System Requirements Specification Index

For

Java Skills Evaluation

Version 1.0

IIHT Pvt. Ltd.
fullstack@iiht.com

USE CASE DESCRIPTION

Use Case 1

Take input of a string from the user and convert all alphabets into their respective reflection in either direction of the alphabet set.
Example:
Input-1: welcome
Output-1: dvoxlnv
Input-2: ab9*c
Output-2: zy9*x

Description

1. Take console input of a string, in **main ()** method, from user
2. Pass the string to method **findReflection()** and write the logic in that method to find the reflections of the alphabets only, rest of the characters will remain the same**(Input-2)**.
3. Return the string from **findRelection()** to **main()** and display the resulting string.
4. Template Code of the same is provided at:
   com.iiht.training.reflections.StringAlphabetReflection

Use Case 2

Write a program to arrange the elements of a given array of integers where all negative integers appear before all the positive integers.
Example:
Input-1: 5 (array size)
1 -2 4 -1 6
Output-1: -2 -1 1 4 6
Input-2: 4(array size)
4 -5 6 -3 2
Output-2: -5 -3 6 2

Description

1. Take console input of array size and the array elements, in **main ()** method from the user.
2. Pass the array to method **getNegativeFirst()** and write the logic in that method to arrange all the negative elements in the beginning and positive elements in the last, make that the positive elements should be in the same order.
3. Return the array from **getNegativeFirst()** to **main()** and display the array elements.
4. Template Code of the same is provided at:
   com.iiht.training.negativefirst.ArrayNegativeBeforePositive

EXECUTION STEPS

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to Application menu (Three horizontal lines at left top) ->Terminal ->New Terminal.
3. This editor Auto Saves the code.
4. If you want to exit(logout) and continue the coding later anytime (using Save & Exit option on Assessment Landing Page) then you need to use CTRL+Shift+B-command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.
5. These are time bound assessments the timer would stop if you logout and while logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.
6. Command to run code for use-case 1:
   ```
   mvn compile exec:java -
   Dexec.mainClass="com.iiht.training.negativefirst.ArrayNegativeBeforePositive"
   ```
7. Command to run code for use-case 2:
   ```
   mvn compile exec:java -
   Dexec.mainClass="com.iiht.training.reflections.StringAlphabetReflection"
   ```

8. You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.

CONSIDERATIONS

1. **Mandatory: Run the following command before the final submission**

   mvn test

   (You can run this command multiple times to identify the test case status, and refactor code to make maximum test cases passed before final submission)
2. **Your code will also be evaluated for code quality, naming conventions, readability etc.**
3. **Make sure you do not modify existing class and method names and their signatures, else it would severely affect final result**
4. **Make sure you do not add any new class or methods, else it would severely affect final result**
5. **Make sure you do not modify any test cases, else it would severely affect final result**