

COMP2026 Problem Solving Using Object Oriented Programming

Laboratory 4

Part A Discovery Exercises

Task 1: Command-Line Arguments

A Java application can accept any number of arguments from the command line. This allows the user to specify configuration information when the application is launched. When an application is executed using the **java** command, Java passes the command-line arguments that appear after the class name in the java command to the application's **main** method as an array of Strings, conventionally named **args**, in the parameter list of **main**.

The given example **CLArgsPrinter.java** prints each of the command-line arguments on a line by itself:

```
public class CLArgsPrinter {
    public static void main(String[] args) {

        new CLArgsPrinter().runApp(args);

    }

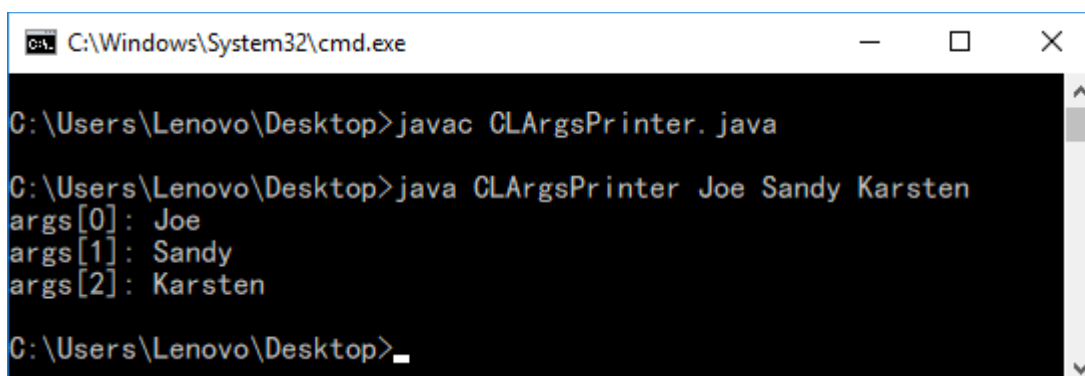
    void runApp(String[] args) {

        for (int i = 0; i < args.length; i++) {
            System.out.println("args[" + i + "]: " + args[i]);
        }

    }

}
```

Compile and run the **CLArgsPrinter.java** in the Command Prompt on Windows:



```
C:\Windows\System32\cmd.exe

C:\Users\Lenovo\Desktop>javac CLArgsPrinter.java

C:\Users\Lenovo\Desktop>java CLArgsPrinter Joe Sandy Karsten
args[0]: Joe
args[1]: Sandy
args[2]: Karsten

C:\Users\Lenovo\Desktop>_
```

The command "**java CLArgsPrinter Joe Sandy Karsten**" passes three arguments, "Joe", "Sandy" and "Karsten" to the application **CLArgsPrinter**. When this command executes, the **main** method receives the three-element array **args** (i.e., **args.length** is 3) in which **args[0]** contains the String "Joe", **args[1]** contains the String "Sandy" and **args[2]** contains the String "Karsten". The programs determine how to use these arguments. In the above example, the **CLArgsPrinter** prints each of the arguments on a line by itself.

While operating upon Strings, there are times when we need to convert a number represented as a String into a number type. The wrapper classes for Java's numeric types have some built in methods to parse a numerical value from a String. For example,

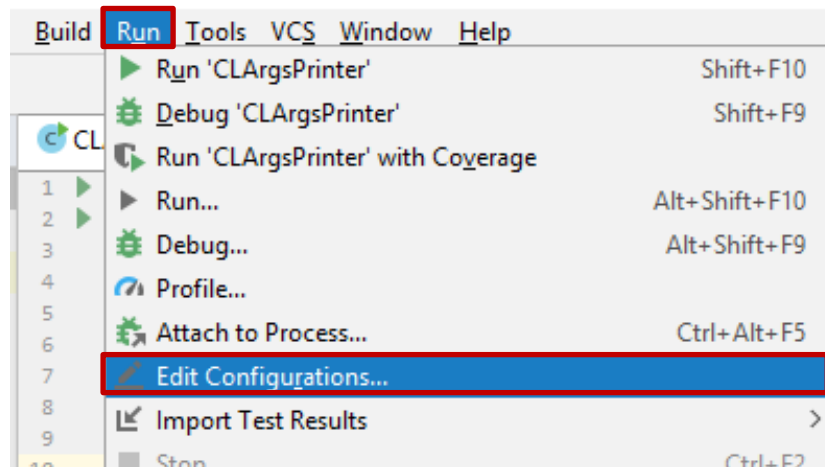
To parse an integer from String:

```
String s = "123";  
int num = Integer.parseInt(s); //convert String "123" into an  
                               //integer 123
```

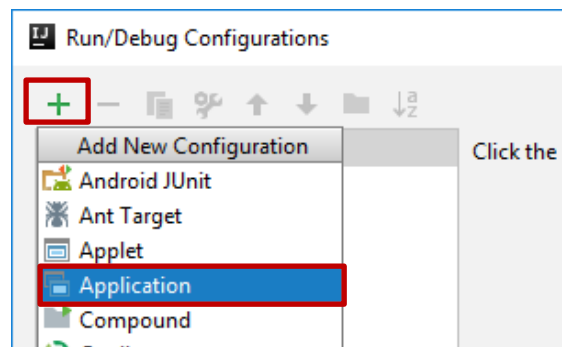
To parse a double from String:

```
String s = "8.45";  
double num = Double.parseDouble(s); //convert String "8.45" into  
                                     //double 8.45
```

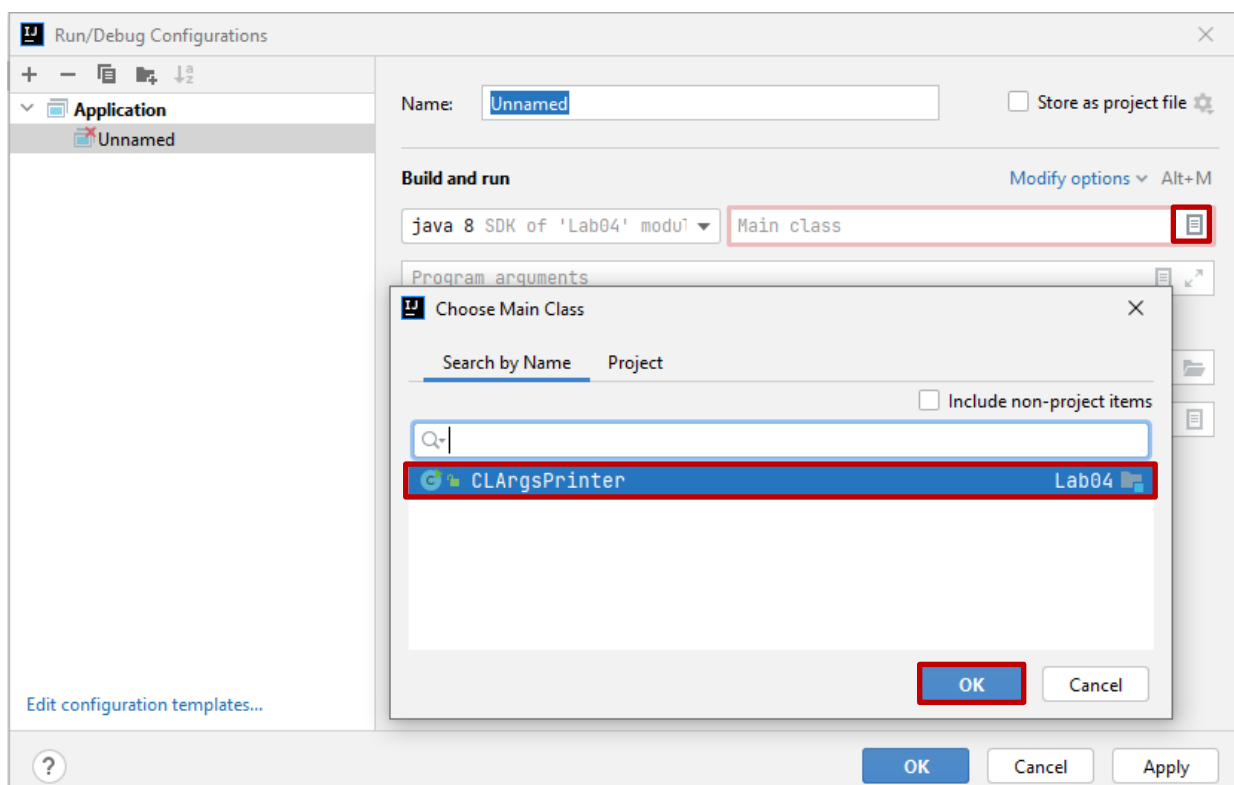
To use command-line arguments in IntelliJ, click **Run → Edit Configurations**.



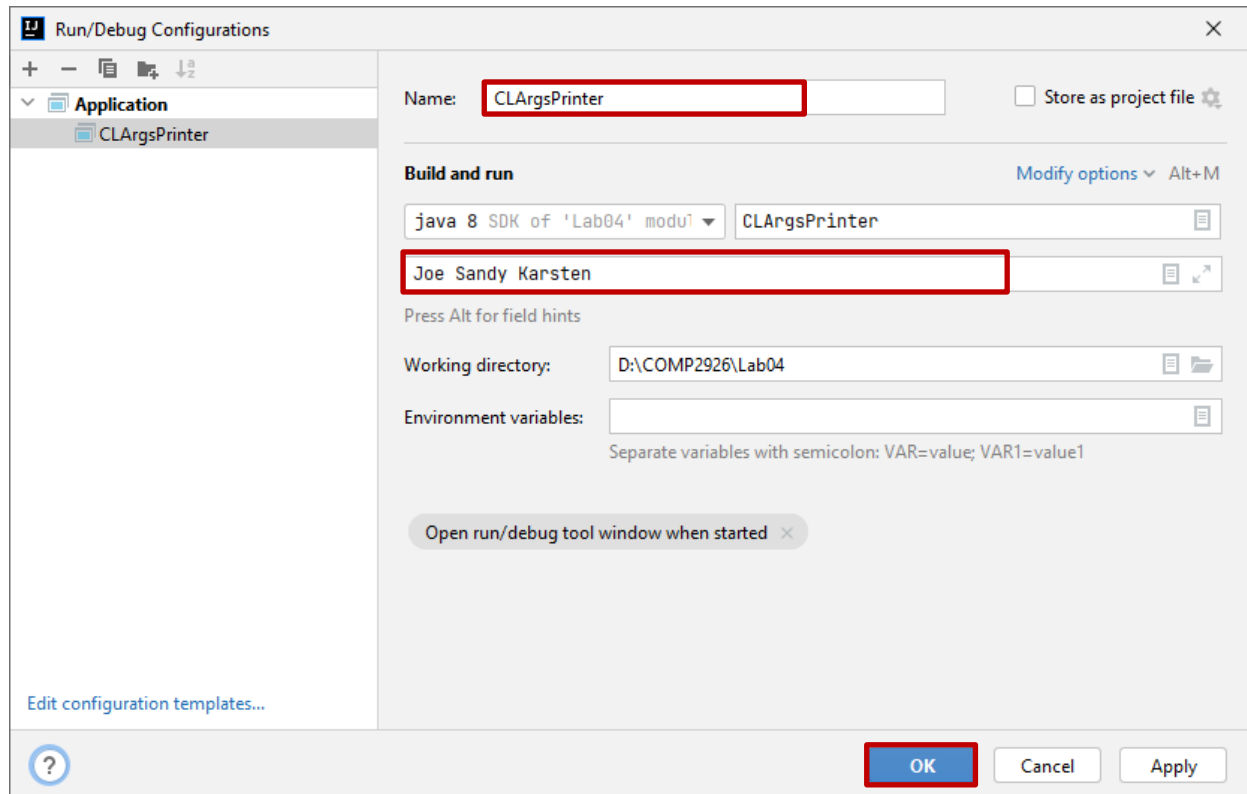
Click the green “+” button and select **Application** to create a new configuration.



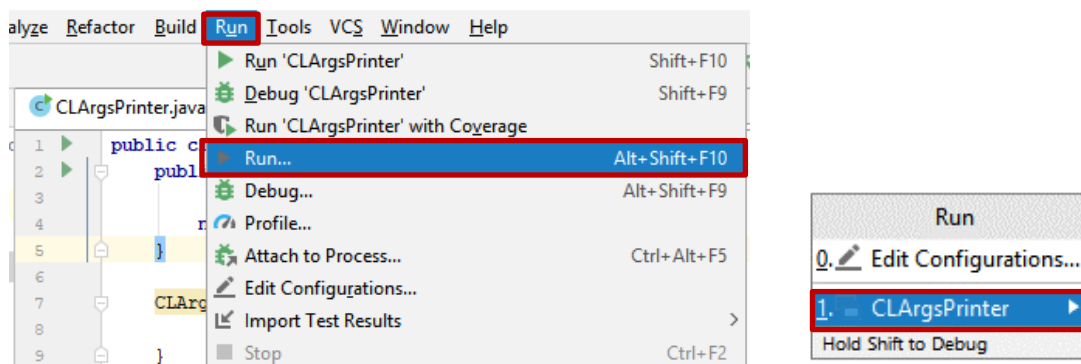
Click the “” button in Main class. Then select **CLArgsPrinter** and click **OK**.



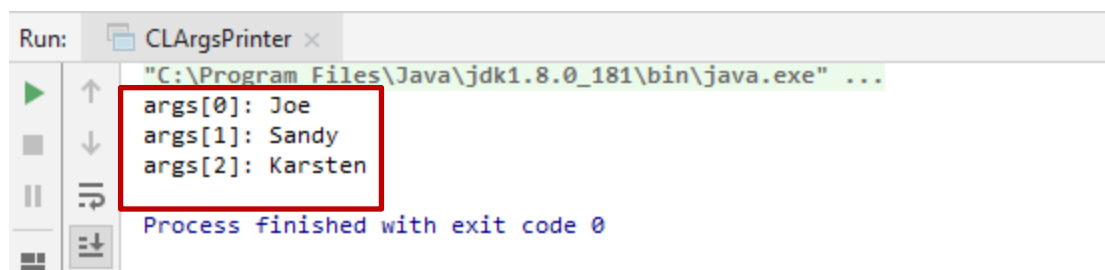
Type in “**CLArgsPrinter**” as the name of the configuration and type the arguments inside the **Program arguments** textbox and then click **OK**.



Run the program as usual. **Run** → **Run...** → **CLArgsPrinter**



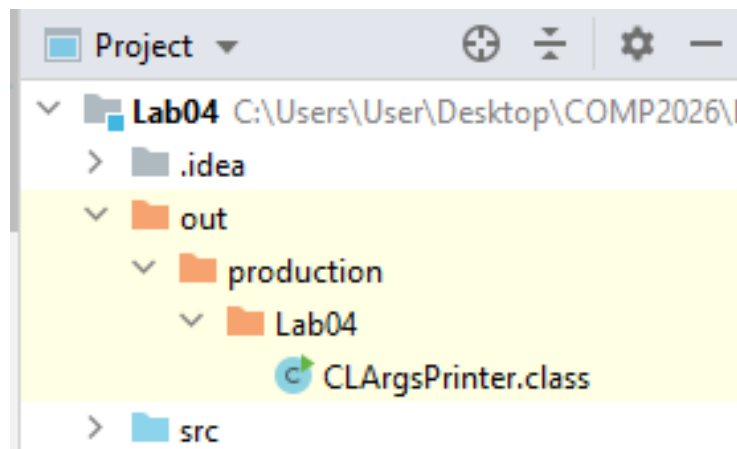
Result:



- a) Compile and execute the given **CLArgsPrinter.java** program with the following program arguments and fill in the following table.

Program arguments	args.length	Values in args
Joe Sandy Karsten	3	args[0]: "Joe" args[1]: "Sandy" args[2]: "Karsten"
1 2 3 4 5		
HelloWorld!		
Chan Tai Man		
"Chan Tai Man"		

After the program is compiled by IntelliJ, the .class file is created inside the "out" folder.



Task 2: Array Manipulation

Given a one-dimensional integer array **x**.

a) Write a statement to print the size of **x**.

b) Write a code fragment that swaps the first and the last element in the array **x**.

c) Write a code fragment that create a new integer array **y** which size is double of the size of **x**.

d) Write a loop to copy all the elements in array **x** to array **y**.

e) The following code fragment intends to print the minimum value in an integer array **a**. Fix all the errors in the code fragment.

```
int[] a = {24, 16, 37, 5, 78};
int min = 0;
for(int i = 0; i < a.length; i++){
    if(a[i] < min){
        min = a[i];
    }
}
System.out.println(min);
```

Task 3: Enhanced for loop

a) Rewrite the given for loop into an enhanced for loop.

Given:

```
int[] a = {2, 16, 7, 5, 8};  
for(int i = 0; i < a.length; i++){  
    System.out.println(a[i]);  
}
```

Answer:

b) Given then following for loop, can you convert it into enhanced for loop? Why or why not?

```
for(int i = 0; i < a.length; i++){  
    a[i] = 0;  
}
```

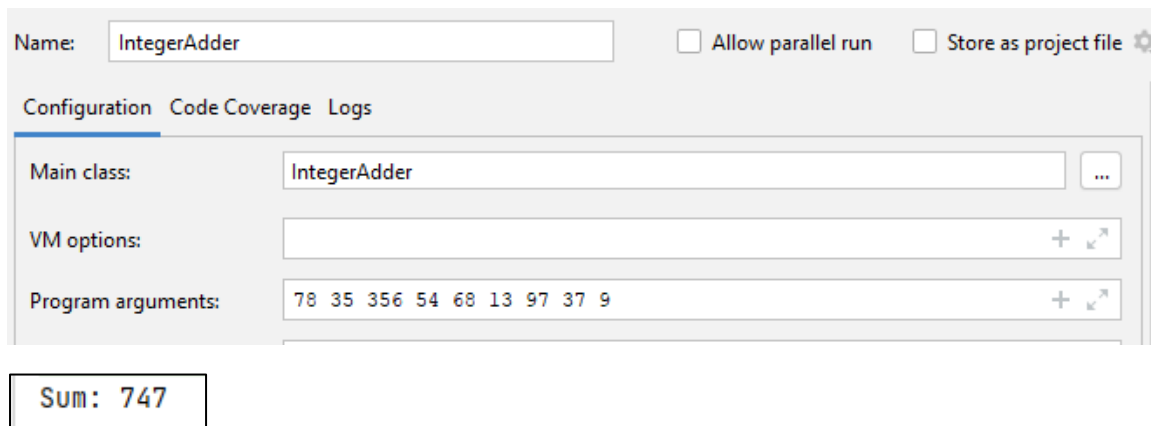
Answer:

Part B Programming Exercises

Task 1: Integer Adder

Write a program called **IntegerAdder** that reads input from command-line arguments and prints the sum of all the command-line arguments. Assume that all command-line arguments are integers represented as Strings.

Sample output in IntelliJ:



Sample output in the Command Prompt on Windows (IntegerAdder.class in Desktop):

```
C:\Users\pikayo\Desktop>java IntegerAdder 78 35 356 54 68 13 97 37 9
Sum: 747
```

Task 2: Finding Character

Write a program called **CharFinder** that reads a line of input as a String and a character value and then prints

- The String, with all the characters that equal the input character value replaced by an underscore
- The number of occurrences of the character value in the String

You are **NOT allowed** to use the String methods except **length()** and **charAt()**.

Sample outputs:

```
Enter a line: Problem Solving Using Object Oriented Programming
Enter a character: i
Problem Solv_ng Us_ng Object Or_ented Programm_ng
No. of occurrences of i: 4
```

```
Enter a line: Computer Science
Enter a character: e
Comput_r Sci_nc_
No. of occurrences of e: 3
```


Task 3: Array Manipulation

Write a program called **ArrayEx** that reads 9 integers and store them into an array. Then prints four lines of output, containing

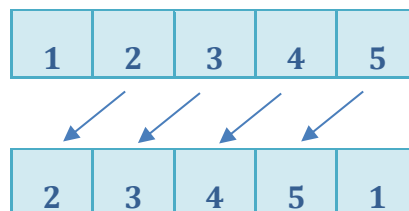
- a) Every element at an even index
- b) Every even element
- c) All elements in reverse order
- d) The alternating sum of all elements. For example, if your program reads input
1 4 8 16 9 7 3 9 10
then it computes
 $1 - 4 + 8 - 16 + 9 - 7 + 3 - 9 + 10 = -5$

Sample output:

```
Enter 9 integers: 1 4 8 16 9 7 3 9 10
Every element at an even index: 1 8 9 3 10
Every even element: 4 8 16 10
All elements in reverse order: 10 9 3 7 9 16 8 4 1
Alternating sum: -5
```

Task 4: Rotate Elements

Complete the given **AryRotation.java** to rotate the elements of the array by one position, moving the initial element to the end of the array, like this:

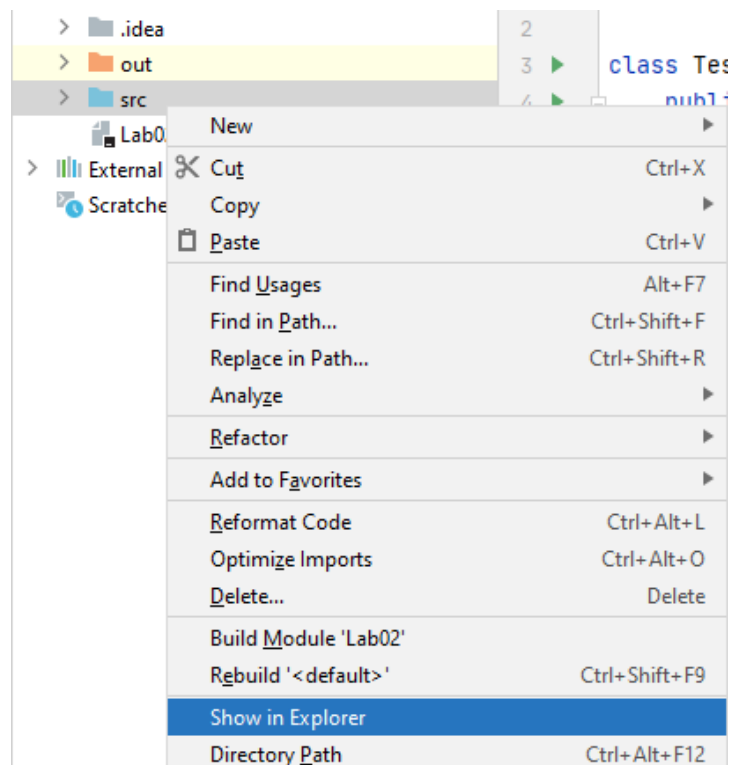


Sample output:

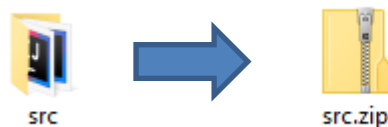
```
|1|2|3|4|5|
|2|3|4|5|1|
```

Part C Submitting Exercises

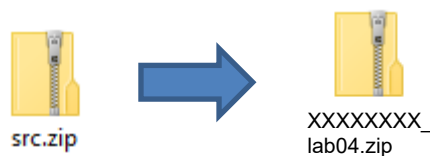
Step 1: Right-click the **src** folder and select **Show in Explorer**



Step 2: Zip the **src** folder into **src.zip**



Step 3: Rename the **src.zip** file to **XXXXXXXX_lab04.zip** where **XXXXXXXX** is your **student id**



Step 4: Submit **XXXXXXXX_lab04.zip** and **XXXXXXXX_lab04.docx** to Moodle.



References

- [1] Bravaco, R., & Simonson, C. (2009). *Java programming: From the ground up*. Dubuque, IA: McGraw-Hill.
- [2] Dean, J., & Dean, R. (2008). *Introduction to programming with Java: A problem solving approach*. Boston: McGraw-Hill.
- [3] Farrell, J. (2012). *Java programming. Boston, MA: Course Technology Cengage Learning*
- [4] Forouzan, B. A., & Gilberg, R. F. (2007). *Computer science: A structured programming approach using C (3rd ed.)*. Boston, MA: Thomson Course Technology.
- [5] Gaddis, T. (2016). *Starting out with Java (6th ed.)*. Pearson.
- [6] Liang, Y. D. (2013). *Introduction to Java programming: Comprehensive version. (8th ed.)*. Pearson.
- [7] Schildt, H. (2006). *Java a beginner's guide*. New York: McGraw Hill.
- [8] Schildt, H., & Skrien, D. J. (2013). *Java programming: A comprehensive introduction*. New York: McGraw-Hill.
- [9] Wu, C. T. (2010). *An introduction to object-oriented programming with Java*. Boston: McGraw Hill Higher Education
- [10] Xavier, C. (2011). *Java programming: A practical approach*. New Delhi: Tata McGraw Hill.
- [11] yet another insignificant Programming Notes. (n.d.). Retrieved from <https://www3.ntu.edu.sg/home/ehchua/programming>
- [12] Zakhour, S., Kannan, S., & Gallardo, R. (2013). *The Java tutorial: A short course on the basics (5th ed.)*.