CPS 2231 2025 Spring
Lab 5: Advanced Classes

Instructor: Dr. Y. Tiffany Tang[1] and Dr. Pinata Winoto[2]
Songlin Shang[3], Cong Ye[4] and Zike Deng[5]

Instruction:

Self-check: In all labs in the future, you can download `check.py` in canvas, which will help you check the output format in labs, to make sure you don't have compile error or logic error before you submit your file. If you didn't finish installing the python environment/JDK installation, you can download the file from canvas named as tutorial.pdf. Please read and follow the tutorial, which teaches you how to install a python environment, for checking your code result. Also, the instructions of all questions also teach you how to use `check.py` to check your code.

Plagiarism: We encourage discussing all labs with your classmate, but **NEVER** copy other's code directly. We will use GradeScope[@] platform, to check the similarity probability, 90% similarity reported by GradeScope will be reviewed by TAs, and will report to your instructor and receive some penalty based on the event.

AI-Generate code issue: We encourage you to use AI tools, to help you compile all labs, **BUT NEVER** directly **copy and paste**. To detect that, we will use different tool to detect AI possibility include:

ChatGPT Code Detector:https://chatgpt.com/c/67bdddec-c728-8010-b1b6-abf3e56b3136

GPTSniffer: https://github.com/MDEGroup/GPTSniffer

Penalty will be given for high AI probability.

Grading: Each lab has the same points, and each test case has the same points. In check.py, you might check your code with given test cases in labs to avoid spelling errors and compile errors. We still have invisible test cases, which test for grading. This might test you to consider all possible positions. All test cases have the same points.

[1] Tiffany Ya Tang, Associate Professor, Department of Computer Science, Wenzhou-Kean University; Email: yatang@kean.edu
[2] Pinata Winoto, Assistant Professor, Department of Computer Science, Wenzhou-Kean University; Email: pwinoto@kean.edu
[3] Songlin Shang, Department of Computer Science, Wenzhou-Kean University; Email: shangs@kean.edu
[4] Cong Ye, Department of Computer Science, Wenzhou-Kean University; Email: yecon@kean.edu
[5] Zike Deng, Department of Computer Science, School of Computing and Data Science, University of Hong Kong; Email: baylordeng@connect.hku.hk

Get Started

Please download Lab5_1.java, Lab5_2.java, Lab5_3.java on canvas.

Lab5_3 is optional.

Before you start the lab, you can create a project on eclipse called Lab5.

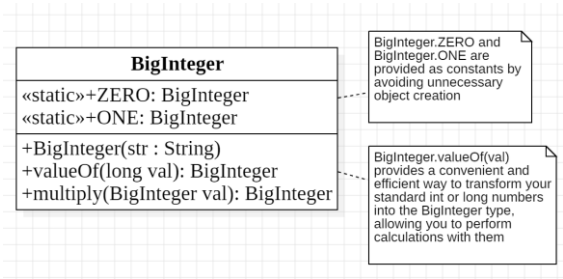
Follow the previous steps in other labs.

The method you may need is as following.

java.lang.Integer

```
-value: int
+MAX_VALUE: int
+MIN_VALUE: int

+Integer(value: int)
+Integer(s: String)
+byteValue(): byte
+shortValue(): short
+intValue(): int
+longValue(): long
+floatValue(): float
+doubleValue(): double
+compareTo(o: Integer): int
+toString(): String
+valueOf(s: String): Integer
+valueOf(s: String, radix: int): Integer
+parseInt(s: String): int
+parseInt(s: String, radix: int): int
```



java.lang.StringBuilder

`+append(data: char[]): StringBuilder`	Appends a `char` array into this string builder.
`+append(data: char[], offset: int, len: int): StringBuilder`	Appends a subarray in `data` into this string builder.
`+append(v: aPrimitiveType): StringBuilder`	Appends a primitive-type value as a string to this builder.
`+append(s: String): StringBuilder`	Appends a string to this string builder.
`+delete(startIndex: int, endIndex: int): StringBuilder`	Deletes characters from `startIndex` to `endIndex-1`.
`+deleteCharAt(index: int): StringBuilder`	Deletes a character at the specified index.
`+insert(index: int, data: char[], offset: int, len: int): StringBuilder`	Inserts a subarray of the data in the array into the builder at the specified index.
`+insert(offset: int, data: char[]): StringBuilder`	Inserts data into this builder at the position offset.
`+insert(offset: int, b: aPrimitiveType): StringBuilder`	Inserts a value converted to a string into this builder.
`+insert(offset: int, s: String): StringBuilder`	Inserts a string into this builder at the position offset.
`+replace(startIndex: int, endIndex: int, s: String): StringBuilder`	Replaces the characters in this builder from `startIndex` to `endIndex - 1` with the specified string.
`+reverse(): StringBuilder`	Reverses the characters in the builder.
`+setCharAt(index: int, ch: char): void`	Sets a new character at the specified index in this builder.

Lab 5_1 Wrapper Class Analysis

Write a method called **observeWrapper** that takes a string as input.

This method should attempt to create an Integer object from the input string.

1. Create the Integer object of both inputs.
2. Print the first Integer object itself.
3. Print the primitive int value obtained by unboxing the Integer object.
4. Print the result of adding 5 to first Input Object.
5. If the second input does not represent a valid integer: Catch the appropriate exception. (eg. Floating-point number, character and string)

Hint: Consider how the **Integer** class handles **String** inputs. To handle exception, you can **directly use the provided structure in template**. Use a try-catch block to handle potential runtime errors by enclosing the risky code within the try block and specifying how to respond to specific exceptions within the catch block.

Example 1
Input:
0
-10
Output:
Integer Object: 0
Primitive int value (after unboxing): 0
Integer Object + 5 = 5
Integer Object (this line might not be reached): -10
Example 2
Input:
999999999
1
Output:
Integer Object: 999999999
Primitive int value (after unboxing): 999999999
Integer Object + 5 = 1000000004
Integer Object (this line might not be reached): 1
Example 3
Input:
abc 123
Output:
Invalid Input.
Example 4
Input:
123 abc
Output:
Integer Object: 123
Primitive int value (after unboxing): 123
Integer Object + 5 = 128
Example 5

Input:
123 1.23
Output:
Integer Object: 123
Primitive int value (after unboxing): 123
Integer Object + 5 = 128

Self-Check:

When you finish coding, you can run following code: `python check.py --lab Lab5_1`

If everything good, you might see following content:

Running test case: Valid input: 0 and -10

********* Running Java program: Lab5_1 *********

☑ Execution succeeded!

☑ Output is correct:
Integer Object: 0
Primitive int value (after unboxing): 0
Integer Object + 5 = 5
Integer Object (this line might not be reached): -10

Running test case: Valid input: large number and 1

********* Running Java program: Lab5_1 *********

☑ Execution succeeded!

☑ Output is correct:
Integer Object: 999999999
Primitive int value (after unboxing): 999999999
Integer Object + 5 = 1000000004
Integer Object (this line might not be reached): 1

Running test case: abc and 123

********* Running Java program: Lab5_1 *********

☑ Execution succeeded!

☑ Output is correct:
Invalid Input.

Running test case: Invalid first input: abc

********* Running Java program: Lab5_1 *********

☑ Execution succeeded!

☑ Output is correct:
Integer Object: 123
Primitive int value (after unboxing): 123
Integer Object + 5 = 128

Running test case: Invalid second input: 1.23 (expecting NumberFormatException)

********* Running Java program: Lab5_1 *********

☑ Execution succeeded!

☑ Output is correct:
Integer Object: 123
Primitive int value (after unboxing): 123
Integer Object + 5 = 128

☑ All checks passed for the specified lab!

Lab 5_2 Precise Calculation of Factorials

Write a method called **calculateFactorial** that calculates the factorial of a given non-negative integer n. Your program must use the **BigInteger** class to ensure accurate calculations for large values of n.

1. Input: The program should take a single non-negative integer n as input from the user.
2. Calculation: Calculate the factorial of n (n!). Recall that n! = n * (n-1) * (n-2) * ... * 2 * 1. By definition, 0! = 1.
3. Output: Print the calculated factorial to the console.

Hint: You'll need to iterate through the given range and reuse the factorial calculation logic.

Example 1
Input:
100
Output:
933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582511852109168640000000000000000000000

Example 2
Input:
-5
Output: Invalid Input.

Example 3
Input:
0
Output: 1
Example 4
Input:
abc
Output: Invalid Input.

Example 5
Input:

Output: Invalid Input.

Self-Check:

When you finish coding, you can run following code:

```
python check.py --lab Lab5_2
```

If everything good, you might see following content:

Running test case: Large input

********* Running Java program: Lab5_2 *********

☑ Execution succeeded!

☑ Output is correct:
933262154439441526816992388562667004907159682643816214685929638952175999932299156089414639761565182862536979208272237582
5118521091686400000000000000000000000000

Running test case: Negative input

********* Running Java program: Lab5_2 *********

☑ Execution succeeded!

☑ Output is correct:
Invalid Input.

Running test case: Zero input

********* Running Java program: Lab5_2 *********

☑ Execution succeeded!

☑ Output is correct:
1

Running test case: Non-numeric input

********* Running Java program: Lab5_2 *********

☑ Execution succeeded!

☑ Output is correct:
Invalid Input.

Running test case: Empty input

********* Running Java program: Lab5_2 *********

☑ Execution succeeded!

☑ Output is correct:
Invalid Input.
☑ All checks passed for the specified lab!

Lab 5_3 Challenge Question (Optional)

Word Reverser

Write a method called **reverseWords** that takes a string representing a sentence as input. This method should use StringBuilder's :

1. Reverse the order of the letters within each individual word in the sentence.
2. Maintain the original order of the words in the sentence.
3. Preserve the original spacing between the words.
4. Return the newly constructed string with the reversed words.

Hint: You'll need to split the sentence into individual words, reverse each word, and then reconstruct the sentence.

Example 1
Input:
hello world
Output:
olleh dlrow

Example 2
Input:
java programming
Output:
avaj gnimmargorp

Example 3
Input:
apple banana cherry
Output:
elppa ananab yrrehc

Example 4
Input:
race car
Output:
ecar rac

Example 5
Input:
hello
Output:
olleh

Self-Check:

When you finish coding, you can run following code: `python check.py --lab Lab5_3`
If everything good, you might see following content:

Running test case: Reverse each word in the sentence

********* Running Java program: Lab5_3 *********

☑ Execution succeeded!

☑ Output is correct:
olleh dlrow

Running test case: Reverse each word in the sentence

********* Running Java program: Lab5_3 *********

☑ Execution succeeded!

☑ Output is correct:
avaj gnimmargorp

Running test case: Reverse each word in the sentence

********* Running Java program: Lab5_3 *********

☑ Execution succeeded!

☑ Output is correct:
elppa ananab yrrehc

Running test case: Reverse each word in the sentence

********* Running Java program: Lab5_3 *********

☑ Execution succeeded!

☑ Output is correct:
ecar rac

Running test case: Reverse a single word

********* Running Java program: Lab5_3 *********

☑ Execution succeeded!

☑ Output is correct:
olleh

☑ All checks passed for the specified lab!

Last Step

Congratulations! You have finished all Labs in Lab5.

Now please run the following code in the end:

```
python check.py --uid 123456
```

(replace 123456 as you student number)

Here is the example operation: (all Lab3 should be replace by Lab5)

It will check all labs again and zip a file. **Since Lab 3_3 is an optional lab,** it will ask you to answer whether you have finish the lab, and you might see following content:

```
Have you completed Lab3_3? (y/n):
```

Input " y",then press " enter" if you finish Lab2_3, otherwise press n.

If everythings good, you might see following information:

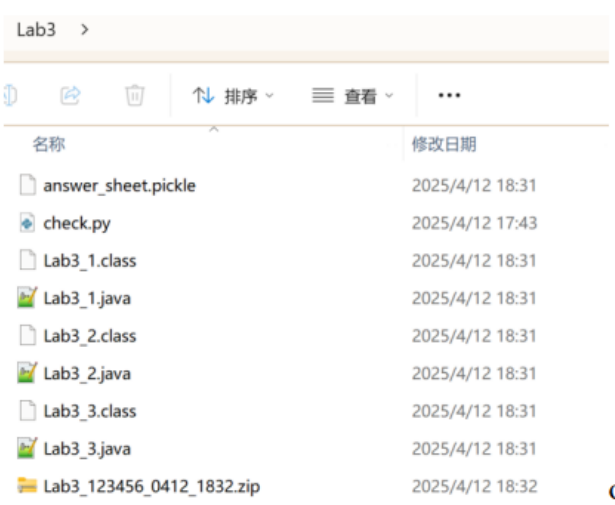
```
Your Student ID is 123456. Is this correct? (y/n):
```

Input " y",then press " enter", you will see following information.

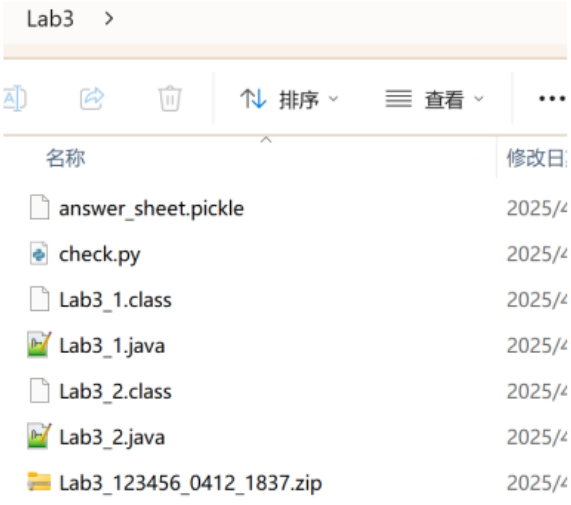
```
********* Zipping Files *********
Zipping Lab3_1.java... Done.
Zipping Lab3_2.java... Done.
Zipping answer_sheet.pickle... Done.
Zipping Lab3_3.java... Done.

Successfully created submission package: Lab3_123456_0412_1832.zip
```

You will see there is a new file under the folder like:



or



Submit this .zip file to canvas!