Tutorial 6

(String handling)

1. Write a program called Bin2Dec to convert an input binary string into its equivalent decimal number. Your output shall look like:

Enter a Binary string: **1011**

The equivalent decimal number for binary "1011" is 11

Enter a Binary string: **1234**

Error: Invalid Binary String "1234"

1. On your phone keypad, the alphabets are mapped to digits as follows: ABC(2), DEF(3), GHI(4), JKL(5), MNO(6), PQRS(7), TUV(8), WXYZ(9).

Write a program called PhoneKeyPad, which prompts user for a String (case insensitive), and converts to a sequence of Keypad digits. Use a nested-if (or switch-case) in this exercise. Modify your program to use an *array* for table look-up later.

Hints: You can use in.next().toLowerCase() to read a string and convert it to lowercase to reduce your cases.

**Part II**

**Hint:**

**String vs StringBuffer vs StringBuilder**

String is immutable whereas StringBuffer and StringBuider are mutable classes.

StringBuffer is thread safe and synchronized whereas StringBuilder is not, thats why [StringBuilder is more faster than StringBuffer.](http://www.journaldev.com/137/stringbuffer-vs-stringbuilder-benchmarking)

String concat + operator internally uses StringBuffer or StringBuilder class.

For String manipulations in non-multi threaded environment, we should use StringBuilder else use StringBuffer class.

**STRING AND STRING BUFFER:**

**Exercise 1:** Create a class containing a method to create the mirror image of a String. The method should return the two Strings separated with a pipe(|) symbol

|  |  |
| --- | --- |
| Method Name | getImage |
| Method Description | Generate the mirror image of a String and add it to the existing string. |
| Argument | String |
| Return Type | String |
| Logic | Accepts One String  Find the mirror image of the String  Add the two Strings together separated by a pipe(|) symbol. For Example  Input : EARTH  Output : EARTH|HTRAE  Hint: Use StringBuffer API (Ex: For this problem reverse method in Stringbuffer can be used) |

**Exercise 2:** Create a method which accepts a String and replaces all the consonants in the String with the next alphabet.

**Note**: Consonant refers to all alphabets excluding vowels

|  |  |
| --- | --- |
| Method Name | alterString |
| Method Description | Replace consonants |
| Argument | String |
| Return Type | String |
| Logic | Return the String replacing all the consonants with the next character.  For Example :JAVA should be changed as  KAWA |

**Exercise 3:** Create a method which can perform the following operations on two String objects S1 and S2. The output of each operation should be added to an arraylist and the arraylist should be returned.(Assume S2 is of smaller size)

Examples for below statements are shown in the Logic part

1. Character in each alternate index of S1 should be replaced with S2
2. If S2 appears more than once in S1, replace the last occurrence of S2 in S1 with the reverse of S2, else return S1+S2
3. If S2 appears more than once in S1, delete the first occurrence of S2 in S1, else return S1
4. Divide S2 into two halves and add the first half to the beginning of the S1 and second half to the end of S1.

Note: If there are odd number of letters in S2, then add (n/2)+1 letters to the beginning and the remaining letters to the end. (n is the number of letters in S2)

1. If S1 contains characters that is in S2 change all such characters to \*

|  |  |  |
| --- | --- | --- |
| Method Name |  | modifyStrings |
| Method Description |  | Perform the above mentioned actions on a String |
| Argument |  | String,String |
| Return Type | String Array | |
| Logic | Do the above mentioned actions on the entered  String.  For Example S1=”JAVAJAVA”  S2=”VA’   1. **VA**A**VA**A**VA**A**VA**A (J replaced with VA, V   replaced with VA etc.)   1. JAVAJAAV 2. JAJAVA 3. VJAVAJAVAA 4. J\*\*\*J\*\*\*     **Output:**{“ **VA**A**VA**A**VA**A**VA**A”,”  JAVAJA**AV**”,” JAJAVA”,”  **V**JAVAJAVA**A**“,”J\*\*\*J\*\*\*“} | |

**Exercise 4:**You are asked to create an application for registering the details of jobseeker. The requirement is:

Username should always end with **\_job** and there should be atleast minimum of 8 characters to the left of **\_job**. Write a function to validate the same. Return true in case the validation is passed. In case of validation failure return false.

|  |  |
| --- | --- |
| Method Name | validateUserName |
| Method Description | Checks if the username is valid |
| Argument | String userName |
| Return Type | boolean |
| Logic | Checks if the username ends with \_job and  contains at least 8 characters to the left of \_job. If valid return true. Else return false. |

**Exercise 5:** Create a method that accepts a number and modifies it such that the each of the digit in the newly formed number is equal to the difference between two consecutive digits in the original number. The digit in the units place can be left as it is. Note: Take the absolute value of the difference. Ex: 6-8 = 2

|  |  |
| --- | --- |
| Method Name | modifyNumber |
| Method Description | Accepts a number and modify it as per the requirement |
| Argument | int number1 |
| Return Type | int |
| Logic | Accept a number and modify it such that the each of the digit in the newly formed number is equal to the difference between two consecutive digits in the original number. For example. Input: 45862  Output:13242 **Algorithm:**  1. Convert number into String  2.Extract each char using charAt method  3. Convert char to int and find the difference  4.Create new StringBuffer object and keep adding the difference  5. Finally convert StringBuffer to int |

**Exercise 6:** Create a method which can perform a particular String operation based on the user’s choice. The method should accept the String object and the user’s choice and return the output of the operation. Options are

A: Add the String to itself

B: Replace alternate positions with \*

C: Remove duplicate characters in the String

D: Change alternate characters to upper case

|  |  |
| --- | --- |
| Method Name | changeString |
| Method Description | Modify the string based on user choice |
| Argument | String string, char ch |
| Return Type | String |
| Logic | Perform the required operation based on the user choice and return the resulting string |

**Exercise 7:** Create a method that accepts a String and checks if it is a positive string. A string is considered a positive string, if on moving from left to right **each** character in the String comes after the previous characters in the Alphabetical order.

For Example

ANT is a positive String (Since T comes after N and N comes after A) APPLE is not positive since L comes before P in the alphabetical order. The method should return true if the entered string is positive

|  |  |
| --- | --- |
| Method Name | checkPositive |
| Method Description | Checks if a String is positive |
| Argument | String |
| Return Type | boolean |
| Logic | Check if a string is positive based on the above  criteria and return true if positive.  Hint:  **Step 1:** Convert to Char array.  **Step 2:** Iterate through array, subtract 1st two characters (A-N). This will give the ASCII difference  **Step 3:** If result is negative, then return false and |
|  | break. Else continue to next loop |

**Exercise 8:** A company requires each employee to maintain a secret code. The secret code needs to pass certain validation for getting accepted.

The validation rules are as given

1. The secret code should be six characters long
2. The first three characters should be cod (Use .startsWith method)
3. There should be at least one digit in the code (Use .isDigit)
4. The first character should always be an upper case letter(Use isUpperCase)
5. The code should contain only alphabets and digits.
6. The number of upper case letters should be greater than lower case letters.

Return true if the above validation is passed.

|  |  |
| --- | --- |
| Method Name | validateCode |
| Method Description | Validate the entered code as per the given validation rules |
| Argument | String code |
| Return Type | boolean |
| Logic | Validate the entered code  Hint: Use the String API methods to extract each character |