German University in Cairo
Media Engineering and Technology
Prof. Dr. Slim Abdennadher
Dr. Wael Abouelsaadat
Dr. Mohammed Abdel Megeed

Introduction to Computer Programming, Spring Term 2017
Practice Assignment 3

Discussion: 4.3.2017 - 9.3.2017

Exercise 3-1 Adder

Write a program that adds up integers that the user enters. First the programs asks how many numbers will be added up. Then the program prompts the user for each number. Finally, it prints the sum.

The output should be something like:

```
How many integers will be added:
5
Enter integer 1:
3
Enter integer 2:
4
Enter integer 3:
-4
Enter integer 4:
-3
Enter integer 5:
7

The sum is 7
```

Solution:

```
import java.util.*;
public class Adding {
  public static void main (String[] args)
  {
    Scanner sc = new Scanner (System.in);

    System.out.println("How many integers will be added?");
    int n = sc.nextInt();
    int i, x, sum = 0;

    for(i = 0; i<n ; i++)
    {
       System.out.println("Enter integer "+(i+1));
       x = sc.nextInt()
       sum += x;
    }
    System.out.println("The sum= "+sum);
  }
}
```

Exercise 3-2 Euclidean Algorithm

The Euclidean algorithm determines the greatest common divisor (GCD) of two positive numbers by repeatedly replacing the larger number with the result of subtracting the smaller one from it until the two numbers are equal.

Write a Java program for Euclidean algorithm where the user has to enter the two numbers and the program should calculate their greatest common divisor. The output should be something like:

```
Please, enter a first number:
45
Please, enter a second number:
22
The GCD of 45 and 22 is 1
```

Solution:

```
import java.util.*;
public class Euclidian {
    public static void main(String[] args)
    {
      Scanner sc = new Scanner (System.in);
      System.out.println("Please, enter a number:");
      int num1 = sc.nextInt();
      System.out.println("Please, enter a second number:");
      int num2 = sc.nextInt();

      int num1Saved = num1;
      int num2Saved = num2;

      while(num1 != num2)
      {
        if(num1 > num2)
            num1 -= num2;
        else
            num2 -= num1;
      }
      System.out.println("The GCD of " + num1Saved + " and " + num2Saved +
                         " is " + num1);
    }
}
```

Exercise 3-3 Caesar Cipher
To be discussed in the tutorials

Write a Java program which takes two input variables `message` of data type `String` and `key` of data type `int`. The program should shift each character in `message` with a distance of `key`. For example: if `key=3` then `a` will be replaced by `d` and `b` will be replaced by `e` and so on.

Hint: You can use the following method

- `charAt(int index)`: Returns the character at the specified index. The first character of the sequence is at index 0, the next at index 1 and so on.

```
String s = "Hello";
char   c = s.charAt(0);
```

The value of `c` is `'H'`.

The output should be somthing like this:

```
Please enter the Message:
Hat
Please Enter the Key:
3
The encrypted word is:
Kdw
```

Solution:

```
import java.util.*;
public class Caesar {
   public static void main(String[] args)
   {
     Scanner sc = new Scanner (System.in);
     System.out.println("Please␣Enter␣a␣Word␣to␣be␣Encrypted:␣");
     String  s  = sc.nextLine();
     System.out.println("Please␣Enter␣a␣Key:␣");
     int     key = sc.nextInt();

     char x;
     int l = s.length();
     int i = 0;
     int ascii = 0;

     while (i<l)
     {
       ascii =  s.charAt(i) + key;
       if((ascii>122 && s.charAt(i)>=97 && s.charAt(i)<=122)
            || (ascii>90 && s.charAt(i)>=65 && s.charAt(i)<=90))
          ascii -= 26;
       x = (char) ascii;
       System.out.print(x);
       i++;
     }
     System.out.print("\n");
   }
}
```

Exercise 3-4 String Manipulation

Write a program that determines the number of consonants, vowels, punctuation characters, and spaces in an input line. Read in the line into a String (in the usual way). Now use the `charAt()` method in a loop to access the characters one by one. Use a switch statement to increment the appropriate variables based on the current character. After processing the line, print out the results.

Solution:

```
import java.util.*;
public class Chars
{
   public static void main (String[] args)
   {
      Scanner sc = new Scanner (System.in);
      int consonant = 0, vowel = 0, punctuation = 0, space = 0, digit = 0;
      String str, lowered;
      System.out.print("Enter␣a␣string:␣");
```

```
        str      = sc.nextLine();

        //Convert the letter of the String 'str' to lower case.
        lowered = str.toLowerCase();

        for (int i = 0; i<lowered.length(); i++){

          switch(lowered.charAt(i)){
            case ' ': space++; break;
            case 'a':
            case 'e':
            case 'i':
            case 'o':
            case 'u': vowel++; break;
            case 'b':
            case 'c':
            case 'd':
            case 'f':
            case 'g':
            case 'h':
            case 'j':
            case 'k':
            case 'l':
            case 'm':
            case 'n':
            case 'p':
            case 'q':
            case 'r':
            case 's':
            case 't':
            case 'v':
            case 'w':
            case 'x':
            case 'y':
            case 'z': consonant++; break;
            case '0':
            case '1':
            case '2':
            case '3':
            case '4':
            case '5':
            case '6':
            case '7':
            case '8':
            case '9': digit++; break;
            default:  punctuation++;
          }
        }
        System.out.println ("Number of consonants: " + consonant);
        System.out.println ("Number of vowels: " + vowel);
        System.out.println ("Number of digits: " + digit);
        System.out.println ("Number of punctuations: " + punctuation);
        System.out.println ("Number of spaces: " + space);
      }
}
```

Exercise 3-5 Fixed Length

Write a program that asks the user to enter two words. The program then prints out both words on one line. The words will be separated by enought dots so that the total line length is 30. We can use it to make an index for a book. The user enters the name of the chapters/sections and the page number and the program generate the index. You can only print one dot at a time.

```
Enter first word:
Chapter 5
Enter second word:
153

Chapter 5...................153
```

Solution:

```
import java.util.*;
public class Word {
   public static void main (String[] args)
   {
     Scanner sc = new Scanner (System.in);
     String word1, word2, line = "";
     int dots;

     System.out.print ("Enter first word: ");
     word1 = sc.nextLine();
     System.out.print ("Enter second word: ");
     word2 = sc.nextLine();

     dots  = 30 - (word1.length() + word2.length());
     line  = line.concat(word1);

     for (int i = 0; i<dots; i++)
        line = line.concat(".");

     line = line.concat(word2);
     System.out.println (line);
   }
}
```

Exercise 3-6 Stream of Numbers

Write a Java program to read a list of nonnegative integers and outputs the maximum integer, the minimum integer, and the average of all the integers. The end of the input is indicated by the user entering a negative number. Note that the negative number is not used in finding the maximum, minimum, or average. The output should be something like this:

```
Please enter a sequence of positive numbers
2
3
5
4
-1
The maximum number is : 5
The minimum number is: 2
The average is: 3.5
```

Use in one program a `while` loop and in another program a `do while` loop.

Solution:

- Using While

```
import java.util.*;
public class NumbersWhile {
  public static void main(String[] args)
  {
    Scanner sc = new Scanner (System.in);
    System.out.println("Please␣enter␣the␣number");
    int num    = sc.nextInt();
    if(num<0)
       System.out.println("No␣positive␣Numbers␣entered");
    else {
      int small, large;
      small = num;
      large = num;
      double sum = 0;
      double avg;
      int count  = 0;

      while (num>=0)
      {
        if(num<small)
          small = um;
        else if (num>large)
          large = num;

        sum += num;
        count++;
        System.out.println("Please␣enter␣another␣number:");
        num = sc.nextInt();
      }
      avg = sum/count;
      System.out.println("The␣average␣of␣the␣numbers␣is␣␣␣␣␣␣" + avg);
      System.out.println("The␣smallest␣integer␣you␣entered␣is␣" + small);
      System.out.println("The␣largest␣integer␣you␣entered␣is␣␣" + large);
    }
  }
}
```

- Using Do While

```
import java.util.*;
class NumbersDoWhile {
  public  static void main (String[] args)
  {
    Scanner sc = new Scanner (System.in);
    System.out.println("Please␣Enter␣a␣sequence␣of␣positive␣numbers:␣");
    int i;
    //Getting the first number and assigning it to min and max
    i = sc.nextInt();
    int max = i, min = i, sum = 0, count = 0;
    if(i<0)
      System.out.println("No␣positive␣numbers␣entered");
    else
    {
      do{
```

```
            if(max<i)
              max = i;
            if(min>i)
              min = i;
            sum += i;
            count++;
            i = sc.nextInt();
          } while(i>0);

          double avg = (double) sum / count;
          System.out.println("The maximum number is " + max);
          System.out.println("The minimum number is " + min);
          System.out.println("The average is "         + avg);
       }
      }
    }
```

Exercise 3-7 Triangle N

Write a Java program to construct a triangle shape of numbers given that `n` is an input from the user. For example if `n=6`, the shape should look like the following:

```
1
12
123
1234
12345
123456
```

Solve using a single loop only.

Solution:

```
import java.util.*;
public class Triangle {
    public static void main(String[] args)
    {
      Scanner sc = new Scanner(System.in);
      System.out.println("Please enter the number");
      int n = sc.nextInt();

      int i,j;
      String s = "";

      for(i = 1; i <= n; i++){
         s += i;
         System.out.println(s);
      }
    }
}
```

Exercise 3-8 Pyramid
To be discussed in the labs

Construct the following pyramid of numbers given that `n` is an input from the user. For example if `n=9`, the pyramid

should look like the following:

```
    1
   123
  12345
 1234567
123456789
```

Solution:

```
import java.util.*;
public class TriangleNumbers{
  public static void main(String[] args)
  {
    String line;
    Scanner sc = new Scanner(System.in);
    System.out.println("Please␣enter␣a␣number␣between␣1␣and␣9");
    int n = sc.nextInt();

    int i, m, j, z;
    if (n/2 < 5)
    {
       for(i = 1; i <= (n/2)+1; i++)
       {
          for(z = (n/2)-i+1; z>0; z--)
             System.out.print("␣");
          for(j = 1; j <= (i*2-1); j++)
             System.out.print(j);
          System.out.println();
       }
    }
    else
       System.out.println("You␣entered␣a␣big␣number␣range");
  }
}
```

Exercise 3-9 Word Count

Write a program that reads a sentence and a word from the user and finds the number of occurrences of the given word in the sentence. For example, the following could be a run of your program

```
Enter the sentence:
the students are enjoying life at the GUC
Enter the word:
the
The sentence is "the students are enjoying life at the GUC".
The word "the" occurs two times in the sentence
```

Solution:

```
import java.util.*;
public class WordFinder{
  public static void main(String[] args)
  {
    Scanner sc  = new Scanner(System.in);
```

```
    System.out.println("Please␣enter␣the␣sentence");
    String line = sc.nextLine();
    System.out.println("Please␣enter␣the␣key␣word");
    String word = sc.nextLine();

    int lineLength  = line.length();
    int wordLength  = word.length();
    int wordCounter = 0;

    //Consider the '.' value variation. In case the user forgets it.
    if( line.charAt(lineLength-1) != '.' )
    {
      line = line+'.';
      lineLength++;
    }
    int i = 0;
    int pointer = 0;
    char c;
    String toBeCompared;

    while(i < lineLength-1)
    {
      //Get word by word
      do
      {
        c = line.charAt(pointer);
        pointer++;
      }while((c != '␣') && (c != ';') && (c != '.')
             && (c != ',') && (pointer < lineLength));

      //substring(x,y) returns the word starting from x to y-1
      toBeCompared = line.substring(i, pointer-1);

      //Now,compare only if the word length equals the key-word length
      if(wordLength == toBeCompared.length())
      {
        //Comparing the words ignoring Upper and Lower cases
        if(word.equalsIgnoreCase(toBeCompared) == true)
           wordCounter++;
      }
      i = pointer;
    }
    System.out.println("Number␣of␣Occurence:␣" + wordCounter);
  }
}
```

Exercise 3-10 Number of Digits

Write a Java program that reads from the user positive integers and count the number of digits in them. The program should keep asking the user for entering integers until he/she enters -1. The output should be something like this:

```
Please enter a number
524
Number of digits in 524 = 3
Please enter a number
24
```

```
Number of digits in 24 = 2
Please enter a number
35790
Number of digits in 35790 = 5
Please enter a number
-1
Thank you!
```

Solution:

```
import java.util.*;
public class DigitCounter {
  public static void main(String[] args)
  {
    System.out.println("Please␣enter␣the␣first␣number");
    int num1 = sc.nextInt();
    int count, num2;

    while(num1 != -1)
    {
      count = 1;
      num2  = num1;

      while(num2/10 != 0)
      {
        num2 = num2/10;
        count++;
      }
      System.out.println("Number␣of␣digits␣in␣" + num1 + "␣is␣" + count);
      System.out.println("Please␣enter␣the␣next␣number");
      num1 = sc.nextInt();
    }
  }
}
```

Exercise 3-11 Divisors

Which integer between 1 and 10000 has the largest number of divisors, and how many divisors does it have? Write a program to find the answers and print out the results. It is possible that several integers in this range have the same, maximum number of divisors. Your program has to print out one of them.

Solution:

```
public class MostDivisors {
   public static void main(String[] args)
   {
     int maxDivisors = 1;
     int numWithMax  = 1;
     for (int n = 2;  n <= 10000;  n++)
     {
        int divisorCount = 0;
        for (int d = 1;  d <= n;  d++)
        {
           if ( n % d == 0 )
              divisorCount++;
        }
```

```
            if (divisorCount > maxDivisors)
            {
                maxDivisors = divisorCount;
                numWithMax  = n;
            }
        }
        System.out.println("Among integers between 1 and 10000,");
        System.out.println("The max number of divisors is " + maxDivisors);
        System.out.println("A num with" + maxDivisors + "div is" + numWithMax);
    }
}
```

Exercise 3-12 Extract Numbers

Write a Java program that takes a string containing text and nonnegative numbers form the user and prints out the numbers contained in the string in separate lines. Use nested loops.

Running example

```
Please enter your string
The year has 365 days and the day has 12 hours
Output
The numbers contained in your string are
365
12
```

Solution:

```
import java.util.*;
public class ExtractNumbers {
    public static void main(String[] args)
    {
        Scanner sc = new Scanner(System.in);
        System.out.println("Please enter your string");
        String s = sc.nextLine();

        int i    = 0;
        int flag = 0;

        while( i < s.length() )
        {
          //the end of the string is not reached and there is a number between 0 and
          while( i<s.length() && s.charAt(i)<=57 && s.charAt(i)>=48 )
          {
            System.out.print(s.charAt(i));
            i++;
            flag++;
          }
          if(flag > 0)
          {
            flag = 0;
            System.out.println();
          }
          i++;
       }
    }
}
```

Exercise 3-13 Midterm Spring 2013
To be discussed in the tutorial

The method `Math.random()` gives a real number between `0.0` and `0.9999...`, and so `6*Math.random()` is between `0.0` and `5.999...`. The type-cast operator, `(int)`, can be used to convert this to an integer: `(int)(6*Math.random())`. Thus, `(int)(6*Math.random())` is one of the integers 0, 1, 2, 3, 4, and 5. To get a number between 1 and 6, we can add 1:

```
(int)(6*Math.random()) + 1
```

Using the statement above, we would like to know how many times we have to roll a pair of dice before they come up snake eyes? **Note:** Snake eyes means that both dice show a value of 1. Write a method that should return the number of rolls that it makes before the pair of dice come up snake eyes. The method should also display the following message, e.g.:

```
It took 100 rolls to get snake eyes.
```

Note: You have to use a do-while loop.

Solution:

```
public static void main(String[] args) {
    int die1, die2; // The values rolled on the two dice.
    int countRolls; // Used to count the number of rolls.
    countRolls = 0;
    do {
        die1 = (int)(Math.random()*6) + 1; // roll the dice
        die2 = (int)(Math.random()*6) + 1;
        countRolls++; // and count this roll
    } while ( die1 != 1 || die2 != 1 );

    System.out.println("It␣took␣" + countRolls + "␣rolls␣to␣get␣snake␣␣␣␣␣eyes.");
}
```

Exercise 3-14 Run Length
To be discussed in the tutorial

a) Given a String containing uppercase characters (A-Z), write a Java program that compresses repeated 'runs' of the same character by storing the length of that run.

Example:

```
Input: WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWWWWWWWWWWWWWWWWWWWWBWWWWWWWWWWWWWW
Output: 12W1B12W3B24W1B14W
```

Solution:

```
public static void main(String[]args)
{
        String input = "
            WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWWWWWWWWWWWWWWWWWWWWBWWWWWWWWWWWWWW
            ";

        int i=0;
        int count=0;
        char current;
```

```
        String output="";
        while(i<input.length())
        {
                current = input.charAt(i);
                while(i<input.length() && input.charAt(i) == current)
                {
                        count++;
                        i++;
                }

                output+=count +""+current;      // update output
                count=0; //reset counter
        }
        System.out.println("Input: " + input);
        System.out.println("Output: " + output);

}
```

b) Moreover, write a Java program that reverses the compression.

Example:

```
Input: 12W1B12W3B24W1B14W
Output: WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWWWWWWWWWWWWWWWWWWWWBWWWWWWWWWWWWWW
```

Solution:

```
public static void main(String[]args)
{
        String input = "12W1B12W3B24W1B14W";

        int i=0;
        int count=0;
        char current;
        String output="";
        String temp="";
        while(i<input.length())
        {
                current=input.charAt(i);

                //find count n
                while(i<input.length() && input.charAt(i)<=57  && input
                    .charAt(i)>=48) //it's a digit
                {
                        temp+=input.charAt(i);
                        i++;
                }
                count = Integer.parseInt(temp);

                //get character
                current = input.charAt(i);

                //concatenate n times
                for(int j=0;j<count;j++)
                        output+=current;

                //reset and update
                temp="";
                count=0;
                i++;
```

```
            }
            System.out.println("Input: " + input);
            System.out.println("Output: " + output);

    }
```