CS 101 - Algorithms & Programming I

Fall 2024 - Lab 6

Due: Week of November 11, 2024

*Remember the **honor code** for your programming assignments.*
*For all labs, your solutions must conform to the CS101 style **guidelines**!*
All data and results should be stored in variables (or constants where appropriate) with meaningful names.

The objective of this lab is to learn static methods. Remember that analyzing the problems and designing the solutions on a piece of paper *before* starting implementation/coding is always a best practice.

For the methods below you should **not** use any built-in methods that perform the mentioned tasks directly. The mentioned methods should be implemented from scratch.

0. Setup Workspace

Start VSC and open the previously created workspace named `labs_ws`. Now, under the `labs` folder, create a new folder named `lab6`. In this lab, you are to have 2 Java classes/files (under `labs/lab6` folder) as described below. A third Java file containing the revision should go under this folder as well. We expect you to submit a total of 3 files including the revision. Do *not* upload other/previous lab solutions in your submission. The user inputs in the sample runs are shown in blue color.
First, we will design and implement some *static* methods. Then these methods will be used in a program to determine the associated properties of the integers input by the user.

1. Simple Mathematical Tool

a. Greatest Common Divisor and Least Common Multiple

The greatest common divisor of two numbers is the maximum number that is divisible by both numbers. For example, the greatest common divisor of 36 and 100 is 4.
The least common multiple of two numbers is the minimum number which is the multiple of both numbers. For example, the Least Common Multiple of 36 and 45 is 180.
In this part, you will write two static methods that calculate the greatest common divisor and least common multiple.

Implement the methods

- `int greatestCommonDivisor(int, int)`: returns the greatest common divisor of two integers given in the parameter.
- `int leastCommonMultiple(int, int)`: returns the least common multiple of two integers given in the parameter.

b. Relatively Prime Numbers

Relatively prime numbers are integers that do not have a common positive multiple other than 1. In this part, you will write a static method that checks if the given two integers are relatively prime. In this part, you can use the methods you implemented before.
Implement the method:

- `boolean isRelativelyPrime(int, int):` returns true if two integers given in the parameter are relatively prime.

c. Binary Reversal

In this question, you will write a program that gets a positive integer (`int`) and returns the reversed binary representation of the input. You will also implement a helper method that converts a number to its binary representation (represented as a `long` containing 0s and 1s as digits) and use it in the binary reversal methods.

For example, the binary representation of 13 is 1101. Its binary reversal is 1011.

For this purpose, you will implement the methods:

Implement helper method:

- `long intToBinary(int):` should return the binary representation of the input

Use helper method in:

- `long binaryReversal(int):` returns the reversed binary representation of the input

Sample runs:

Enter a decimal number: 26 Binary representation of 26 is 11010 Binary reversal of 26 is 1011	Enter a decimal number: 45 Binary representation of 45 is 101101 Binary reversal of 45 is 101101	Enter a decimal number: 16 Binary representation of 16 is 10000 Binary reversal of 16 is 1

d. Simple Menu for Number Operations

In this part, implement a simple program to use the above methods. Create a new/empty file of your own under the `lab6` folder named `Lab06_Q1.java` with a class with the same name that examines the properties of numbers input by the user. Using the methods you implemented previously inside your main method your program should output the properties of each number.

Sample runs:

Enter the first number: 26 Enter the second number: 62 26 and 62 are not relatively prime. Greatest common divisor of 26 and 62 is 2 Least common multiple of 26 and 62 is 806	Enter the first number: 45 Enter the second number: 11 45 and 11 are relatively prime. Greatest common divisor of 45 and 11 is 1 Least common multiple of 45 and 11 is 495	Enter the first number: 3 Enter the second number: 15 3 and 15 are not relatively prime. Greatest common divisor of 3 and 15 is 3 Least common multiple of 3 and 15 is 15

2. Registration System

In this part, you will implement a simple registration system for university students. Again, you are not allowed to use any built-in methods.

a. Email validation

A valid academic email has the following structure:

`<student_name>.<student_surname>@<university_name>.edu.tr`

In this part, you will implement a method that checks if the given email is a valid academic email. If it is a valid email the method should return `true`, otherwise `false`. Also, if the email is valid, you will print the student name, surname, and university name inside the method (see part c for a sample run).

You will implement a method

- `boolean validateEmail(string)`: This method takes email as a parameter. If the email is valid, it prints the information of name, surname, and university (see part c for a sample run) and returns `true`. Otherwise, it returns `false`.

b. Password Validation

A valid password should be at least 8 characters long. It must include at least one uppercase letter, one lowercase letter, and one digit.

You will implement a method

- `boolean validatePassword(string)`: This method takes the password as a parameter. It returns `true` if the password is valid. Otherwise, it returns `false`.

c. Simple menu for registration operations

In this part, implement a simple menu program. Create a new/empty file of your own under the `lab6` folder named `Lab06_Q2.java` with a class with the same name. Your application should validate data input by the user, using the methods implemented in steps a and b. First, input the email address and check if it is valid using the method you implemented before. If the user enters an invalid email, ask the email again until s/he enters a valid email. After entering a valid email, input the password. As in the email, input the password until the user enters a valid password.

Sample runs:

Enter the email: ali.celik@bilkent.edu.tr Student name: Ali Student Surname: Celik University Name: Bilkent Enter the Password: 12345678 The password is not valid, please enter another password Enter the Password: xyz The password is not valid, please enter another password Enter the Password: A15s1cdf The password is valid Registration is successful	Enter the email: elif.demir-ıtu.edu.tr The email is not valid, please enter a valid email Enter the email: elif@.itu.edu.tr The email is not valid, please enter a valid email Enter the email: elif.demir@hacettepe.edu.tr Student name: Elif Student Surname: Demir University Name: Hacettepe Enter the Password: asdfgh12 The password is not valid, please enter another password Enter the Password: 5A2S2R1ex The password is valid Registration is successful