UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

NAME:	Abenes, Enrico O.	**DATE:**	July 11, 2023
COURSE:	CC1 – INTL 1	**SCHEDULE:**	1:30 pm – 5:20 pm MT

Activity Number:

Activity Type: Laboratory Activity

TITLE: Loops

LEARNING OBJECTIVES:

At the end of this activity, the students should be able to:

1. Identify available loop control structures in Java.

2. Differentiate the different loop control structures in Java.

3. Implement these different loop control structures for a given problem.

4. Create a complete Java program that simulates these basic operations.

INSTRUCTIONS:

1. Make sure you have your own individual account/monitor.

2. Always save your work and log off when not using the computer.

3. By now you should have been familiarized with using your text editor/net beans.

4. By now you should know how to create, save, compile, execute, and debug programs in Java.

5. Use the skills and learning obtained in Midterms for you to successfully finish the learning objectives of this module.

DURATION: One to two Meetings

HANDS-ON:

1. Log on using your own individual account. Use your own username and

password.

2. Open your text editor/NetBeans.

3. Write your next Java program:

```
/* Programmed by: <write your name here>
   Program title: Sum1.java
   Program Date: <write the date today here> */

import java.io.*;
public class Sum1{
  public static void main(String[] args){
      int start = 0, end = 0, sum;
      String input = " ";

      BufferedReader in = new BufferedReader(new
                            InputStreamReader(System.in));

      try{
        System.out.print("Input starting number: ");
        input = in.readLine();
        start = Integer.parseInt(input);

        System.out.print("Input ending number: ");
        input = in.readLine();
        end = Integer.parseInt(input);
      }catch(IOException e){
        System.out.println("Error!");
      }

      if(start >= end){
        System.out.print("Starting number should be lesser ");
        System.out.println("than the ending number. ");
        System.out.println("Try again.");
        System.exit(0);
      }

      if(start%2 == 0)
      {
            start = start + 1;
      }

      sum = 0;
      while(start <= end)
      {
            sum = sum + start;
            start = start + 2;
      }
      System.out.println("Sum = " + sum);
  }
}
```

3.1. Write your next program by copying the source code shown below to your text editor.

3.2. Save the program as Sum1.java then compile your program until no errors and warnings are reported.

3.3. Run your program.

3.4. Simulate and write what will be displayed on the screen.

```
Input starting number: 3

Input ending number: 10

Sum = 24
```

4. Now let's try another implementation. Copy the source code below and save it as Sum2.java

```
/* Programmed by: <write your name here>
   Program title: Sum2.java
   Program Date: <write the date today here> */

import java.io.*;
public class Sum2{
  public static void main(String[] args){
      int start = 0, end = 0, sum;
      String input = " ";

      BufferedReader in = new BufferedReader(new
                          InputStreamReader(System.in));
      try{
        System.out.print("Input starting number: ");
        input = in.readLine();
        start = Integer.parseInt(input);

        System.out.print("Input ending number: ");
        input = in.readLine();
        end = Integer.parseInt(input);
      }catch(IOException e){
        System.out.println("Error!");
      }

      if(start >= end){
        System.out.print("Starting number should be lesser ");
        System.out.println("than the ending number. ");
        System.out.println("Try again.");
        System.exit(0);
      }
      if(start%2 == 0){
            start = start + 1;
      }
      sum = 0;
      for(start = start; start <= end; start = start + 2){
            sum = sum + start;
      }
      System.out.println("Sum = " + sum);
  }
}
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

4.1. Simulate and write what will be displayed on the screen. Try using the same input values in your Sum1.java

```
Input starting number: 3

Input ending number: 10

Sum = 24
```

5. Let us try another implementation. Copy the source code and save it as Sum3.java

```
/* Programmed by: <write your name here>
   Program title: Sum3.java
   Program Date: <write the date today here> */

import java.io.*;
public class Sum3{
  public static void main(String[] args){
      int start = 0, end = 0, sum;
      String input = " ";

      BufferedReader in = new BufferedReader(new
                          InputStreamReader(System.in));
      try{
        System.out.print("Input starting number: ");
        input = in.readLine();
        start = Integer.parseInt(input);
        System.out.print("Input ending number: ");
        input = in.readLine();
        end = Integer.parseInt(input);
      }catch(IOException e){
        System.out.println("Error!");
      }
      if(start >= end){
        System.out.print("Starting number should be lesser ");
        System.out.println("than the ending number. ");
        System.out.println("Try again.");
        System.exit(0);
      }
      if(start%2 == 0){
            start = start + 1;
      }
      sum = 0;
      do{
            sum = sum + start;
            start = start + 2;
      }while(start <= end);

      System.out.println("Sum = " + sum);
  }
}
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

5.1. Simulate and write what will be displayed on the screen. Try using the same input values in your Sum1.java and Sum2.java

```
Input starting number: 3

Input ending number: 10

Sum = 24
```

6. After your simulation of the three programs, what do you think is the main objective of these programs?

Calculating the total of all odd numbers in a certain range is the main goal of the three Java files. To do this, they gather user input for the range's starting and ending numbers and conduct checks to confirm the accuracy of the information. After that, they iterate across the range, adding each odd number to the total, using various looping constructions (while loop, for loop, and do-while loop). All three programs attempt to compute the sum of odd numbers inside the given range, although employing various looping algorithms.

7. Identify what are the different loop control structures in Java.

for loop – It enables you to use an initialization phrase, condition, and update statement with a code block to run it again for a predetermined number of times.

while loop – If a certain condition is true, a block of code is repeatedly executed.

do-while loop – The only difference between it and the while loop is that it checks the condition at the end to make sure the loop runs at least once.

8. What are the usual similar components of these loop constructs?

Loop Condition – Before beginning each iteration, the condition is examined to determine whether the loop should be continued or ended.

Loop Body – As long as the loop condition is true, this particular piece of code will be run endlessly.

Loop Initialization – Before the loop starts, it requires initializing or setting up any necessary conditions or variables. It establishes the starting variables or conditions needed for the loop to begin.

Loop Update – It specifies how to update or increase the loop control variable after each iteration. At the conclusion of each iteration, it describes the procedure to update the loop control variable.

Loop Entry/Exit – Depending on the circumstance, it specifies how the loop is entered and exits.

9. What do you think is different among these identified loop constructs?

Justify your answer.

Java's **for, while, and do-while loops** are distinct from one another in terms of their structure, loop entry behavior, loop control, and usual usage.

The **for loop** has three distinct statements: initialization, condition, and update. It employs a loop control variable, verifies the condition before each iteration, and enters the loop after setup.

The **while loop**'s structure is more adaptable since it checks the condition before going into the loop body. It is dependent on an outside circumstance and continues as long as it holds true.

Although it also has a flexible structure, the **do-while loop** performs the loop body first to guarantee that it is run at least once. After each iteration, the condition is checked.

10. Create a complete Java program that shall allow the user to accept three integer values representing the START, END, and STEP respectively. The START should always be lesser than the END value, and the STEP is always greater than zero. The program shall print vertically the values starting from the START to the last value which can be equal to or lesser than the END incremented by the STEP value.

Example Output 1: if START = 1, END = 10, STEP = 2

```
Input START value    = 1
Input END value      = 10
Input STEP value     = 2

1
3
5
7
9

Do you want to try again (Y/N)? Y
```

```
Input START value    = -10
Input END value      = 20
Input STEP value     = 5

-10
-5
0
5
10
15
20

Do you want to try again (Y/N)? N
```

Example Output 2: if START = -10, END = 20, STEP = 5

10.1. Your program should automatically detect any errors in the initial input.

10.2. Your program should have an additional feature that asks the user whether the user wants TO TRY AGAIN. The Program will not terminate until the user inputs any character other than 'Y' or 'y'.

10.3. Implement the program using all looping constructs identified earlier.

10.4. Write your complete programs in the space provided.

WHILE LOOP:

```
/* Programmed by: Abenes, Enrico O.
   Program Title: WhileLoop.java
   Program Date: July 11, 2023*/

package intl.cc1;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

public class WhileLoop {
    public static void main(String[] args) {
        int start, end, step;
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

```
        while (tryAgain) {
            try {
                System.out.print("Input START value   = ");
                choice = basa.readLine();
                start = Integer.parseInt(choice);

                System.out.print("Input END value     = ");
                choice = basa.readLine();
                end = Integer.parseInt(choice);

                System.out.print("Input STEP value    = ");
                choice = basa.readLine();
                step = Integer.parseInt(choice);

                System.out.println();
                int i = start;
                while (i <= end) {
                    System.out.println(i);
                    i += step;
                }

                System.out.println();
                System.out.print("Do you want to try again (Y/N)? ");
                choice = basa.readLine().toUpperCase();
                tryAgain = choice.equals("Y");
            } catch (IOException | NumberFormatException e) {
                System.out.println("Error!");
                tryAgain = true;
            }
        }
    }
}
```

FOR LOOP:

```
/* Programmed by: Abenes, Enrico O.
   Program Title: ForLoop.java
   Program Date: July 11, 2023*/

package intl.cc1;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

public class ForLoop {
    public static void main(String[] args) {
        int start, end, step;
        String choice;
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

```
        BufferedReader basa = new BufferedReader(new
InputStreamReader(System.in));

        for (choice = "Y"; choice.equals("Y"); ) {
            try  {
                System.out.print("Input START value   = ");
                choice = basa.readLine();
                start = Integer.parseInt(choice);

                System.out.print("Input END value     = ");
                choice = basa.readLine();
                end = Integer.parseInt(choice);

                System.out.print("Input STEP value    = ");
                choice = basa.readLine();
                step = Integer.parseInt(choice);

                System.out.println();
                for (int i = start; i <= end; i+= step) {
                    System.out.println(i);
                }

                System.out.println();
                System.out.print("Do you want to try again (Y/N)? ");
                choice = basa.readLine().toUpperCase();
            } catch (IOException | NumberFormatException e) {
                System.out.println("Error!");
                choice = "Y";
            }
        }
    }
}
```

DO-WHILE LOOP:

```
/* Programmed by: Abenes, Enrico O.
   Program Title: DoWhileLoop.java
   Program Date: July 11, 2023*/

package intl.cc1;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

public class DoWhileLoop {
    public static void main(String[] args) {
        int start, end, step;
        String choice;
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

```
        BufferedReader basa = new BufferedReader(new
InputStreamReader(System.in));

        boolean tryAgain;
        do {
            try {
                System.out.print("Input START value   = ");
                choice = basa.readLine();
                start = Integer.parseInt(choice);

                System.out.print("Input END value     = ");
                choice = basa.readLine();
                end = Integer.parseInt(choice);

                System.out.print("Input STEP value    = ");
                choice = basa.readLine();
                step = Integer.parseInt(choice);

                System.out.println();
                int i = start;
                do {
                    System.out.println(i);
                    i += step;
                } while (i <= end);

                System.out.println();
                System.out.print("Do you want to try again (Y/N)? ");
                choice = basa.readLine().toUpperCase();
                tryAgain = choice.equals("Y");
            } catch (IOException | NumberFormatException e) {
                System.out.println("Error!");
                tryAgain = true;
            }
        } while (tryAgain);
    }
}
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

Rubrics:

Code Content	The source code submitted covers all the items specified in the activity and satisfactorily meets all these requirements. It also makes use of the specific features of Java specified in the activity.	20 pts
Code Function	The source code submitted works completely with no errors and provides the correct expected output when run.	20 pts
Code Syntax	The source code submitted has sound logic and follows proper syntax for Java, with no unnecessarily complicated code.	10 pts
Total		**100 pts**

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

NAME:	Abenes, Enrico O.	**DATE:**	July 11, 2023
COURSE:	CC1 – INTL 1	**SCHEDULE:**	1:30 pm – 5:20 pm MT

Activity Number: 2

Activity Type: Laboratory Activity

TITLE: One-Dimensional Array

LEARNING OBJECTIVES:

At the end of this activity, the students should be able to:

1. Declare a one-dimensional array in Java.

2. Initialize and reference elements in a one-dimensional array.

3. Perform basic operations available for a one-dimensional array.

4. Create a complete Java program that simulates the application of a one-dimensional array.

INSTRUCTIONS:

1. Make sure you have your own individual account.

2. Always keep your account secret from others to avoid unauthorized access to your files.

3. Always save your work and log off when not using the computer.

4. By now you should have been familiarized with using your text editor.

5. By now you should know how to create, save, compile, execute, and debug programs in Java.

DURATION: Two to Three Meetings

HANDS-ON:

1. Log on using your own individual account. Use your own username and password.

2. Open your text editor.

3. Write your next Java program:

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

3.1. Write your next program by copying the source code shown below to your text editor.

```
/* Programmed by: <write your name here>
   Program title: Count.java
   Program Date: <write the date today here> */

import java.io.*;
public class Count{
  public static void main(String[] args){
   int i, n, ctr;
   String input = " ";
   ctr = 0;

   BufferedReader in = new  BufferedReader(new
                           InputStreamReader(System.in));

   for(i = 1; i<=10; i++){
      System.out.print("Input integer number: ");
      try{
        input = in.readLine();
      }catch(IOException e){
        System.out.println("Error!");
      }

      n = Integer.parseInt(input);

      if(n >=0){
       ctr = ctr + 1;
       }
   }
   System.out.println("The total value for counter is " + ctr);
  }
}
```

3.2. Save your program as Count.java then compile your program until no errors and warnings are reported.

3.3. Run your program.

3.4. Simulate and write what will be displayed on the screen.

```
Input integer number: 32
Input integer number: 42
Input integer number: 32
Input integer number: 54
Input integer number: 34
Input integer number: 42
Input integer number: 42
Input integer number: 43
Input integer number: 65
Input integer number: 43
The total value for counter is 10
```

3.5. What do you think is being counted in the program?

The computer program is keeping track of how many positive integer inputs the user has provided. (quantity)

---

4. Revise your Count.java program and save it as Count2.java.

Thereafter, the user will be asked to input 10 integer numbers and display how many of these 10 numbers are even and how many are odd. Use only one variable for your counter.

4.1. Write your complete Java program here:

```
/* Programmed by: Abenes, Enrico O.
   Program Title: Count2.java
   Program Date: July 11, 2023*/

package intl.cc1;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

public class Count2 {
    public static void main(String[] args) {
        int i, n, ctr;
        String input = " ";
        ctr = 0;

        BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
```

```
        for (i = 1; i <= 10; i++) {
            System.out.print("Input integer number: ");
            try {
                input = in.readLine();
            } catch (IOException e) {
                System.out.println("Error!");
            }

            n = Integer.parseInt(input);

            if (n % 2 == 0) {
                ctr++;
            }
        }
        System.out.println("Number of even integers: " + ctr);
        System.out.println("Number of odd integers: " + (10 - ctr));
    }
}
```

4.3. Run your program.

4.4. Simulate and write what will be displayed on the screen.

```
Input integer number: 1
Input integer number: 2
Input integer number: 3
Input integer number: 6
Input integer number: 7
Input integer number: 9
Input integer number: 12
Input integer number: 13
Input integer number: 14
Input integer number: 17
Number of even integers: 4
Number of odd integers: 6
```

4.5. What do you think is the purpose of counters?

A **counter variable** is a variable that counts the instances at which a particular line of code is run.

5. Write your next program by copying the source code shown below to your text editor.

5.1. Create a new program and save it as Accum.java

```
/* Programmed by: <write your name here>
   Program title: Accum.java
   Program Date: <write the date today here> */

public class Accum{
  public static void main(String[] args){
      int i, n, sum = 0;
      String input = " ";
       BufferedReader in = new  BufferedReader(new
                           InputStreamReader(System.in));

      for(i = 0; i<10; i++){
        System.out.print("Input integer number: ");
        try{
          input = in.readLine();
        }catch(IOException e){
          System.out.println("Error!");
        }

        n = Integer.parseInt(input);
       sum = sum + n;
      }
      System.out.println("The sum of the integers is " + sum);
  }
}
```

5.2. Save and then compile your program until no errors and warnings are reported.

5.3. Run your program.

5.4. Simulate and write what will be displayed on the screen

```
Input integer number: 12
Input integer number: 54
Input integer number: 63
Input integer number: 45
Input integer number: 76
Input integer number: 98
Input integer number: 87
Input integer number: 65
Input integer number: 46
Input integer number: 74
The sum of the integers is 620
```

5.5. What do you think is being accumulated in your program?

The user-inputted sum of 10 integer numbers is being added up by the application. It asks for an integer ten times from the user, reads the input, and turns it into an integer. It displays the **user-inputted integers' combined total.**

6. Revise your Accum.java program and save it as Accum2.java.

Thereafter, the user will be asked to input 10 integer numbers and display the sum of all even numbers and the sum of all odd numbers.

6.1. Write your complete Java program here:

```
/* Programmed by: Abenes, Enrico O.
   Program Title: Accum2.java
   Program Date: July 10, 2023*/

package intl.cc1;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;
```

```
public class Accum2 {
    public static void main(String[] args) {
        int i, n, sum_even = 0, sum_odd = 0;
        String input = " ";

        BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));

        for (i = 0; i < 10; i++) {
            System.out.print("Input integer number: ");
            try {
                input = in.readLine();
            } catch (IOException e) {
                System.out.println("Error!");
            }

            n = Integer.parseInt(input);

            if (n % 2 == 0) {
                sum_even += n;
            } else {
                sum_odd += n;
            }
        }
        System.out.println("The sum of even integers is: " + sum_even);
        System.out.println("The sum of odd integers is: " + sum_odd);
    }
}
```

6.3. Run your program.

6.4. Simulate and write what will be displayed on the screen.

```
Input integer number: 54
Input integer number: 76
Input integer number: 54
Input integer number: 68
Input integer number: 97
Input integer number: 65
Input integer number: 43
Input integer number: 73
Input integer number: 67
Input integer number: 64
The sum of even integers is: 316
The sum of odd integers is: 345
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

6.5. What do you think is the purpose of accumulators?

The results of successive additions are partially stored in an **accumulator variable.** Usually, the value that is added comes from another variable.

7. Differentiate new counters from accumulators.

An **accumulator** is used to gather data and compute a cumulative value over numerous operations, whereas a **counter** is used to keep track of the quantity of occurrences or iterations.

Rubrics:

Code Content	The source code submitted covers all the items specified in the activity and satisfactorily meets all these requirements. It also makes use of the specific features of Java specified in the activity.	20 pts
Code Function	The source code submitted works completely with no errors and provides the correct expected output when run.	20 pts
Code Syntax	The source code submitted has sound logic and follows proper syntax for Java, with no unnecessarily complicated code.	10 pts
Total		**100 pts**

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

NAME:	Abenes, Enrico O.	**DATE:**	July 11, 2023
COURSE:	CC1 – INTL 1	**SCHEDULE:**	1:30 pm – 5:20 pm MT

Activity Number: 3

Activity Type: Laboratory Activity

TITLE: One-Dimensional Array

LEARNING OBJECTIVES:

At the end of this activity, the students should be able to:

1. Declare a one-dimensional array in Java.

2. Initialize and reference elements in a one-dimensional array.

3. Perform basic operations available for a one-dimensional array.

4. Create a complete Java program that simulates the application of a one-dimensional array.

INSTRUCTIONS:

1. Make sure you have your own individual account.

2. Always keep your account secret from others to avoid unauthorized access to your files.

3. Always save your work and log off when not using the computer.

4. By now you should have been familiarized with using your text editor.

5. By now you should know how to create, save, compile, execute, and debug programs in Java.

DURATION: Two to Three Meetings

HANDS-ON:

1. Log on using your own individual account. Use your own username and password.

2. Open your text editor.

3. Write your next Java program:

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

3.1. Write your next program by copying the source code shown below to your text editor.

```
/* Programmed by: <write your name here>
   Program title: List.java
   Program Date: <write the date today here> */

import java.io.*;
public class List{
  public static void main(String[] args){
    int list[] = new int[10];
    int i, num = 0;
    String input = " ";

    BufferedReader in = new BufferedReader(new
                            InputStreamReader(System.in));

    for(i = 0; i < 10; i++){
      list[i] = 0;
    }
    for(i = 0; i < 10; i++){
      System.out.print("Input value for list[" + i +
                                "] = ");
      try{
       input = in.readLine();
      }catch(IOException e){}
      num = Integer.parseInt(input);
      list[i] = num;
     }

     for(i = 0; i < 10; i++){
       System.out.println("list[" + i + "] = " +list[i]);
     }
  }
}
```

3.2. Save the program as List.java then compile your program until no errors and warnings are reported.

3.3. Run your program.

3.4. Simulate and write what will be displayed on the screen.

```
Input value for list [0) = 3
Input value for list [1) = 5
Input value for list [2) = 5
Input value for list [3) = 7
Input value for list [4) = 5
Input value for list [5) = 2
Input value for list [6) = 6
Input value for list [7) = 9
Input value for list [8) = 6
Input value for list [9) = 7
list [0] = 3
list [1] = 5
list [2] = 5
list [3] = 7
list [4] = 5
list [5] = 2
list [6] = 6
list [7] = 9
list [8] = 6
list [9] = 7
```

3.5. What do you think is the purpose of the first for-loop?

Before the user enters any specific values, the **first for-loop** initializes the array elements to 0, giving each element a starting point value.

3.6. What do you think is the purpose of the second for-loop?

The user can input values for each element of the list array using the **second for-loop**, giving the application user-specific data to work with afterwards.

3.7. What do you think is the purpose of the third for-loop?

The **third for-loop** is in charge of reporting the list array's values to the console so you can check the values the user supplied and make sure they were properly stored in the array.

3.8. How many elements can your list contain?

The fixed size of the list array in the provided code is 10, therefore it may hold **10 elements**. We can modify the array declaration's size to change how many elements the array can hold.

3.9. What index value is the first element located?

The first entry in the context of the list array in the code is at **index 0**.

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

3.10. Given the <size> of the list, at what index is the last element located?

The list's fixed length is 10 items. Consequently, **index 9** is where the final element can be found.

---

4. Create a new program and save it as List2.java. Thereafter, create 3 lists and name is as list1, list2, and list3, having 10 elements each. The user should input 10 integer values for list1 and 10 integer values for list2. Your program should add the contents of list1 and list2 then store the sum to list3. Your program should display horizontally the values of list1, list2, and list3. Use loops.

Example Output:

```
List1 : 1  3  2  5  7  8  5  6  9  4
List2 : 2  1  4  3  2  1  4  2  0  2
List3 : 3  4  6  8  9  9  9  8  9  6
```

Hint: use for -loops!
list3[0] = list1[0] + list2[0]
list3[1] = list1[1] + list2[1]
list3[2] = list1[2] + list2[2]
list3[3] = list1[3] + list2[3]
list3[4] = list1[4] + list2[4]
list3[5] = list1[5] + list2[5]
list3[6] = list1[6] + list2[6]
list3[7] = list1[7] + list2[7]
list3[8] = list1[8] + list2[8]
list3[9] = list1[9] + list2[9]

4.1. Write your complete Java program here:

```
/* Programmed by: Abenes, Enrico O.
   Program Title: List2.java
   Program Date: July 11, 2023*/

package intl.cc1;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

public class List2 {
    public static void main(String[] args) {
        String input = " ";
```

```
        int[] list1 = new int[10];
        int[] list2 = new int[10];
        int[] list3 = new int[10];

        BufferedReader basa = new BufferedReader(new
InputStreamReader(System.in));

        try {
            System.out.println("Enter Ten(10) Integer Values for
List1");
            for (int i = 0; i < 10; i++) {
                System.out.print("Input value for list1[" + i + "] =
");
                input = basa.readLine();
                list1[i] = Integer.parseInt(input);
            }

            System.out.println(" ");
            System.out.println("Enter Ten(10) Integer Values for
List2");
            for (int i = 0; i < 10; i++) {
                System.out.print("Input value for list2[" + i + "] =
");
                input = basa.readLine();
                list2[i] = Integer.parseInt(input);
            }

            System.out.print("List1 : ");
            for (int i = 0; i < 10; i++) {
                System.out.print(list1[i] + "  ");
            }
            System.out.println();

            System.out.print("List2 : ");
            for (int i = 0; i < 10; i++) {
                System.out.print(list2[i] + "  ");
            }
            System.out.println();

            for (int i = 0; i < 10; i++) {
                list3[i] = list1[i] + list2[i];
            }

            System.out.print("List3 : ");
            for (int i = 0; i < 10; i++) {
                System.out.print(list3[i] + "  ");
            }
            System.out.println();

        } catch (IOException e) {
            System.out.println("Error!");
        }
    }
```

4.2. Save then compile your program.

4.3. Run your program.

4.4. Simulate and write what will be displayed on the screen.

```
Enter Ten(10) Integer Values for List1
Input value for list1[0] = 1
Input value for list1[1] = 3
Input value for list1[2] = 4
Input value for list1[3] = 2
Input value for list1[4] = 1
Input value for list1[5] = 5
Input value for list1[6] = 4
Input value for list1[7] = 2
Input value for list1[8] = 4
Input value for list1[9] = 2

Enter Ten(10) Integer Values for List2
Input value for list2[0] = 5
Input value for list2[1] = 3
Input value for list2[2] = 2
Input value for list2[3] = 5
Input value for list2[4] = 1
Input value for list2[5] = 4
Input value for list2[6] = 3
Input value for list2[7] = 5
Input value for list2[8] = 2
Input value for list2[9] = 1
List1 : 1  3  4  2  1  5  4  2  4  2
List2 : 5  3  2  5  1  4  3  5  2  1
List3 : 6  6  6  7  2  9  7  7  6  3
```

5. Revise your List2.java program. Thereafter, your program should display the highest value in list3 and the lowest value in list3.

5.1. Write the additional codes here:

```
int mataas = list3[0];
int mababa = list3[0];

for (int i = 1; i < 10; i++) {
    if (list3[i] > mataas) {
        mataas = list3[i];
    }
    if (list3[i] < mababa) {
        mababa = list3[i];
    }
}
```

```
System.out.println("Highest value in List3: " + mataas);
System.out.println("Lowest value in List3: " + mababa);
```

6. Revise your List2.java program. Thereafter, your program should allow the user to input an integer value to be searched in list3. Your program should display whether the inputted integer value is found in list3, how many of them are in list3, and what are their locations in list3.

Example output:

```
List1 : 1  3  2  5  7  8  5  6  9  4
List2 : 2  1  4  3  2  1  4  2  0  2
List3 : 3  4  6  8  9  9  9  8  9  6

Input value to search in List3: 9

The value 9 is in List3!
There are 4 of it in List3.
Located at: list3[4], list3[5], list3[6], list3[8]
```

6.1. Save and then compile your program until no errors and warnings are reported.

6.2. Run your program.

6.3. Simulate and write what will be displayed on the screen.

```
Enter Ten(10) Integer Values for List1
Input value for list1[0] = 1
Input value for list1[1] = 4
Input value for list1[2] = 5
Input value for list1[3] = 3
Input value for list1[4] = 2
Input value for list1[5] = 1
Input value for list1[6] = 5
Input value for list1[7] = 4
Input value for list1[8] = 5
Input value for list1[9] = 3
```

```
Enter Ten(10) Integer Values for List2
Input value for list2[0] = 1
Input value for list2[1] = 4
Input value for list2[2] = 2
Input value for list2[3] = 5
Input value for list2[4] = 4
Input value for list2[5] = 2
Input value for list2[6] = 1
Input value for list2[7] = 4
Input value for list2[8] = 3
Input value for list2[9] = 4

List1 : 1  4  5  3  2  1  5  4  5  3
List2 : 1  4  2  5  4  2  1  4  3  4
List3 : 2  8  7  8  6  3  6  8  8  7

Input value to search in List3: 8

The value 8 is in List3!
There are 4 of it in List 3.
Located at: list3[1]. list3[3]. list3[7]. list3[8]
```

6.4. Write your complete source code here:

```
/* Programmed by: Abenes, Enrico O.
   Program Title: List2.java
   Program Date: July 11, 2023*/

package intl.cc1;

import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

public class List2 {
    public static void main(String[] args) {
        String search_input, input;
        boolean hanap = false;
        int search_value, ctr = 0;

        int[] list1 = new int[10];
        int[] list2 = new int[10];
        int[] list3 = new int[10];

        BufferedReader basa = new BufferedReader(new
InputStreamReader(System.in));
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

```
        try {
            System.out.println("Enter Ten(10) Integer Values for
List1");
            for (int i = 0; i < 10; i++) {
                System.out.print("Input value for list1[" + i + "] =
");
                input = basa.readLine();
                list1[i] = Integer.parseInt(input);
            }

            System.out.println(" ");
            System.out.println("Enter Ten(10) Integer Values for
List2");
            for (int i = 0; i < 10; i++) {
                System.out.print("Input value for list2[" + i + "] =
");
                input = basa.readLine();
                list2[i] = Integer.parseInt(input);
            }

            System.out.println(" ");
            System.out.print("List1 : ");
            for (int i = 0; i < 10; i++) {
                System.out.print(list1[i] + "  ");
            }
            System.out.println();

            System.out.print("List2 : ");
            for (int i = 0; i < 10; i++) {
                System.out.print(list2[i] + "  ");
            }
            System.out.println();

            for (int i = 0; i < 10; i++) {
                list3[i] = list1[i] + list2[i];
            }

            System.out.print("List3 : ");
            for (int i = 0; i < 10; i++) {
                System.out.print(list3[i] + "  ");
            }
            System.out.println();

            System.out.println(" ");
            System.out.print("Input value to search in List3: ");
            search_input = basa.readLine();
            search_value = Integer.parseInt(search_input);

            StringBuilder lokasyon = new StringBuilder();
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

```
            System.out.println(" ");
            for (int i = 0; i < 10; i++) {
                if (list3[i] == search_value) {
                    if (!hanap) {
                        System.out.println("The value " + search_value
+ " is in List3!");
                        hanap = true;
                    }
                    ctr++;
                    lokasyon.append("list3[").append(i).append("]. ");
                }
            }

            if (hanap) {
                System.out.println("There are " + ctr + " of it in List
3.");
                System.out.println("Located at: " +
lokasyon.substring(0, lokasyon.length() - 2));
            } else {
                System.out.println("The value " + search_value + " is
not found in List3");
            }

        } catch (IOException e) {
            System.out.println("Error!");
        }
    }
}
```

Rubrics:

Code Content	The source code submitted covers all the items specified in the activity and satisfactorily meets all these requirements. It also makes use of the specific features of Java specified in the activity.	20 pts
Code Function	The source code submitted works completely with no errors and provides the correct expected output when run.	20 pts
Code Syntax	The source code submitted has sound logic and follows proper syntax for Java, with no unnecessarily complicated code.	10 pts
Total		**100 pts**

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

NAME:	Abenes, Enrico O.	**DATE:**	July 11, 2023
COURSE:	CC1 – INTL 1	**SCHEDULE:**	1:30 pm – 5:20 pm MT

Activity Number: 4

Activity Type: Laboratory Activity

TITLE: Java Methods

LEARNING OBJECTIVES:

At the end of this activity, the students should be able to:

1. Identify the types of methods in the Java programming language.

2. Identify what portions of the main program can be subdivided into sub-programs.

3. Create methods or sub-programs that can be called in the main program.

4. Distinguish the different parameters used in creating methods.

5. Create a complete Java program that utilizes methods.

INSTRUCTIONS:

1. Make sure you have your own individual account.

2. Always keep your account secret from others to avoid unauthorized access to your files.

3. Always save your work and log off when not using the computer.

4. By now you should have been familiarized with using your text editor.

5. By now you should know how to create, save, compile, execute, and debug programs in Java.

DURATION: Two to Three Meetings

HANDS-ON:

1. Log on using your own individual account. Use your own username and password.

2. Open your text editor.

3. Write your next Java program:

    3.1. Write your next program by copying the source code shown below to your text editor.

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

```
/* Programmed by: <write your name here>
   Program title: PrintHello.java
   Program Date: <write the date today here> */

public class PrintHello{
     public static void printHello(){
          System.out.print("HELLO");
     }
     public static void printSpace(){
          System.out.print(" ");
     }
     public static void printWorld(){
          System.out.println("WORLD!");
     }
     public static void allTogetherNow(){
          printHello();
          printSpace();
          printWorld();
     }

     public static void main(String[] args){
          AllTogetherNow();
     }
}
```

3.2. Save the program as PrintHello.java then compile your program until no errors and warnings are reported.

3.3. Run your program.

3.4. Simulate and write what will be displayed on the screen.

```
Hello WORLD!
```

3.5. Enumerate all user-defined methods:

printHello() – "Hello" is printed to the console using this way.

printSpace() – This technique prints the character "space" to the console.

printWorld() – "WORLD!" is printed to the console by this way.

allTogetherNow() – The printHello(), printSpace(), and printWorld() methods are sequentially invoked by this method.

3.6. Enumerate all pre-defined methods:

System,out print() – Text can be printed on the console using this technique.

String[] args – This is the input for the main method, a built-in method that acts as the starting point for Java programs.

> ***HINT:*** To call a method, indicate the name of the method preceded by the **()** symbol. User-define methods are methods you created , while pre-defined functions are functions already inherent to the compiler.

4. Is the method parameter always void? No

```
public class OneParam{
     public static void oneParam(int x){
          System.out.println("Number is " + x);
     }

     public static void main(String[] args){
          int x;
          x = 100;
          oneParam(x);
     }
}
```

4.1 To answer that, let's simulate the following program

4.2 Save your program as OneParam.java

4.3 Run and write what will be displayed on the screen.

```
Number is 100
```

4.4. Insert the statement oneParam(75);

What can you conclude? That is, is it possible to use a constant value as a parameter when a method is called?

When you wish to send a method a certain value that is known at compile time and doesn't need to be dynamically computed during runtime, using constant values as method parameters **might be helpful.**

4.5 Declare another integer variable in your main method as follows:

int y; Initialize the variable y to 150. Thereafter, write another statement OneParam(y);

What can you conclude? That is, is it possible to use a variable name that is not the same as that used in the method definition?

Using a variable name other than the one given in the method description is technically **conceivable**. As long as the types of the arguments and parameters match, various variable names can be used to pass values when invoking a method.

4.6 Insert the statement OneParam(x+y);

What can you conclude? That is, is it possible to use an expression (that will evaluate an integer value) as a parameter when the method is called?

When invoking a method, it is **possible** to pass an expression that evaluates to an integer value as a parameter.

4.7 Insert the statement OneParam();

What can you conclude? That is, can you call a method requiring a parameter with a passing one? What error is reported if any?

A method that requires a parameter **cannot be called** without one being sent in. There is a **compilation problem** when OneParam() is called without any parameters.

5. Let us create another program and save it as TwoParams.java

5.1. Copy the source code

```
public class TwoParams{
  public static void twoParams(int x, int y){
     System.out.println("First parameter is " + x);
     System.out.println("Second parameter is " + y);
  }

  public static void main(String[] args){
     int x = 5;
     int y = 10;

     int a;
     int b;

     twoParams(10, 20);
     twoparams(x + 2, y * 10);

     a = -2;
     b = 22;
     twoParams(a, b);
  }
}
```

5.2. Save and then compile your program until no errors and warnings are reported.

5.3. Run your program.

5.4. Simulate and write what will be displayed on the screen.

```
First parameter is 10
Second parameter is 20
First parameter is 7
Second parameter is 100
First parameter is -2
Second parameter is 22
```

5.5. Insert the statement twoParams(y, x);

What can you conclude? Is the result the same as of twoParams(x,y)?

Both "First parameter is 10" and "Second parameter is 5" will be printed. TwoParams(x, y); prints "First parameter is 5" and "Second parameter is 10", however this outcome **is different.**

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

5.5.1 What will happen if we forgot to type the comma between parameter? For example, will twoParams(x y) work?

Java will encounter a **syntax error** as a result. The code won't successfully compile.

5.6. What will happen if we only supplied one parameter? For example, will twoParams(x) work? No, compilation error.

5.7. What will happen if we forgot all the parameters?

For example, will twoParams() work? No, compilation error.

5.8. What will happen if we used more than two parameters? For example, will twoParams(x, y, a) work? No, compilation error.

6. Are all methods of return type void? No

6.1. To answer that let's simulate the following program:

```
//Return value

public class WithReturnValue{
  public static int printSum(int x, int y){
     z = x + y;
     return z;
  }

  public static void main(String[] args){
     int a, b, c;

     System.out.println("Sum = " + printSum(5, 10));
     c = printSum(100, 300);
     System.out.println("c = " + c);

     a = 25;
     b = 75;

     System.out.println("Sum = " + printSum(a, b));
     System.out.println("Sum = " + Sum(a+2, b - 3));

  }
}
```

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

6.2. Save your program as WithReturnValue.java then compile and run and simulate your program. Write what will be displayed on the screen:

```
Compilation Error

Command execution failed.

BUILD FAILURE
```

6.3. What will happen if the statement return z; was omitted?

When the code attempts to delete the method's return value, a **compilation error** will still result.

6.4. Write a method named printDifference that accepts two integer parameters named x and y then computes the difference (x – y). Call this method inside the main().

```
/* Programmed by: Abenes, Enrico O.
   Program Title: Difference.java
   Program Date: July 11, 2023*/

package intl.cc1;

public class Difference {
    public static int printDifference(int x, int y) {
        int negatibo = x - y;
        return negatibo;
    }

    public static void main(String[] args) {
        int a = 10;
        int b = 5;

        int resulta = printDifference(a, b);
        System.out.println("Difference = " + resulta);
    }
}
```

6.5. Write a method named printProduct that accepts two double-type parameters named x and y. The method should compute and return the value of the product of x and y. Use double as return type. Call this method inside main();

UNIVERSITY OF THE CORDILLERAS
College of Information Technology and Computer Science
CC1 – Computing Fundamentals
Term & School Year: Third Trimester S.Y. 2022-2023
Faculty: Mailynn K. Sagayo

```
/* Programmed by: Abenes, Enrico O.
   Program Title: Product.java
   Program Date: July 11, 2023*/

package intl.cc1;

public class Product {
    public static double printProduct(double x, double y) {
        double producto = x * y;
        return producto;
    }

    public static void main(String[] args) {
        double a = 4.6;
        double b = 5.9;

        double resulta = printProduct(a, b);
        System.out.println("Product = " + resulta);
    }
}
```

Rubrics:

Code Content	The source code submitted covers all the items specified in the activity and satisfactorily meets all these requirements. It also makes use of the specific features of Java specified in the activity.	20 pts
Code Function	The source code submitted works completely with no errors and provides the correct expected output when run.	20 pts
Code Syntax	The source code submitted has sound logic and follows proper syntax for Java, with no unnecessarily complicated code.	10 pts
Total		**100 pts**