Contents

One-Dimensional Arrays

Solutions for those exercises.[1]

Multiple Choices

1. What is the correct way of creating an array of `int` of size 5 named `myArray`?

   - ☐ `int[] myArray = new int[5];`
   - ☐ `int[] myArray = int[5];`
   - ☐ `int[5] myArray = new int[];`
   - ☐ `int[4] myArray = new int[];`
   - ☐ `int myArray = new int[5];`
   - ☐ `int[] myArray = new int[4];`
   - ☐ `int[] myArray = new int(5);`
   - ☐ `int[] myArray = int[4];`

2. Consider the following code:

   ```
   int[] grades = {10, 20, 5, 15};
   Console.WriteLine(grades[2]);
   ```

   What will it display?

   - ☐ 5
   - ☐ Nothing
   - ☐ 20
   - ☐ 15
   - ☐ grades
   - ☐ grades(2)

[1] https:/princomp.github.io/exercises_w_sol/collections/1darrays

- ☐ 10

3. In the following statement, the value 5 is called the…

```
string[] names = new string[5];
```

   - ☐ allocation limit
   - ☐ size declarator
   - ☐ length
   - ☐ upper bound

4. Each element in an array is assigned a unique number called the…

   - ☐ subscript
   - ☐ index
   - ☐ position
   - ☐ rank

5. Consider the following code:

```
char[] grades = {'A', 'B', 'C', 'D', 'F'};
int i = 0;
while(i < grades.Length){
    i++;
    Console.WriteLine(grades[i]);
}
```

   Something is wrong with it, can you tell what?

   - ☐ There will be an "Index was outside the bounds of the array." error.
   - ☐ The array is not properly initialized.
   - ☐ The loop is infinite
   - ☐ `grades.Length` is not declared.

Exercises

Syntax

1. Write a statement that creates a 10-element `int` array named `numbers`.
2. Write a statement that creates and initializes an array of `double` with the values 12.5, 89.0 and 3.24.
3. In the following, what is the value of the size declarator? What is the value of the index?

```
int[] numbers;
numbers = new int[8];
numbers[4] = 9;
```

4. What is "array bounds checking"? When does it happen?
5. Is there an error with the following code? If you think there is one, explain it, otherwise draw the content of the `myIncomes` array once those statements have been executed.

```
double[] myIncomes = new double[5];
myIncomes[1] = 3.5;
// No income on day two.
myIncomes[3] = 5.8;
myIncomes[4] = 0.5;
myIncomes[5] = 1.5;
```

6. What would be the size of the `test` array after the following statement has been executed?

```
int[] test = {3, 5, 7, 0, 9};
```

7. What is wrong with the following array declaration?

```
int[] books = new int[-1];
```

8. Suppose we have an array named `temp` that has been declared and initialized. How can we know the number of elements in this array?
9. What is the value of `count` and the content of `number` once the following has been executed?

```
int count=2;
int[] number={3, 5, 7};
number[count--] = 8;
number[count]--;
```

10. Describe what the following code would do.

```
int[] record = { 3, 8, 11 };
int accumulator = 0;
foreach (int i in record)
    accumulator += i;
```

Displaying Arrays

1. Write code that displays the first and last element of an array.
2. What will be displayed at the screen by the following code?

```
int[] values = new int[6];
for (int i = 0 ; i < 6 ; i++)
    values[i] = (i * 2 );
```

```
foreach (int j in values)
    Console.WriteLine(j);
```

3. Suppose we are given an `int` array `dailyPushUp` with 7 elements. Write a piece of code that displays the value of the elements stored in the array `dailyPushUp`.
4. Write code that displays every element in a `numbers` array of integers.
5. Write code that displays every element in a `numbers` array of integers in reverse order.
6. Write code that displays every other element in a `numbers` array of integers (that is, every even index).
7. Given an array `numbers` and a variable `x`, write code that displays every value in `myArray` that is equal to or greater than `x`.
8. Write code that displays every unique value of *a sorted* array `numbers`. This array could contain, for example, 1, 1, 1, 4, 4, 5, 8, 9, 11: values are increasing and can occur multiple times. In our example, the values 1, 4, 5, 8, 9 and 11.

Filling Arrays

1. Given an array `myArray` and some value `x`, write code that sets the value of every array element to `x`.
2. Given some positive number `n`, write code that first declares an array of length `n`, then sets its contents to sequentially increasing values 1, 2, 3, ..., n.
3. Given an array of integers, and two integer variables `oldValue` and `newValue`; write code that replaces every occurrence of `oldValue` in the array with `newValue`.
4. Write code that squares every value in an `myArray` integer array. For example, an array containing 2, 3, 4 would after the program contain 4, 9, 16.

Looking For Values

1. Given an array `myArray` and a value `val`, write code that checks if `myArray` contains `val`. The result should be **`true`** if `val` occur in `myArray` and **`false`** otherwise.
2. Given an array `myArray` and a variable `x`, write code that computes the number of times `x` occurs in `myArray`.

3. Given an array `myArray` and two values `x` and `y`, write code that checks if `myArray` contains *either* `x` or `y`. The result should be **`true`** if `x` or `y` occur in `myArray` and **`false`** otherwise.
4. Given an array `myArray` and two values `x` and `y`, write code that checks if `myArray` contains *both* values `x` and `y`. The result should be **`true`** when both values occur and **`false`** otherwise.
5. Given an integer `myArray` and a strictly positive integer value `x`, find an array element whose value is largest while also being strictly less than `x` and display it, or display 0 if there is no such value. For example, in an array containing 1, 2, 6, 7, 3, 9 with x being 8, the solution is 7.
6. Consider an array of `char`. Implement code to check if the array values form a palindrome, i.e., it reads the same way forwards and backwards.

Manipulating Two Arrays

1. Assuming we have two `int` arrays of the same size, `firstA` and `secondA`, write a program that copies the content of `firstA` into `secondA`.
2. Given two arrays `array1` and `array2`, write a program to determine if there exists a value that occurs in both arrays. If such value exists, the result should be **`true`** and **`false`** otherwise.
3. Write a program that combines two `string` arrays called `array1` and `array2` into a single array containing first all the elements from `array1`, then all the elements from `array2`.
4. Given two arrays `arrayA` and `arrayB`, write code to check if every element in `arrayB` occurs in `arrayA`, then display the result as **`true`** or **`false`**.

Methods

1. Write a static method (header included) that takes as an argument an `int` array, and displays on the screen the value of each element of that array.
2. Write a static method (header included) that takes as an argument an `int` array, and stores the value 10 in each element of that array.

Simple Algorithms

1. Write a program that computes the sum of values stored in a `numbers` array of integers and displays it.

2. Given an array of positive integers, count how many even values occur in that array.
3. Write a program that computes the average of the elements in a `arrayP` numeric array.
4. Write a program that finds the largest value in an integer array `arrayP`.
5. Write a program that finds the smallest value in an integer array `arrayP`.
6. Write a program that finds the *second* smallest value in an array of integers `arrayP`.
7. Write code that finds the index of the first occurrence of a value `val` in an array `arrayP`. If the array does not contain the value, the result should be -1.
8. Write code that finds the index of the last occurrence of a value in an array. If the array does not contain the value, the result should be -1.
9. Write code to reverse the contents of an array `myArray`. For example, an array containing 1, 4, 3, 2, 5 should contain, after the program was executed, 5, 2, 3, 4, 1.

Wrap-Up Problems

1. Declare and initialize three arrays:
   - Choose different data type for each array,
   - Make the arrays have different lengths: 3, 5, 10 elements respectively,
   - Initialize each array with appropriate values of your choice (depends on the type).

   After you have declared and initialized the arrays, display on the screen
   - The first value from array 1 (0th index),
   - The last value from array 2 (4th index),
   - The first three values from array 3 (indexed 0 - 2).
2. Consider this array of words:

```
string[] words = {"voice", "effect", "day", "orange",
↪  "appliance", "fly", "cloud", "degree", "engine",
↪  "society"};
```

Write code to display an answer to the following questions. If the solution requires looping, use **foreach** loop when possible. Otherwise, use a **for** loop.

(a) Does words array contain "engine"? (true/false)
(b) Does words array contain "day" at least 2 times? (true/false)
(c) What is the position (index) of the word "society"? If it does not exist answer should be -1. Find the position of the first occurrence in case there are multiple matches.

After you have implemented your code, change the array contents and make sure your code still works and does not crash.

Here is another array of words to test your solution:

```
string[] words = {"addition", "day", "start", "dock",
 ↪  "fowl", "fish", "seat", "day"};
```