```

At present the implementation is very simple. It includes two classes, **Main** and **NumberGuessingGame** in the **Java** implementation, and one class, **NumberGuessingGame** in the file **game.py** and one module, **main.py**, in the **Python** implementation. Below you can find two implementations. Note: the team has also developed unit tests, but these are not important for this exam question and so are omitted.

Choose one implementation (Python or Java), study it, then indicate in which ways the code *does not* respect the practices of XP (i.e., it is not clean), referring to line numbers and file/class names in the code listings. Please consider:

(a) names [7%]

(b) methods/functions [8%]

(c) comments [3%]

(d) formatting [5%]

(e) objects/classes [7%]

Java Version

FILE: NumberGuessingGame.java

```java
import java.util.Random;

public class NumberGuessingGame {
    private int number;

    NumberGuessingGame(int limit) {
        Random generator = new Random();
        number = generator.nextInt(limit);
    }

    public int getNumber() {
        return number;
    }

    public String guess(int g) {
        if ( g > number )
            return "bigger";
        else if ( g < number )
            return "smaller";
        else
            return "same";
    }

    protected void setNumber(int number) {
        this.number = number;
    }
}
```

FILE: Main.java

```java
import java.util.Scanner;

public class Main {
    public static final int LIMIT = 1000;

    static public void main(String args[]) {
        NumberGuessingGame game = new NumberGuessingGame(LIMIT);
        String result = getAndEvalGuess(game);
        while (result != "same") {
            result = getAndEvalGuess(game);
        }
    }

    private static String getAndEvalGuess(NumberGuessingGame game) {
        int g = getGuess();
        String result = game.guess(g);
        System.out.println(result);
        return result;
    }

    private static int getGuess() {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter your guess: ");
        return Integer.parseInt(scanner.nextLine());
    }
}
```

Python Version

FILE: game.py

```
1  from random import randint
2
3
4  class NumberGuessingGame(object):
5      def __init__(self, limit):
6          self.number = randint(0, limit - 1)
7
8      def get_number(self):
9          return self.number
10
11     def guess(self, g):
12         if g > self.number:
13             return "bigger"
14         elif g < self.number:
15             return "smaller"
16         else:
17             return "same"
18
19     def set_number(self, number):
20         self.number = number
```

FILE: main.py

```
1   from game import NumberGuessingGame
2
3   LIMIT = 1000
4
5   def main():
6       game = NumberGuessingGame(LIMIT)
7       result = get_and_eval_guess(game)
8       while result != "same":
9           result = get_and_eval_guess(game)
10
11
12  def get_and_eval_guess(game):
13      g = get_guess()
14      result = game.guess(g)
15      print(result)
16      return result
17
18
19  def get_guess():
20      return int(input("Enter your guess: "))
21
22  if __name__ == "__main__":
23      main()
```