Lab Test 4

Check Your Lab Test Environment

- Check whether your computer is working well.
- Print “Hello World!” in both IntelliJ IDEA Community version and CLion.
- Please check the path of the *.java files and “main.cpp” for the submission.

Announcement

- Cheating (including googling) is forbidden during the test. There will be a strong penalty if you’re caught cheating.
- You can modify any part of the files if it meets the instructions.

Skeleton Code

- Download skeleton codes for the problem 1. (.java files)
- There is no skeleton code for the problem 2.

Submission

- Zip all .java files for **problem 1** and “main.cpp” for **problem 2** as “**20XX-XXXXX.zip**”.
- Upload the zip file to ETL.

Problem 1 (Java) - NameValidator (1/5)

- Complete the Java classes, `NameValidator`, and several exceptions.
- `NameValidator::validate` method validates the format of the input string.
- `NameValidator::validate` throws a subclass of `NameFormatException`.
- The caller of the `NameValidator::validate` should be able to catch the exceptions in the next slide according to each condition.
- Otherwise, `NameValidator::validate` returns true without throwing any exception.

Problem 1 (Java) - NameValidator (2/5)

Throwing Exception Class	Condition
`WrongCharacterException`	The string contains a character except an alphabet or a space.
`SpaceException`	The string should contains only one space, and the space is not the first or the last character of the string. **Otherwise throw this exception.**
`FirstnameNotCapitalizedException`	The first character of the first name is not an upper character.
`LastnameNotCapitalizedException`	The first character of the last name is not an upper character.

Problem 1 (Java) - NameValidator (3/5)

- There are priorities in the four exceptions in the previous slide as follows:
    a. `WrongCharacterException`
    b. `SpaceException`
    c. `FirstnameNotCapitalizedException`
    d. `LastnameNotCapitalizedException`
- For example, “you ngki% lee” meets the conditions of both `WrongCharacterException` and `SpaceException`, then `WrongCharacterException` is thrown.

Problem 1 (Java) - NameValidator (4/5)

- The caller of `NameValidator::validate` can catch `FirstnameNotCapitalizedException` and `LastnameNotCapitalizedException` as `NotCapitalizedException`.
- The caller of `NameValidator::validate` can catch all previously mentioned exceptions as `NameFormatException`.
- Maybe you just need to add a few words to some of the files. Don’t think this is weird, and don’t ask TA about this.

Problem 1 (Java) - NameValidator (5/5)

- Use `String::toCharArray` to convert a string to a list of character if you need.
- Use `String::charAt` to get the nth character of the string if you need.
- Use helper functions `isAlphabet`, `isUpperAlphaber`, `isLowerAlphaber`, and `hasOnlyOneSpace` in `NameValidator` class if you need.

Problem 2 (C++) - Functions (1/3)

- Write a C++ code “main.cpp” that generates an arbitrary polynomial function and determines if a certain input point is above, on, or below the given function’s graph.
- The first input is the degree n of the polynomial.
- 2nd ~ (n+2)th input are the coefficients of the polynomial terms from largest degree to lowest degree.
- (n+3)th, (n+4)th inputs are the input point (x,y) to be tested.
- 1st input is non-negative integer.
- 2nd ~ (n+4)th inputs are arbitrary integer numbers.
- Assume that the input, output, and any intermediate result in evaluating the polynomial do not go beyond.

Problem 2 (C++) - Functions (2/3)

Input

```
3
2
3
1
3
3
80
```

degree 3

f(x) = $2x^3+3x^2+x+3$

test point (3, 80)

Output

```
The point is below the polynomial.
```

Input

```
2
1
1
0
1
2
```

f(x) = x^2+x

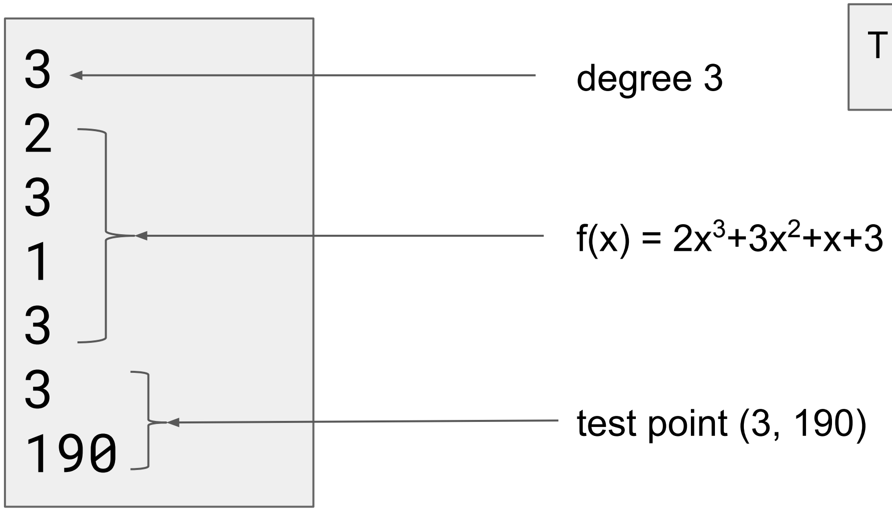
test point (1, 2)

Output

```
The point is on the polynomial.
```

Problem 2 (C++) - Functions (3/3)

Input



```
3
2
3
1
3
3
190
```

degree 3

f(x) = $2x^3+3x^2+x+3$

test point (3, 190)

Output

```
The point is above the polynomial.
```