Class Staque (Documentation)

Author: Nick Robaqidze.

email: [nika42568@gmail.com](mailto:nika42568@gmail.com)

mob: +995 598 19 07 03

Libraries and their usage:

* iostream - For printing data on screen.
* Vector or other associative and readable containers library. To get many data together.

When creating a class by default the it's pointers are assigned empty values (nullptr) by this way program does not automatically enter an invalid memory address (from buffer).

Private functions of the class:

* Class Node, in which the class Staque will act as a dynamic memory for data. It, as a regular class, has its own private variables that can only be accessed from the Staque class:
  + Data, save the data directly here (integer of type int).
  + Next, points to (saves the memory address) to the following Node.
  + previouslyAdded, points to (stores memory address) Node previously added to Staque.
* Node\* begin pointer that stores the memory address of the first element of the Staque, or the first Node.
* Node\* end pointer that stores the memory address of the last element of the Staque, or the last Node.
* Node\* last pointer, which stores the memory address of the element added at the end of the Staque, or Node.
* int size The number of elements in the Staque is stored here.

Public access functions of the class:

* The class constructor is overloaded:
  + A non-parametric constructor that performs nothing by default. Graphical user interface, text, application

    Description automatically generated
  + A constructor to which only one parameter is passed, another Staque class object. It creates a Staque consisting of all the elements that are included in the transmitted object.

Graphical user interface, text, application, chat or text message

Description automatically generated

* + A constructor to which only one parameter is passed, a container consisting of integers. It creates a Staque consisting of all the elements that are included in the transferred container. Also, in Staque, all items will be positioned so that even numbers will be located at the beginning and odd numbers at the bottom.

A screenshot of a computer

Description automatically generated with medium confidence

* isEmpty Constant function because it does not change the value of variables in the class. The non-parametric function returns only two values (bool type):
  + true, which means that Staque is empty.
  + false which means that there is at least one element in Staque.

Text

Description automatically generated

* show the Constant function because it does not change the value of the variables in the class. The function takes as a parameter returns nothing to the data source, only displays the values of the Staque elements or a message if it is empty:

Text

Description automatically generated

Note, the "<<" operator is overloaded.

* expandBy function that adds elements to Staque. When adding, the function adds even numbers to the top and odd numbers to the bottom. Also, when added, the last added Node memory address is stored in the last variable of Staque, which is will be store in the Node in the previouslyAdded variable. The function is overloaded:
  + If a function is passed two parameters, two x and n integers (of type int). The first number (x) is this element that is added to the Staque. The second number (n) by default equal to 1 and this number indicates how many times the first input parameter must be added to the Staque. In other words, the number x is added to the Staque n times. To avoid the case when the function is passed negative (n) n = abs (n). Text

    Description automatically generated
  + If a function is passed a single parameter, a container. All items from this container will be added to Staque. Text

    Description automatically generated
* removeFirst function removes elements from Staque by a LIFO (Last In First Out) method, ie the elements are removed from the end of the sequence in which it was added. For example, if numbers were added in the following order: 1 2 3 4 5, then according to this rule, it would be deleted in the following order: 5 4 3 2 1. Before deleting, it is checked whether there is any item in Staque. If Staque is empty, a message will appear on the screen. Count from the Node stored in the last variable of Staque elements. Also, the function is overloaded:
  + If a function is passed a single parameter, an integer of type n (int type), it deletes the last n added number.

Text

Description automatically generated

* + If a function is passed two parameters, an integer n (type int) and a letter ‘E’ - even or ‘O’ - odd (char type), it will delete the last n added even or odd numbers. The second parameter is checked first, if the function is given another letter (i.e. neither ‘E’ nor ‘O’), then a message appears on the screen. Next, start counting even numbers as they are placed in the back and it is easier to recalculate. The number of even numbers is subtracted from the Staque size, so the number of odd numbers is calculated. After that, it is checked whether there are n odd or even numbers in Staque. In other cases, a message appears on the screen.

Graphical user interface

Description automatically generated with medium confidence

* The "=" assignment operator is overloaded. When calling this operator, the assigning Staque, ie object to which the value is assigned, copies all the elements that are included in the assigned object. The addresses of the objects are checked first to avoid assigning themselves. Then all the elements in the assigning Staque are decomposed, after which the elements start copying from another object.

A screenshot of a computer screen

Description automatically generated with medium confidence

* At the end of the program, all elements in the Staque are destroyed and the memory is emptied by the Staque Destructor. It destroys all Nodes from the beginning to the end of Staque.