Prof. Dr.-Ing. Ralf Schenkel
Tobias Zeimetz
Trier University

Wintersemester 22/23

Exercises for the Class

Elements of Computer Science: Programming

Assignment 02

Submission of solutions until 3:30 p.m. at 14.11.2022
at `moodle.uni-trier.de`

- Every task needs to be edited in a meaningful way in order to get a point!
- Please comment your solutions, so that we can easy understand your ideas!
- If you have questions about programming or the homeworks, just ask you teachers!
- **Submission that can't be compiled are rated with 0 points!**

Exercise 1 (Evaluation: Numbers)

The task is essentially based on the fourth task of the first assignment:

(a) The given program reads a number (`size`) via the scanner andthen creates an array with the size of the inputted number
(b) At each index in the array, the array should be given the value of the square of the index
(c) The values in the array should then be printed in the console

Now modify (b) and (c) in this program as follows:

(b') Change the program so that as many numbers are stored in the array as the size of the array allows. The numbers should be read by the scanner and stored in the array.
(c') The output should take place in reverse order (i.e. the last input as the first output).

The input "`5 0 1 2 3 4`" therefore becomes "`4 3 2 1 0`". Remember, the first number (namely `5`) is only the size of the array and therefore is not part of the sequence. When evaluating, you will see further example sequences that your program must reproduce.

```java
import java.util.Scanner;
public class Inverse_Output {
  public static void main(String[] args) {
/* The task is essentially based on the fourth task of the first exercise: */
    Scanner sc = new Scanner(System.in);
    int index;
    int size;
    int[] array;
/* Part (a) */
```

```
    size = sc.nextInt();
    array = new int[size];

/* Part (b)  */
    index = 0;
    while ( index < size )
    {
      array[index] = index * index;
      index = index + 1;
    }

/* Part (c)  */
    index = 0;
    while ( index < size )
    {
      System.out.print( array[index] + "␣" );
      index = index + 1;
    }

/* The following brackets must be retained. */
  }
}
```

Exercise 2 (Evaluation: Numbers)

The exercise builds on the previous one:

- The size and then the corresponding values are to be read into a field again.
- The new part: After that another number (the variable `barrier`) is entered.
- Now all positions in the field are to be found (and output), at which a larger number than this barrier occurs.
- The input `5 22 10 41 35 27 30` should result in an output of `2 3`, since the values stored at the indexes 2 and 3 are 41 and 35, which are larger than the barrier (the last inputted number) `30`.
- To compare the numbers, you can use the following statement:

  ```
  if ( barrier < array[index] ) ...
  ```

  This line is analogous to line 22 in the example program `K2B06E_Arrays.java`.

```
import java.util.Scanner;

public class Exercise {
  public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);
    int index;
    int size;
    int[] array;
    int barrier;

/* The solution has to be typed in the following area: */




/* The following brackets must be retained. */
```

```
  }
}
```

Exercise 3 (Evaluation: Numbers)

The exercise is based on the previous one: As before, the program should read as first value the size of an array and afterwards create an array using the entered size. Next, the programm should read as many numbers as the size of the array allows via the scanner and store them in the created array. Your task will be to test **how often** the value `index` is in the field at the position `index`.

You can program the necessary test for equality as follows:

```
if ( array [index] == index ) ...
```

The input `5   2 1 4 3 2` would result in the output `2`, because at the two positions 1 and 3 in the field (starting with position 0) there are also the values 1 and 3.

```
import java.util.Scanner;

public class Exercise {
  public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);
    int index;
    int size;
    int[] array;
    int counter;

/* The solution has to be typed in the following area: */




/* The following brackets must be retained. */
  }
}
```

Exercise 4 (Evaluation: Numbers)

As before, the program should read as first value the size of an array and afterwards create an array using the entered size. Next, the programm should read as many numbers as the size of the array allows via the scanner and store them in the created array.

After that, you should start a "scavenger hunt" in the array, starting with position `index=0`: The value stored in the array at an index defines the next index that should be searched at. If, for example, you search at index 2, and the value stored there is 4, the next index you should search at is 4 (a more complete example is shown below). Use the setting `index = array[index}` to determine the next position to be considered. Output the first 10 indices that you search at in this manor

Example: If `5   2 4 3 1 0` is entered, your program should do as follows:

- array[0] -> 2, therefore the next field in the array to be considered is field 2;
- array[2] -> 3, next field is 3;
- array[3] -> 1, next field is 1;
- array[1] -> 4, next field is 4;
- array[4] -> 0, there we will go back to 0;
- array[0] -> 2, and so on...

from here on, the values found are repeated.

The correct output would then be (10 values!): `0 2 3 1 4 0 2 3 1 4`

```java
import java.util.Scanner;

public class Exercise {
  public static void main(String[] args) {

    Scanner sc = new Scanner(System.in);
    int index;
    int size;
    int[] array;
    int counter;

/* The solution has to be typed in the following area: */




/* The following brackets must be retained. */
  }
}
```