TECHNISCHE UNIVERSITEIT EINDHOVEN

Examination 2IP90
4 November 2014, 9:00–12:00

This exam consists of 3 compound questions on 4 pages.

- Put your name, number, and the date of today at the top of *every Java file* you submit.
- Add comments to your code where clarification is needed.
- Before you submit your solutions, check that you have included all the files you want to submit and that you have saved them.
- Submit your solutions as compilable `.java` files in the folder `2IP91submitfolder` on your desktop (non-Windows users choose a name themselves).
- Do not use named packages for your code. Use only letters, digits, and underscores in your file names.
- **Before you leave the exam, report to a supervisor to verify that your work has been submitted.**
- You are allowed to consult on your laptop the course material (lecture slides, reader, programs you have made during the course) and the Java API.
- Use of the internet or other means of communication is *not* allowed during the examination.

1 Miscellaneous *(35 pt)*

Submit your answers to these questions in a file `Miscellaneous.java`.

For answers that are not Java code but text, such as question 1.2 below, insert the text as comment (between `/*` and `*/`. Submit a compilable Java file.

Precede every answer with the number of the question in comment, e.g.,

```
/*
 * 1.1
 */
```

answer...

(7) 1. Write a method with header `int possum( int[] numbers)` that adds up all the positive numbers in the array `numbers`. For example, if `a` contains the numbers 1, 2, 2, -3, 5, the call `possum(a)` will return 10.

(7) 2. Describe in words what the following method returns

```
boolean m(int[] numbers) {
    boolean result = false;
    for (int i = 0; i < numbers.length; i++) {
        if (numbers[i] == 0) {
            result = true;
        }
    }
    return result;
}
```

(7) 3. In the method above, replace the body of the for-loop with

```
result = numbers[i] == 0;
```

What does the method return now?

(7) 4. What's wrong with the following code? Repair it. (That there is no main method is not considered wrong.)

```
class Square {
    double size;

    void setSize( double s ) {
        size = s;
    }

    double getArea() {
        return size * size;
    }
}

class SquareDemo() {
    void demo() {
        Square mySquare;
```

```
                mySquare.setSize( 5 );
                System.out.println( mySquare.getArea() );
        }
    }
```

(7) 5. Consider the following program.

```
class Course {
    String title;
    int[] grades;

    void printGrades( int threshold ) {
        int grade;
        for (int i=0; i<grades.length; i++) {
            grade = grades[i];
            if ( grade >= threshold ) {
                System.out.print( grade + " " );
            }
            System.out.println();
        }
    }
}
```

Mention all the local variables (and the local variables only) in this program.

2 Sevens *(30 pt)*

Submit your answers to these questions in a file named `Sevens.java` .

(10) 1. Write a method `int lastDigit(int n)` that returns the last digit of the number `n`. E.g., `lastDigit(73)` returns 3, `lastDigit(5)` returns 5.

Write a method `int chop(int n)` that removes the last digit of the number `n`. E.g., `chop(73)` returns 7, `chop(1234)` returns 123, `chop(5)` returns 5.

Hint: Use the operators `%` and `/` .

(10) 2. Write a method `int sevencount(int n)` that returns the number of 7s that the number `n` contains in decimal notation. E.g., `sevencount(7372)` returns 2.

Write a *recursive* method (no loops) to get full points. A non-recursive method will give you only half of the points, maximally.

(10) 3. The function *drop7* removes all digits 7 from a non-negative integer. E.g., $drop7(12) = 12$, $drop7(77) = 0$, $drop7(7372) = 32$. It is defined on non-negative integers as follows.

$$drop7(n) = \begin{cases} 0 & \text{if } n = 0 \\ drop7(chop(n)) & \text{if } n > 0 \text{ and the last digit of } n \text{ is a 7} \\ 10 \cdot drop7(chop(n)) + d & \text{if } n > 0 \text{ and the last digit of } n \text{ is } d \text{ and is not a 7} \end{cases}$$

Here, *chop* is the function `chop` of question 2.

Write a *recursive* method (no loops) to get full points. A non-recursive method will give you only half of the points, maximally.

3 Ballroom *(35 pt)*

The enclosed program `Ballroom.java` shows a blue ball that falls down. When it reaches the bottom of the window, it disappears. This is intended.

(7) 1. Add another falling blue ball 80 pixels to the right and at the same height.

(7) 2. Add a subclass `RedBall` of `Ball` that is almost like a blue ball, except that it is red. Don't put more code in RedBall than is needed.

   Add a falling red ball (class RedBall) to the scene, starting at the same height as the blue balls, 80 pixels to the left of the first blue ball.

(7) 3. Add a class BlinkingBall that changes every 200 ms from blue to red and vice versa. Add a falling blinking ball to the scene, between the red and the blue ball.

(14) 4. Add a subfolder BouncingBallRoom to your folder and copy the Java file(s) related to Ballroom to this folder. Make modifications for this question in the files in this folder and leave the file(s) made for the previous questions 3.1–3.3 untouched.

   Until now, when the balls reach the bottom of the screen, they disappear. Instead, make them change direction, roughly when they reach the bottom of the screen.

   Several solutions are possible. Explain briefly in comment what your solution is and motivate it.

   Change no more classes than necessary.

Good luck!

Grading: The total number of points achieved is the grade g for this examination. The final grade is the result of the following formula rounded to the nearest integer number: $0.6 \cdot g + 0.4 \cdot h$. Here g is the grade for this exam and h is the average of the 6 highest grades for the homework assignments (an assignment that was not submitted is graded with a 0). Furthermore, the grade g has to be at least 5.0 to pass.
