Basic Programming Practicum
Jobsheet 9 Array 2

Name
Dicha Zelianivan Arkana

NIM
2241720002

Class
1i

Department
Information Technology

Study Program
D4 Informatics Engineering

1 Laboratory

1.1 Experiment 1: Declare, Initialize, and Display 2D Array

1. Create a new project
2. Create a new class, name it **Arr1**
3. Write the basic structure of the Java programming language which contains the `main()` function
4. Create an array of integer type named `number` with a row capacity of 2 elements and a column of 3 elements

   ```
   int[][] number = new int[2][3];
   ```

5. Fill in each element of the value array as follows:

   ```
   number[0][0] = 12;
   number[0][1] = 14;
   number[0][2] = 34;
   number[1][0] = 20;
   number[1][1] = 24;
   number[1][2] = 67;
   ```

6. Display all contents of the elements to the screen

   ```
   System.out.println(number[0][0] + " " + number[0][1] + " " + number[0][2]);
   System.out.println(number[1][0] + " " + number[1][1] + " " + number[1][2]);
   ```

7. Compile and run the program. Match the results of the running programs that you have created according to the following Display

```
12 14 34
20 24 67
```

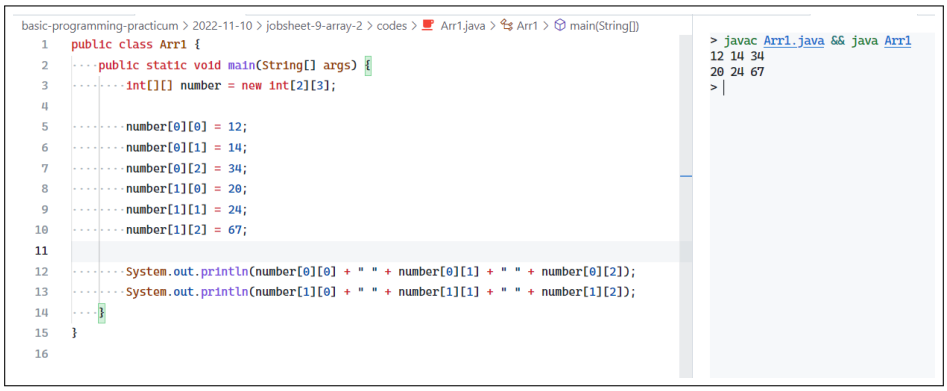

Figure 1: Experiment 1 code and output

Questions!

1. Should the array elements be filled sequentially? Explain!

   No. As long as the array has been initialised, we can fill it however we want. The order doesn't matter.

2. In step 5, modify the code so that the filled elements are only array elements in odd row positions! Can this be done? Prove it!

   Yes, we can do this as shown on figure 2

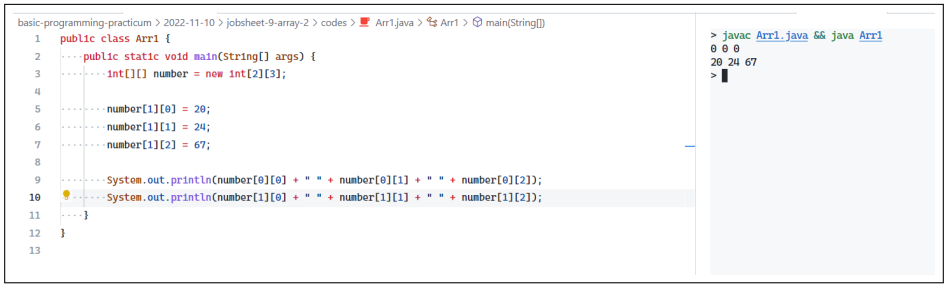

Figure 2: Array with odd elements

1.2 Experiment 2: Display 2 Dimensional Array Elements Using Loop

1. Create a new class, name it **Arr2**
2. Write the basic structure of the Java programming language which contains the `main()` function
3. Create an array of integer type named `number` with a row capacity of 2 elements and a column of 3 elements

   ```
   int[][] number = new int[2][3];
   ```

4. Fill in each element of the value array as follows:

   ```
   number[0][0] = 12;
   number[0][1] = 14;
   number[0][2] = 34;
   number[1][0] = 20;
   number[1][1] = 24;
   number[1][2] = 67;
   ```

5. Using a loop, display all the contents of the elements from the `number` array

   ```
   for (int i = 0; i < 2; i++) {
       for (int j = 0; j < 3; j++) {
           System.out.print(number[i][j] + " ");
       }
       System.out.println("");
   }
   ```

6. Compile and run the program. Match the results of the running programs that you have created according to the following Display

```
12 14 34
20 24 67
```

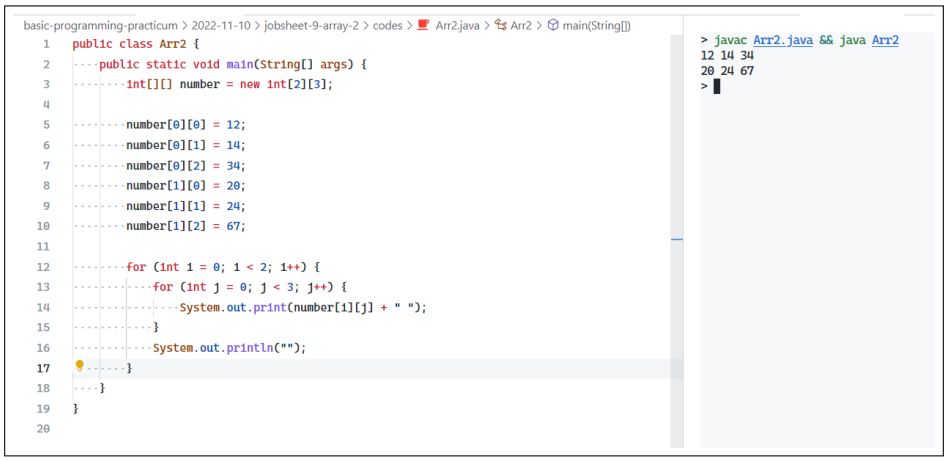

Figure 3: Experiment 2 code and output

Questions!

1. How many columns was the array in Experiment 2? Change the number of columns to 4 so that the array declaration and instantiation looks like the following code

   ```
   int[][] number = new int[2][4];
   ```

   Then, fill in the array elements with any value, corresponding to the addition of these columns. Run the program again, what happened?

   - There were 3 columns as shown by `int[2][3]`
   - The output still shows 2 rows and 3 columns (shown by figure 4) because we changed the number of rows but we didn't change the limit for the loop.

Figure 4: Experiment 2 with 2 rows and 4 columns

2. In step 5, change the program code as follows

   ```
   for (int i = 0; i < number.length; i++) {
       for (int j = 0; j < number[0].length; j++) {
           System.out.print(number[i][j] + " ");
       }
       System.out.println("");
   }
   ```

   Run the program after the change, what happened?

   The output will be the same. The reason is because we changed the hardcoded value to the length of the array. If we were to change the length of the array, we

no longer need to update the loop because it will automatically gets the length of the array whether if it's 2, 3, 4, so on and so forth.

3. Regarding displaying all array elements, change the program code to display array elements as follows

```
for (int array[] : number) {
    for (int r : array) {
        System.out.print(r + " ");
    }
    System.out.println("");
}
```

Run the results of these changes, what happened?

The output is the same because we only changed the `for` loop into a `for-each` loop. In this case, they do the exact same thing. It's just an alternative way of doing what we did before using `for` loop.

1.3 Experiment 3: Filling in 2 Dimensional Array Elements via Keyboard

1. Create a new class, name it **Arr3**
2. Write the basic structure of the Java programming language which contains the `main()` function
3. Add the Scanner library
4. Make a **Scanner** declaration with the name `input`
5. Create an array of integer type named `number` with a row capacity of 2 elements and a column of 3 elements

```
int[][] number = new int[2][3];
```

6. Using a loop, create an input to fill in the `number` array element

```
for (int i = 0; i < number.length; i++) {
    for (int j = 0; j < number[0].length; j++) {
        System.out.print("Enter a number [" + i + "][" + j + "]: ");
        number[i][j] = input.nextInt();
    }
    System.out.println("---------------");
}
```

7. Using a loop, display all the contents of the elements from the `number` array

```
for (int i = 0; i < number.length; i++) {
    for (int j = 0; j < number[0].length; j++) {
        System.out.print(number[i][j] + " ");
    }
    System.out.println("");
}
```

8. Compile and run the program. Match the results of the running programs that you have created according to the following Display

```
Enter a number [0][0]: 7
Enter a number [0][1]: 3
Enter a number [0][2]: 9
---------------
Enter a number [1][0]: 11
Enter a number [1][1]: 4
Enter a number [1][2]: 2
---------------
7 3 9
11 4 2
```

Figure 5: Experiment 3 code and output

Questions!

1. In step 6 can position j be replaced with position i? Explain!

   Yes, it can, but the result will be flipped. The row becomes the column and vice versa.

2. Add program code to determine the number of rows and columns of array elements dynamically (rows and columsn are determined when the program runs through the keyboard)!

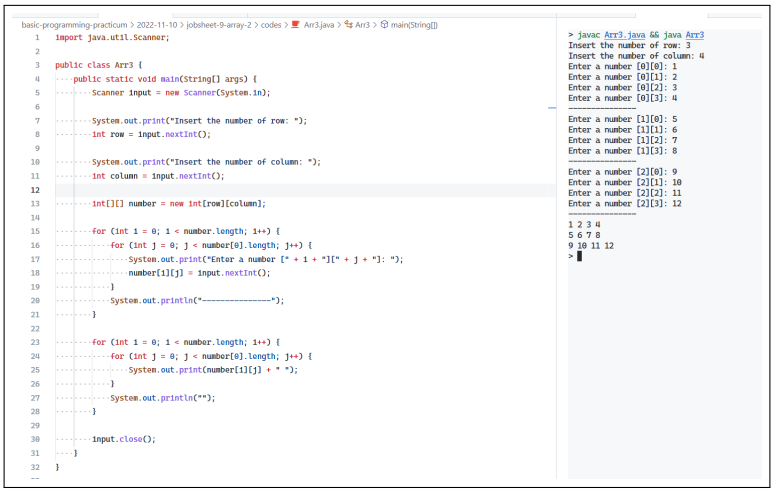

Figure 6: Experiment 3 with dynamic rows and columns from the user input

3. Modify the program code to display array elements using `foreach`!

Figure 7: Experiment 3 using foreach instead of for-loop

2 Assignment

1. Create a program that has two arrays as follows:

   - The first array is a one-dimensional array `char code[10]`, containing the license plate codes
   - The second array is a two-dimensional array containing the city names which are paired with the license plate codes

   The illustration of the array display is as follows

A	B	A	N	T	E	N				
B	J	A	K	A	R	T	A			
D	B	A	N	D	U	N	G			
E	C	I	R	E	B	O	N			
F	B	O	G	O	R					
G	P	E	K	A	L	O	N	G	A	N
H	S	E	M	A	R	A	N	G		
L	S	U	R	A	B	A	Y	A		
N	M	A	L	A	N	G				
T	T	E	G	A	L					

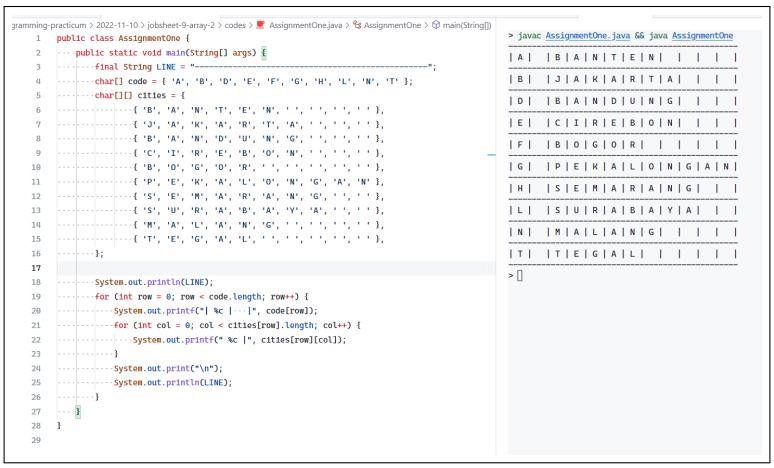

Figure 8: Assignment 1 code and output

2. Create a program containing a two-dimensional array having the row and column sizes obtained from keyboard input. Then, make input to fill the array elements. Next, make a menu choice that consists of:

   a. **MIN Value**. Display the value of the smallest array element to the screen

   b. **MIN Value & Amount**. Display to the screen the smallest value and how many the smallest value is, and also display the row and column location of the minimum value.

   c. **Array conditions**. Display the word "FOUND" on the screen if there is a value of 50 between the two-dimensional array elements, otherwise print the word "NOT FOUND".

Code

```
import java.util.Scanner;

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

        System.out.print("Insert the number of row: ");
        int row = input.nextInt();
        System.out.print("Insert the number of column: ");
        int col = input.nextInt();
        int[][] numbers = new int[row][col];

        // store the variable ahead of time so we don't need to calculate it on demand
        int minValue = Integer.MAX_VALUE;
        int minValueAmount = 0;
        String minValuePosition = "";
        final int MAX_NUM = 50;
        boolean isBigNumberFound = false;

        for (int r = 0; r < numbers.length; r++) {
            for (int c = 0; c < numbers[r].length; c++) {
                System.out.printf("Insert the number for row %d and column %d: ", r, c);
                int inputValue = input.nextInt();
                numbers[r][c] = inputValue;

                if (inputValue <= minValue) {
                    // reset since the max number changed
                    if (inputValue != minValue) {
                        minValueAmount = 0;
                        minValuePosition = "";
                    }

                    minValue = inputValue;
                    if (inputValue == minValue) {
```

```
                        minValuePosition += String.format(
                            "%d -> [row: %d, col: %d]\n",
                            minValueAmount + 1, r, c
                        );
                        minValueAmount++;
                    }
                }

                if (inputValue > MAX_NUM) {
                    isBigNumberFound = true;
                }
            }
        }

        int chosenMenu;
        while (true) {
            System.out.println("Menu:");
            System.out.println("1. Display MIN Value");
            System.out.println("2. Display MIN Value & Amount");
            System.out.println("3. Array conditions");
            System.out.print("Choose which menu to open (1-3): ");
            chosenMenu = input.nextInt();
            if (chosenMenu >= 1 && chosenMenu <= 3) break;
            System.out.println("Please insert the menu number correctly!");
        }

        switch (chosenMenu) {
            case 1:
                System.out.printf("The MIN value is: %d\n", minValue);
                break;
            case 2:
                System.out.printf("The MIN value is: %d\n", minValue);
                System.out.printf("The MIN value amount is: %d\n", minValueAmount);
                System.out.printf("The MIN value position is: \n%s\n", minValuePosition);
                break;
            case 3:
                System.out.println(isBigNumberFound ? "FOUND" : "NOT FOUND");
                break;
        }

        input.close();
    }
}
```

Output

```
> javac AssignmentTwo.java && java AssignmentTwo
Insert the number of row: 2
Insert the number of column: 2
Insert the number for row 0 and column 0: 4
Insert the number for row 0 and column 1: 48
Insert the number for row 1 and column 0: 4
Insert the number for row 1 and column 1: 20
Menu:
1. Display MIN Value
2. Display MIN Value & Amount
3. Array conditions
Choose which menu to open (1-3): 1
The MIN value is: 4
Do you want to choose another menu? (y/n): y
Menu:
1. Display MIN Value
2. Display MIN Value & Amount
3. Array conditions
Choose which menu to open (1-3): 2
The MIN value is: 4
The MIN value amount is: 2
The MIN value position is:
1 -> [row: 0, col: 0]
2 -> [row: 1, col: 0]

Do you want to choose another menu? (y/n): y
Menu:
1. Display MIN Value
2. Display MIN Value & Amount
3. Array conditions
Choose which menu to open (1-3): 3
NOT FOUND
Do you want to choose another menu? (y/n): n
>
```

Figure 9: Assignment 2 output