***CSC 3020***

***Java Programming***

**Assignment 02**

**40 points**

**Due 09/27/2023 (11:45 A.M.)**

Assignment Objectives:

■■ To compare and test characters using the static methods in the **Character** class.

■■ To represent strings using the **String** class.

■■ To format output using the **System.out.printf** method.

■■ To write programs for executing statements repeatedly using a **while** loop.

■■ To follow the loop design strategy to develop loops.

■■ To control a loop with the user confirmation or a sentinel value.

■■ To write loops using **do-while** statements.

■■ To write loops using **for** statements.

■■ To discover the similarities and differences of three types of loop statements.

■■ To write nested loops.

■■ To implement program control with **break** and **continue.**

■■ To process characters in a string using a loop.

**Solution to this assignment will not be posted on Canvas; however, any question can be discussed in the class upon request of a student.**

All assignments must be submitted by the Canvas. **No email or hard copy** is accepted. You must follow the following format:

1. For non-programming questions, use a word file to type your answers. Don’t use the text box on the Canvas to answer the questions or to write comments, we will not read it.
2. State your answer clearly.
3. For programming questions, include only the source file for each problem.
4. Submit your file to the Canvas. You must submit your assignment on time; otherwise, you will receive zero. In addition, you cannot submit your file more than one time.
5. There will be several folders on the Canvas. You need to upload your file(s) using the correct folder on the Canvas.
6. Name each file: “Assignment Number(Question number(s))”.
7. To upload your file(s):

* In Course Navigation, click the Assignments link.
* Click the title of the assignment.
* Click the **Submit** Assignment button.
* Add **File**. ...
* Add Another **File**. ...
* **Submit** Assignment. ...
* View **Submission**.

**It is your responsibility to make sure that each file is uploaded correctly. If you uploaded a wrong file, you receive zero; files will not be accepted after due date even if you have a prove that the file is created before the due date.**

**Make sure you review the Cheating & Plagiarism policy on Canvas.**

Answer questions 1 to 8 on a single text file (include question number and the answer); write a program for each of Q.9 - Q.11. Covert .java files to text files. Submit 4 text files to Canvas by the due date.

1. **(4 points)** Give a brief answer for each of the following questions:
2. Which of the following are correct literals for characters?

'1', '\u345dE', '\u3fFa', '\b', '\t'

‘1’ -> 1

\u345dE'->

㑝E

'\u3fFa'->

㿺

‘b’ -> backspace

‘t’ -> tab

1. Write the code that generates a random lowercase letter.

public class Main

{

public static void main(String[] args) {

//Unicode provides less flexibility in limiting an alphabet than

//this structure which most closely resembles a cipher key

String lowercaseletters = "abcdefghijklmnopqrstuvwxyz";

int randomLetter = (int)(Math.random() \* lowercaseletters.length());

char letterOutput = lowercaseletters.charAt(randomLetter);

System.out.println(letterOutput);

}

}

1. What is wrong in the following code?

The issue is that the nextline character after nextint is eating whatever isn’t an int when the user gives input. Therefore instead of reading the next user input, it is eating whatever is left over from the previous user input.

**import** java.util.Scanner;

**public** **class** Test {

**public** **static** **void** main(String[] args) {

Scanner input = **new** Scanner(System.in);

System.out.print("Enter an integer: ");

**int** value = input.nextInt();

System.out.println("The value is " + value);

System.out.print("Enter a line: ");

String line = input.nextLine();

System.out.println("The line is " + line);

}

}

1. What are the format specifiers for outputting a Boolean value, a character, a decimal integer, a floating-point number, and a string?

Boolean value -> %b

Character value -> %c

Decimal integer -> %d

Floating point number -> %f

String -> %s

1. **(2 points)** Write an expression that obtains a random integer between 34 and 55 inclusive. Write an expression that obtains a random integer between 0 and 999.

**//note to self, Math.random() does not include the upper bound of 1**

**Int randomthing = (int)(Math.random() \* 22) + 34;**

**Int randomthing2 = (int)(Math.random() \* 1000);**

1. **(4 points)** Can the following conversions involving casting be allowed? If so, find the converted result.

**char** c = 'A';

**int** i = (**int**)c;

**//this gives 65**

**float** f = 1000.34f;

**int** i = (**int**)f;

**//this gives 1000**

**double** d = 1000.34;

**int** i = (**int**)d;

**//this also gives 1000.**

**int** i = 97;

**char** c = (**char**)i;

**//this gives ‘a’**

1. **(6.5 points)** Suppose that s1, s2, and s3 are three strings, given as follows:

String s1 = "Welcome to Java";

String s2 = "Programming is fun";

String s3 = "Welcome to Java";

What are the results of the following expressions?

a. s1 == s2

false

b. s1 == s3

true

c. s1.equals(s2)

false

d. s1.equals(s3)

true

e. s1.compareTo(s2)

7

f. s2.compareTo(s3)

-7

g. s2.compareTo(s2)

0

h. s1.charAt(0)

w

i. s1.indexOf('j')

-1

j. s1.length()

15

k. s1.substring(5)

“me to Java”

l. s1.toLowerCase()

welcome to java

m. s1.toUpperCase()

WELCOME TO JAVA

1. **(3.5 points)** Suppose that s1 and s2 are two strings. Which of the following statements or expressions are incorrect?

String s1 = "Welcome to Java";

String s2 = "Welcome to Java";

1. String s3 = s1 + s2;

//this isn’t per say incorrect as it gives an output.   
Welcome to JavaWelcome to Java

1. String s3 = s1 - s2;  
   incorrect
2. s1 == s2;  
   true
3. s1 >= s2;  
   incorrect. Bad operand type
4. s1.compareTo(s2);  
   0
5. **char** c = s1(0);  
   incorrect, this isn’t python :c.
6. **char** c = s1.charAt(s1.length());  
   incorrect, this produces an error because indexes start at zero.
7. **(3 points)** Show the output of the following statements.

(a) System.out.printf("amount is %.1f\n", 32.36);

Amount is 32.4

(b) System.out.printf("amount is %5.2d\n", 32.327);

Exception error. This is because the number provided is a double, not a floating point integer.

(c) System.out.printf("%6b\n", (1 > 2));  
false

(d) System.out.printf("%6s\n", "Java");  
 Java  
//yes those spaces are included, look at them. They are there.

(e) System.out.printf("%6b%-8s\n", (1 > 2), "Java");  
falseJava

(f) System.out.printf("%,5d %,6.1f\n", 312342, 315562.932);  
312,342 315,562.9

1. **(2 points)** Convert the following for loop statement to a while loop and to a do-while loop:

**long** sum = 0;

**for** (**int** i = 0; i <= 1000; i++)

sum = sum + i;  
  
public class Main

{

public static void main(String[] args) {

long sum = 0;

int i = 0;

do{

sum = sum+i;

i++;

}while(i<=1000);

System.out.println(sum);

}

}

//this gives 500500, which is the same.

1. **(2 points)** The for loop in (a) is converted into the while loop in (b). What is wrong? Correct it.

(a)

**int** sum = 0;

**for** (**int** i = 0; i < 4; i++) {

**if** (i % 3 == 0) **continue**;

sum += i;

}

(b)

**int** i = 0, sum = 0;

**while** (i < 4) {

**if** (i % 3 == 0) **continue**;

sum += i;

i++;

}  
//The issue is that the continue statement comes first, meaning that i will get stuck in an infinite continue loop when it is divisible by 3. This is easily fixed, just move that continue statement to the back.   
  
int i = 0, sum = 0;

while (i < 4) {

i++;

if (i % 3 == 0){

continue;

}

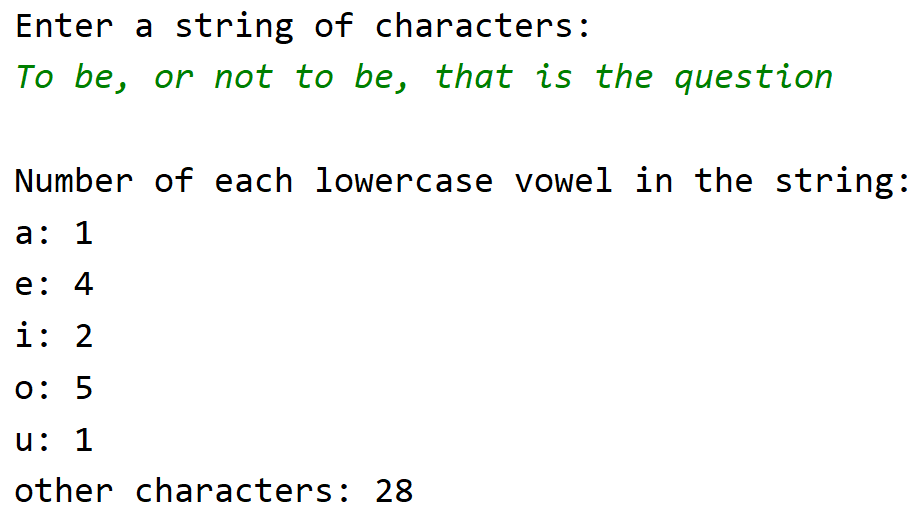
sum += i;

}

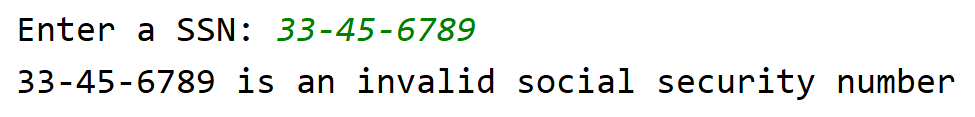
**Programming Questions**

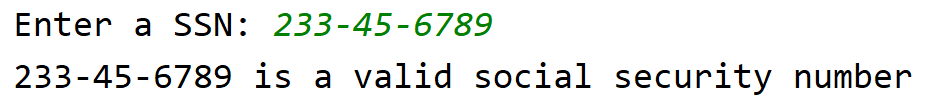
1. **(5 points)** Write a class (Vowels) that reads a string from the user, then determines and prints how many of each lowercase vowel (a, e, i, o, and u) appear in the entire string. Have a separate counter for each vowel. Also count and print the number of nonvowel characters.

**Sample output:**

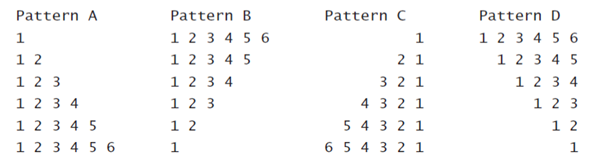


1. **(4 points)** Write a class (Social Security) that prompts the user to enter a Social Security number in the format DDD-DD-DDDD, where D is a digit. Your program should check whether the input is valid. Here are sample runs:

****

****

1. **(4 points)** Use nested loops that display the following patterns in four separate nested loops.

****