

Questions and Exercises

These questions and exercises is an opportunity to see what you've learnt from the lecture as well as practice the new things we've been talking about. In other words, these questions and exercises are completely optional but it's recommended to do them. In the end of the document you will find the answers to the questions as well as possible solutions to the exercises, note that one can solve an exercise in different ways. There will also be some suggestions about what one could code if one want to continue with some more advanced things. These suggestions will not come with a possible solution and might include things that haven't been covered in the lecture.

Question 1

How does one create an integer array variable with an array that is 5 elements long?

Question 2

The following code is supposed to print out the content of the array. What's wrong with it?

```
int[] myArray = {3, 1, 4};  
for (int e ; myArray) {  
    System.out.println(e);  
}
```

Question 3

The following code is storing a multidimensional array in an already declared variable. How many dimensions does the array have?

```
numbers = {{{5}}, {{3, 2}}, {2}}, {{-2, 0}, {2}, {0, 2}}};
```

Question 4

What would the following code print out on the screen? Answer the question before you run the code.

```
int[][] twoDimensionalArray = {{3, 1, 2}, {2}};  
int[] oneDimensionalArray = twoDimensionalArray[0];  
oneDimensionalArray[1] = 20;  
System.out.println(twoDimensionalArray[0][1]);
```

Exercise 1

Write a program that asks the user for 10 integers. Store them and sort them with the numbers sorted highest to lowest. Print the sorted values.

Exercise 2

Write a program where the user enters a username and a password. The username and password combination should be compared against some existing users. If the user entered a valid username

and password combination, the program should exit.

The program should also be able to read data from the command line arguments. If one argument is called "name" the following argument should be used as the name. If one argument is called "password" the next argument is going to be used as a password.

For instance if the program is run as "run Exercise2 name Vswe" then the program should just ask the user for the password. If we run it as "run Exercise2 password 1234 name Vswe" then the user shouldn't enter anything in the program itself.

Further explorations 1

Start with the Tic Tac Toe code from the lecture and improve it so it detect all winning conditions, print out the board even after the game is over and also allow the player to choose to play multiple games in a row. Also add so it keeps score of how many times the Xs and Os have won. The code from the lecture can be found here <https://dl.dropboxusercontent.com/u/46486053/TicTacToe.java>

Further explorations 2

Write a program that contains 5 different questions with 5 answers each (one of them being the correct one). The program should randomize which question to ask and supply the user with three of the five answers (1 being the correct one and 2 being wrong). Which of the wrong answers that are being used should also be randomized. The program should then wait for the user's response and then check if the answer from the user was correct.

The program should be written so it's easy to expand it for more questions.

Answers and solutions

Answer to Question 1

```
int[] myVariableName = new int[5];
```

The square brackets after the type creates an array variable of that type. The array itself is created with `new int[5]` since we wanted it to be 5 elements long.

Answer to Question 2

When using a for each loop one should use a colon between the element variable and the array, not a semi colon like in for loops. The code should look like the following

```
int[] myArray = {3, 1, 4};  
for (int e : myArray) {  
    System.out.println(e);  
}
```

Answer to Question 3

The array is a three dimensional array. This can easily be seen on the three opening curly brackets in the beginning or on the three closing curly brackets in the end. When creating multidimensional arrays with given values using curly brackets one simply adds the different dimensions inside each

other, that's why one can count the opening or closing brackets.

Answer to Question 4

The code will print 20. The reason is that when we get the one dimensional array from the two dimensional one we do NOT get a copy of the value, we get the actual array. Therefore when we change anything in the one dimensional one, we're changing a part of the two dimensional one at the same time, just because both the one dimensional array and the first element of the two dimensional one is the exact same array.

Possible solution to Exercise 1

```
import java.util.Scanner;

public class Exercise1 {
    public static void main(String[] args) {
        Scanner myScanner = new Scanner(System.in);

        int[] numbers = new int[10];
        System.out.println("Please enter 10 integers");
        //get the numbers from the user
        for (int i = 0; i < numbers.length; i++) {
            numbers[i] = myScanner.nextInt();
        }

        //perform a so called bubble sort
        for (int i = 0; i < numbers.length; i++) {
            for (int j = numbers.length - 1; j > i; j--) {
                if (numbers[j] > numbers[j - 1]) {
                    //if the values are in the wrong order, swap them
                    int temp = numbers[j];
                    numbers[j] = numbers[j - 1];
                    numbers[j - 1] = temp;
                }
            }
        }

        //print out the result
        for (int e : numbers) {
            System.out.println(e);
        }
    }
}
```

Possible solution to Exercise 2

```
import java.util.Scanner;

public class Exercise2 {
    public static void main(String[] args) {
        Scanner myScanner = new Scanner(System.in);

        String[][] users = {
            {"Vswe", "1234"},
            {"TestUser", "SillyPassword"},
            {"Bob", "Cart"}
        };

        String name = "";
        String password = "";

        boolean loggedIn = false;
        while (!loggedIn) {
            //Get the name or/and password from the command line arguments
            for (int i = 0; i < args.length - 1; i++) {
                if (args[i].equals("name")) {
                    name = args[i + 1];
                } else if (args[i].equals("password")) {
                    password = args[i + 1];
                }
            }

            //Get the name
            while (name.equals("")) {
                System.out.println("Please enter your username");
                name = myScanner.next();
            }

            //Get the password
            while (password.equals("")) {
                System.out.println("Please enter your password");
                password = myScanner.next();
            }

            //check if the user information is valid
            for (String[] user : users) {
                if (name.equals(user[0]) && password.equals(user[1])) {
                    loggedIn = true;
                    break;
                }
            }
        }
    }
}
```

```
//reset the values
if (!loggedIn) {
    System.out.println("Invalid username or password");
    name = "";
    password = "";
}
}
System.out.println("Welcome " + name);
}
```