**CSA09 Programming in Java**

**Day 2**

**ASSIGNMENT – 2**

**Questions for Developing a code in Java**

1. Write a java program
2. to compare two strings lexicographically, ignoring case differences.
3. to check whether a given string ends with the contents of another string.
4. to print current date and time in the specified format.
5. to get the index of all the characters of the alphabet.
6. To replace each substring of a given string that matches the given regular expression with the given replacement. In the below string replace all the fox with cat.
7. to get a substring of a given string between two specified positions.
8. to trim any leading or trailing whitespace from a given string.
9. to convert all the characters in a string to lowercase.
10. to get the length of a given string.
11. to check whether two String objects contain the same data

Sample string: "The quick brown fox jumps over the lazy dog."

Import java.time.LocalDateTime;

Import java.time.format.DateTimeFormatter;

Import java.util.ArrayList;

Import java.util.List;

Import java.util.regex.Matcher;

Import java.util.regex.Pattern;

Public class StringOperations {

Public static void main(String[] args) {

// i. Compare two strings lexicographically, ignoring case differences

String str1 = “apple”;

String str2 = “APPLE”;

If (str1.equalsIgnoreCase(str2)) {

System.out.println(“Strings are equal (ignoring case)”);

} else {

System.out.println(“Strings are not equal (ignoring case)”);

}

// ii. Check whether a given string ends with the contents of another string

String mainStr = “Hello world”;

String subStr = “rld”;

If (mainStr.endsWith(subStr)) {

System.out.println(“String ends with the contents of the given string”);

} else {

System.out.println(“String does not end with the contents of the given string”);

}

// iii. Print current date and time in the specified format

LocalDateTime now = LocalDateTime.now();

DateTimeFormatter formatter = DateTimeFormatter.ofPattern(“dd/MM/yyyy HH:mm:ss”);

System.out.println(“Current date and time: “ + now.format(formatter));

// iv. Get the index of all the characters of the alphabet

String str = “The quick brown fox jumps over the lazy dog.”;

List<Integer> indexes = new ArrayList<>();

For (char ch = ‘a’; ch <= ‘z’; ch++) {

Int index = str.indexOf(ch);

If (index != -1) {

Indexes.add(index);

}

}

System.out.println(“Indexes of all the characters of the alphabet: “ + indexes);

// v. Replace each substring of a given string that matches the given regular expression with the given replacement

String regex = “fox”;

String replacement = “cat”;

String result = str.replaceAll(regex, replacement);

System.out.println(“Result after replacement: “ + result);

// vi. Get a substring of a given string between two specified positions

Int startPos = 4;

Int endPos = 9;

String substring = str.substring(startPos, endPos);

System.out.println(“Substring between “ + startPos + “ and “ + endPos + “: “ + substring);

// vii. Trim any leading or trailing whitespace from a given string

String strWithWhitespace = “ Hello world “;

String trimmedStr = strWithWhitespace.trim();

System.out.println(“Original string: ‘” + strWithWhitespace + “’”);

System.out.println(“Trimmed string: ‘” + trimmedStr + “’”);

// viii. Convert all the characters in a string to lowercase

String upperCaseStr = “CONVERT TO LOWERCASE”;

String lowerCaseStr = upperCaseStr.toLowerCase();

System.out.println(“Original string: “ + upperCaseStr);

System.out.println(“Lowercase string: “ + lowerCaseStr);

// ix. Get the length of a given string

String someString = “Some string”;

Int length = someString.length();

System.out.println(“Length of the string ‘” + someString + “’: “ + length);

// x. Check whether two String objects contain the same data

String strA = “The quick brown fox jumps over the lazy dog.”;

String strB = “The quick brown fox jumps over the lazy dog.”;

If (strA.equals(strB)) {

System.out.println(“Strings contain the same data”);

} else {

System.out

}}}

1. Implement a class Account. An account has

* a balance
* functions to add
* and withdraw money,
* And a function to inquire the current balance.

Condition:

1. Pass a value into a constructor to set an initial balance.
2. If no value is passed the initial balance should be set to $0.
3. Charge a $5 penalty if an attempt is made to withdraw more money than available in the account.
4. Enhance the Account class to compute interest on the current balance. (10)

Public class Account {

Private double balance;

Public Account() {

This.balance = 0.0;

}

Public Account(double initialBalance) {

This.balance = initialBalance;

}

Public void deposit(double amount) {

This.balance += amount;

}

Public void withdraw(double amount) {

If (amount > this.balance) {

System.out.println(“Insufficient funds. A $5 penalty will be charged.”);

This.balance -= 5.0;

} else {

This.balance -= amount;

}

}

Public void computeInterest(double rate) {

Double interest = this.balance \* rate / 100.0;

This.balance += interest;

}

Public double getBalance() {

Return this.balance;

}

}

//Here’s an example usage of the Account class:

Public class AccountDemo {

Public static void main(String[] args) {

Account account = new Account(1000.0); // initialize account with $1000

Account.deposit(500.0);

System.out.println(“Current balance after deposit: “ + account.getBalance());

Account.withdraw(1500.0);

System.out.println(“Current balance after withdrawal: “ + account.getBalance());

Account.withdraw(1500.0);

System.out.println(“Current balance after withdrawal: “ + account.getBalance());

Account.computeInterest(2.5);

System.out.println(“Current balance after interest computation: “ + account.getBalance());

}

}

Questions for Debugging a code in Java

1. Given two strings needle and haystack, return the index of the first occurrence of needle in haystack, or -1 if needle is not part of haystack.

Example 1:

Input: haystack = "sadbutsad", needle = "sad"

Output: 0

Explanation: "sad" occurs at index 0 and 6.

The first occurrence is at index 0, so we return 0.

Public int indexOfString(String haystack, String needle) {

If (haystack == null || needle == null) {

Return -1;

}

Int lenHaystack = haystack.length();

Int lenNeedle = needle.length();

For (int I = 0; I <= lenHaystack – lenNeedle; i++) {

If (haystack.substring(I, I + lenNeedle).equals(needle)) {

Return I;

}

}

Return -1;

}

Example 2:

Input: haystack = "leetcode", needle = "leeto"

Output: -1

Explanation: "leeto" did not occur in "leetcode", so we return -1.

Constraints:

1 <= haystack.length, needle.length <= 104

haystack and needle consist of only lowercase English characters.

Public int lengthOfLastWord(String s) {

If (s == null || s.isEmpty()) {

Return 0;

}

S = s.trim();

Int lastSpaceIndex = s.lastIndexOf(‘ ‘);

If (lastSpaceIndex == -1) {

Return s.length();

} else {

Return s.length() – lastSpaceIndex – 1;

}

}

Given a string s consisting of words and spaces, return the length of the last word in the string.

A word is a maximal

substring

consisting of non-space characters only.

Example 1:

Input: s = "Hello World"

Output: 5

Explanation: The last word is "World" with length 5.

Example 2:

Input: s = " fly me to the moon "

Output: 4

Explanation: The last word is "moon" with length 4.

Example 3:

Input: s = "luffy is still joyboy"

Output: 6

Explanation: The last word is "joyboy" with length 6.

Constraints:

1 <= s.length <= 104

s consists of only English letters and spaces ' '.

There will be at least one word in s.

Public int lengthOfLastWord(String s) {

// Trim the string to remove leading and trailing spaces

S = s.trim();

Int len = s.length();

Int lastWordLen = 0;

// Traverse the string from right to left and count the length of the last word

For (int I = len – 1; I >= 0; i--) {

// If a space is encountered, return the length of the last word

If (s.charAt(i) == ‘ ‘) {

Return lastWordLen;

}

// Otherwise, increment the length of the last word

lastWordLen++;

}

// If the string contains only one word, return its length

Return lastWordLen;

}

Questions for Finding error in Java to determine the factor

import java.io.\*;

import java.util.\*;

class factor {

public static void main(String args[]) {

try {

Scanner sc=new Scanner(System.in);

in count=0,n=100,i,j=0,m=4;

int []a=new int [10];

System.out.println("Enter the number:");

n=sc.nextInt();

if(n<=0)

{

System.out.println("Enter valid number");

}

else {

for(i=1;i<=n;i--);

{

if(n%i!=0)

{

a[j] = i;

System.out.println("..." + i);

count++;

j++;

}

}

System.out.println("The number of factors:"+count);

}

System.out.println(m + "th item " + a[m-1]);

}

catch(Exception e) {

System.out.println("Enter only numbers");

}

}

}

Correct code:

Import java.util.\*;

Class Factor {

Public static void main(String args[]) {

Try {

Scanner sc = new Scanner(System.in);

Int count = 0, n = 100, I, j = 0, m = 4;

Int[] a = new int[10];

System.out.println(“Enter the number:”);

N = sc.nextInt();

If (n <= 0) {

System.out.println(“Enter valid number”);

Return;

} else {

For (I = n; I >= 1; i--) {

If (n % I == 0) {

A[j] = I;

System.out.println(“…” + i);

Count++;

J++;

}

}

System.out.println(“The number of factors:” + count);

}

System.out.println(m + “th item “ + a[m – 1]);

} catch (Exception e) {

System.out.println(“Enter only numbers”);

}

}

}