***Computer Lab #5.***

# Dynamic Class Attributes in C++

*In computer science, a SET is an abstract data structure that can store certain values, without any particular order, and no repeated values. It is a computer implementation of the mathematical concept of a finite set. Some set data structures are designed for static sets that do not change with time, and allow only query operations — such as checking whether a given value is in the set, or enumerating the values in some arbitrary order. Other variants, called dynamic or mutable sets, allow also the insertion and/or deletion of elements from the set.*

**Task 1.**

Download the source code from the Course Webpage: open the archive and copy the ***Set.h*** and ***SetMain.cpp*** files to your C/C++ development environment. Create the missing ***Set.cpp*** source file, add it to your project and implement a FIFO set structure. Follow the comments and instructions in the code.

a) The implementations in ***Set.cpp*** source file should be based on declarations (see the ***Set.h*** header file).

b) Run the test cases in ***SetMain.cpp***. Observe the program execution in debugger mode. Make sure that the execution result is correct.

c) Add an *empty()* method to the set class, which task is to clear the set (delete all elements). What is the difference between *empty()* function and a class destructor?

Decomposition of the program:

- *Set.h* file: class declaration header;
- *Set.cpp*: implementation of methods declared in the header;
- *SetMain.cpp*: the main program with testing code.

---

**Homework.**

*FIFO is an acronym for First In First Out, an abstraction in ways of organizing and manipulation of data relative to time and prioritization. In computer science this term refers to the way data stored in a queue is processed. Each item in the queue is stored in a queue data structure. The first data to be added to the queue will be the first data to be removed, then processing proceeds sequentially in the same order.*

Develop and implement a *FIFO* class. Use the *Set* class from Task 1 as an example. Consider the following mandatory functions and restrictions:

a) A *push* operation inserts a new element to FIFO, and a *pop* operation returns the firstly entered element.

b) Follow the principles of encapsulation: do not allow making the container inconsistent from the outside.
c) Dynamic memory management is to be used.
d) Based on the tests that can be found in SetMain.cpp, create test cases for FIFO class.

Decomposition of the program:

- *Fifo.h* file: class declaration header;
- *Fifo.cpp*: implementation of methods declared in the header;
- *FifoMain.cpp*: the main program with testing code.

---