Assignment 2 - Template Functions and Classes

- The problems of this assignment must be solved in C++.
- The TAs are grading solutions to the problems according to the following criteria: https://grader.eecs.jacobs-university.de/courses/320143/2018_1r2/Grading-Criteria-C++.pdf

Problem 2.1 *Template array search* (1 point)

Presence assignment, due by 18:30 h today

First write a generic `... array_search(...)` function using templates. The array search function should receive an array of elements, the corresponding number of elements, the element searched for and should determine if that element is in the array or not by returning its index or -1 if not in the array. Then write a test program with a `main` function which uses the function above for multiple types (some example basic data types and `Complex` objects).
You can assume that the array will contain at least one element. Make sure that your code works not only with basic data types but also with a simple `Complex` class.

Problem 2.2 *Stack with templates I* (2 points)

Design and implement a generic stack class. By using templates your stack has to be able to store any type of data. The underlying data structure should be an array.
While designing and implementing your class you cannot use any of the container classes from the Standard Template Library (STL).
Your class should support the following operations (read all the requirements before starting to write code):

- `Stack()`: this constructor initializes the stack to a size of 10.
- `Stack(const Stack&)`: copy constructor, create a real copy of the stack.
- `Stack(int size)`: this constructor sets the stack's size to the given size.
- `bool push(T element)`: adds `element` to the top of the stack. It should return `true`, if the element has been successfully pushed. So as long as you do not provide an `extend()` method, you need to check for available space and return `false` if there is no more space.
- `bool pop(T& out)`: pops an element from the top of the stack. The element is put into `out`. It should not crash if there are no elements on the stack, but rather return `false`, otherwise it should return `true`.
- `T back(void)`: returns the data on the top of the stack, without changing the stack.
- `int getNumEntries()`: returns the number of entries of the stack at a given moment.
- Destructor for the template class.

Finally, write a simple program which tests the functionality of your stack.
Name the files `Stack.h` and `testStack.cpp`.
You can assume that the input will be valid.

Problem 2.3 *Stack with templates II* (2 points)

Continue with your previous code by adding the following methods:

- `void resize(int size)`: resizes the stack. If `size` is smaller than the number of current entries, then the older elements (in terms of adding) should be lost.
- `int getSize()`: returns the size (the maximum amount of entries) of the stack.

Name the files `Stack.h` and `testStack.cpp`.
You can assume that the input will be valid.

Bonus Problem 2.4 *List with templates as doubly linked list* (2 points)

Implement a generic list using templates using a doubly linked list instead of an array. Provide constructors, destructor and methods for returning (with and without remove) the first and last element as well as adding a new element at the front and at the end of the list. Also implement a method which returns the amount of elements in the list.
Name the files `LinkedList.h` and `testLinkedList.cpp`.
You can assume that the input will be valid.

How to submit your solutions

- Your source code should be properly indented and compile with `g++` without any warnings (You can use `g++ -Wall -o program program.cpp`). Insert suitable comments (not on every line ...) to explain what your program does.
- Please name the programs according to the suggested filenames (they should match the description of the problem) in Grader.
  Each program **must** include a comment on the top like the following:

```
/*
   CH08-320143
   a2_p1.cpp
   Firstname Lastname
   myemail@jacobs-university.de
*/
```

- You have to submit your solutions via *Grader* at
  `https://grader.eecs.jacobs-university.de`.
  If there are problems (but **only** then) you can submit the programs by sending mail to `k.lipskoch@jacobs-university.de` **with a subject line that begins with CH08-320143**.
  It is important that you do begin your subject with the coursenumber, otherwise I might have problems to identify your submission.
- Please note, that after the deadline it will not be possible to submit any solutions. It is useless to send late solutions by mail, because they will not be accepted.

This assignment is due by Wednesday, March 7th, 10:00 h.