LABWORK RULES

- Labwork ends at 18.50.
- **Name of your file should be in the format**: **labwork3_name_surname.c**
    - Name: is your own name
    - Surname: is your own surname
    - Example: labwork1_name_surname.c
    - If you do not know how to create a c file, please look at “how to run your code” file
- **Your file should be on the desktop**
- **At 18.50**, we collect your works from your desktop. Do not turn off your computer until you are told to do so...
- You will not have an internet connection until we collect your works. When the access is permitted, submit your work on coadsys.
- The work we collect and you submitted on coadsys should be exactly the same.
- Read your labwork documents carefully and listen to the explanation of your assistant.
- There can be restrictions on your labwork, obey them if you want to get a full grade.
- If you did not understand the labwork or there is an unclear point, do not make assumptions and ask your assistant immediately.
- There will be some example input and output for each labwork. Do not write a program that only runs for the given example input. Be sure you understand the question well.

FORBIDDEN

- Talking with your friends
- Looking at your friend’s monitor
- Using any mobile phone or smart devices
- Books, notebooks, papers or notes
- Usb or any kind of device that is used to share information
- Internet

LABWORK 3

In the computing field, one of essential topics is the compression of data. There are different algorithms to compress different data types like images, words and binary files. In this labwork, you will compress an array of integers. The algorithm finds the numbers repeated consecutively and compresses it by writing the count and value. As it is seen from the example below, There are three consecutive twos, thus we write to the new array 3 and 2. Similarly there are five consecutive threes, so we write 5 and 3 to the array. To indicate the end, we put -1 after the compression is done.

Original data array

2	2	2	3	3	3	3	3

Compressed array

3	2	5	3	-1
count of value	value	count of value	value	termination symbol

1. **Function 1:**
   - **Name of the function:** compressArray
   - **Return type of the function:** void
   - **1. Parameter:** **a pointer to the integer array** to be compressed
   - **2. Parameter:** the size of the array **(integer)**
   - **3. Parameter:** **an array** that stores the compress data
   - Use **pointer arithmetic** to navigate in the first array. Thus, you will compress the array in the first parameter and write it to the array in the third parameter. Do not forget to put a -1 to the end of the compress array.

2. **Function 2:**
   - **Name of the function:** findSize
   - **Return type of the function:** void
   - **1. Parameter:** a compressed array **(integer)**
   - **2. Parameter:** the max capacity of the compressed array **(integer)**
   - **3. Parameter:** the actual size of the data used in the array. **(integer pointer)**

- Since it is possible that all the capacity of the array is not used. Thus, you need to find how much space is used in the array and assign it to the variable passed as the third parameter. Do not forget that -1 is used to show the end of the compressed data.

3. **Main function:**
   - Write a main function to test your code, you do not need to take values from the user. You can manually create your array.
   - Be careful, it is possible to create a compressed data which has a larger size than the original array. Declare size carefully.

Example Output:

```
3 3 3 0 0 0 0 0 0 0 0 2 2 2 2 3 4 4 4 5 5 5 5 5 5 9 9 7 8 8 8 8 8 8 8
3 3 8 0 4 2 1 3 3 4 6 5 2 9 1 7 7 8
the size is reduced from 35 to 18
```