

# Queue

Software Documentation

Author: matiwa

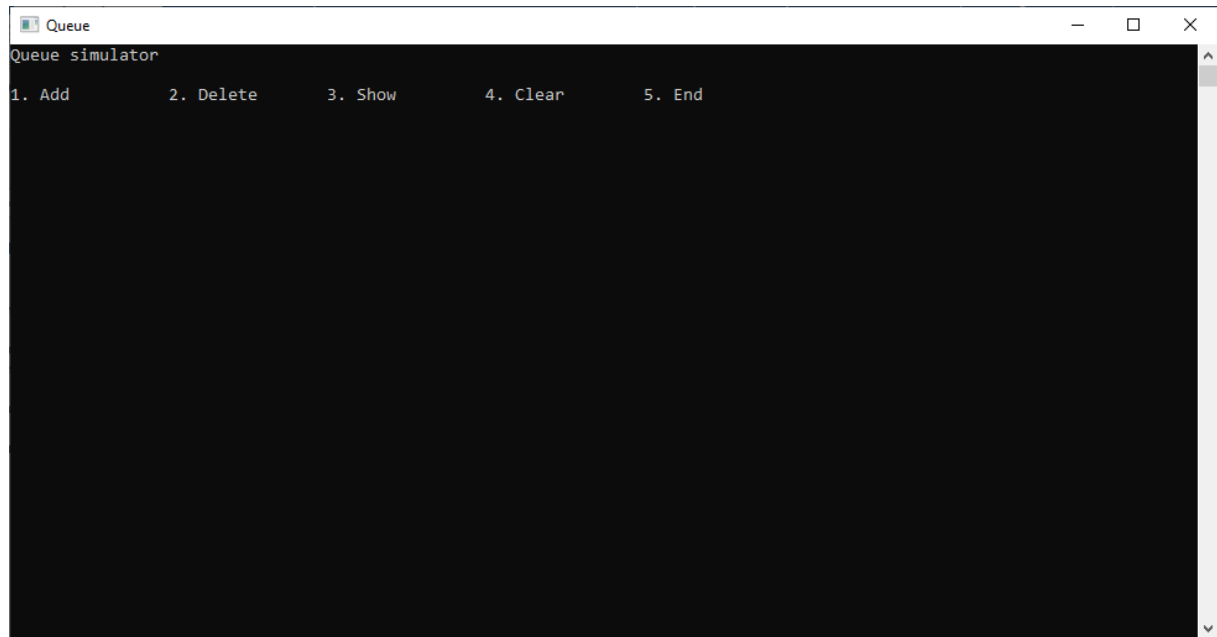
## Table of contents

Table of contents.....	2
Introduction.....	3
Describing of the application's operation.....	3
What is needed for use?.....	5
Some about queue.....	5
Interface description.....	5
Source code description.....	5
List of drawings.....	8
List of listings.....	8
Bibliography.....	8

## Introduction

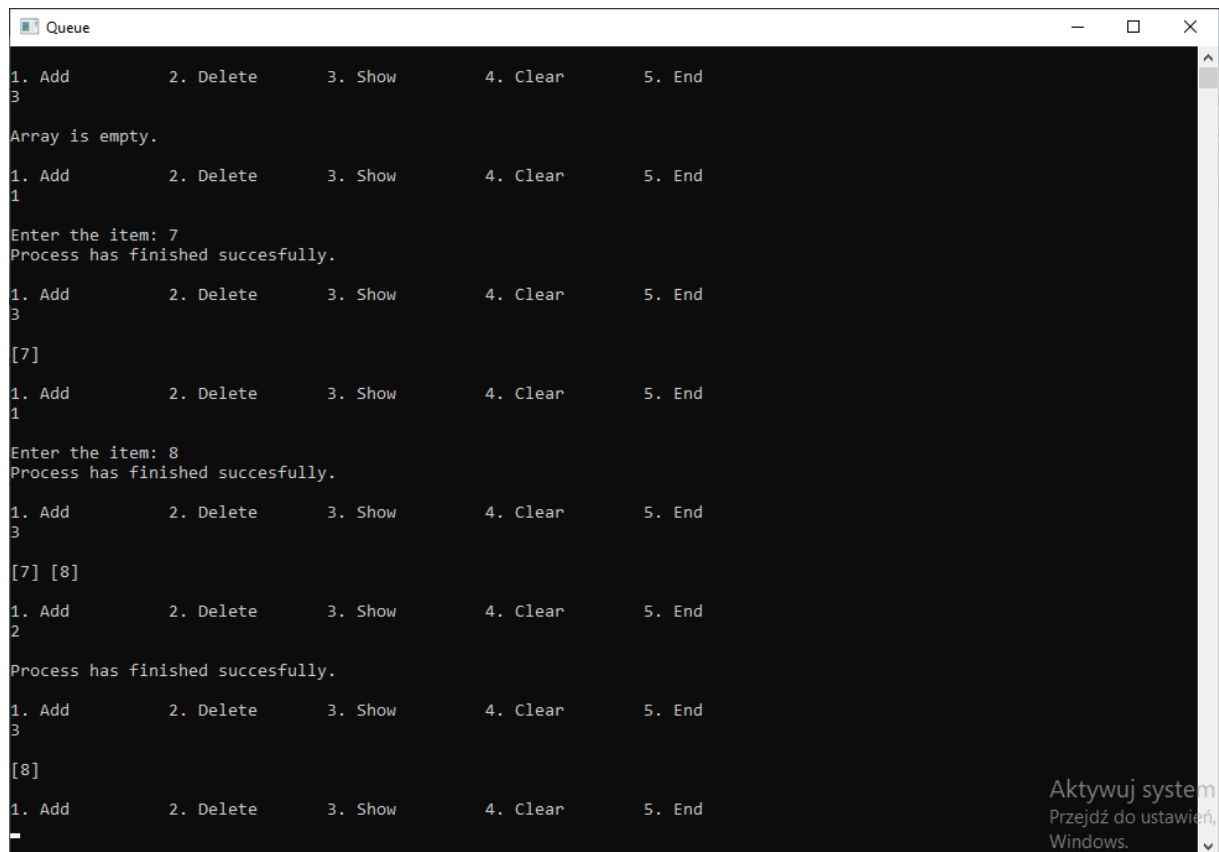
This software documentation includes: description of the application's operation, what is needed for use, algorithms used, interface description and source code description. This application is used to simulate the operation of the queue.

## Describing of the application's operation



Drawing 1: The beginning of the application's operation [own study]

The user has to choose whether to add, remove an item from the stack, view, clear the contents of the queue, or close the application. For this, the user must enter a number between 1 and 5. Initially, the contents of the queue are empty.




The screenshot shows a window titled "Queue" with a dark background and white text. The menu at the top lists: 1. Add, 2. Delete, 3. Show, 4. Clear, 5. End. The log shows the following sequence of events: 1. Initial state: "Array is empty." 2. Operation 1 (Add): Input "1", output "[7]". 3. Operation 2 (Add): Input "1", output "Enter the item: 7", "Process has finished succesfully." 4. Operation 3 (Add): Input "3", output "[7] [8]". 5. Operation 4 (Add): Input "2", output "Process has finished succesfully." 6. Operation 5 (Add): Input "3", output "[8]". The window also features a Windows watermark in the bottom right corner: "Aktywuj system. Przejdź do ustawień, Windows."

```
Queue
1. Add      2. Delete    3. Show     4. Clear    5. End
3
Array is empty.
1. Add      2. Delete    3. Show     4. Clear    5. End
1
Enter the item: 7
Process has finished succesfully.
1. Add      2. Delete    3. Show     4. Clear    5. End
3
[7]
1. Add      2. Delete    3. Show     4. Clear    5. End
1
Enter the item: 8
Process has finished succesfully.
1. Add      2. Delete    3. Show     4. Clear    5. End
3
[7] [8]
1. Add      2. Delete    3. Show     4. Clear    5. End
2
Process has finished succesfully.
1. Add      2. Delete    3. Show     4. Clear    5. End
3
[8]
1. Add      2. Delete    3. Show     4. Clear    5. End
```

Drawing 2: Operation on the queue [own study]

The application has no protection against an invalid option. This does not cause errors, the program does not exit, but does nothing.



The screenshot shows a window titled "Queue" with a dark background and white text. The menu at the top lists: 1. Add, 2. Delete, 3. Show, 4. Clear, 5. End. The log shows the following sequence of events: 1. Initial state: "Queue simulator". 2. Operation 1 (Add): Input "0", no output. 3. Operation 2 (Add): Input "-1", no output. 4. Operation 3 (Add): Input "50", no output. 5. Operation 4 (Add): Input "50", no output. The window also features a Windows watermark in the bottom right corner: "Aktywuj system. Przejdź do ustawień, Windows."

```
Queue
Queue simulator
1. Add      2. Delete    3. Show     4. Clear    5. End
0
1. Add      2. Delete    3. Show     4. Clear    5. End
-1
1. Add      2. Delete    3. Show     4. Clear    5. End
50
1. Add      2. Delete    3. Show     4. Clear    5. End
50
```

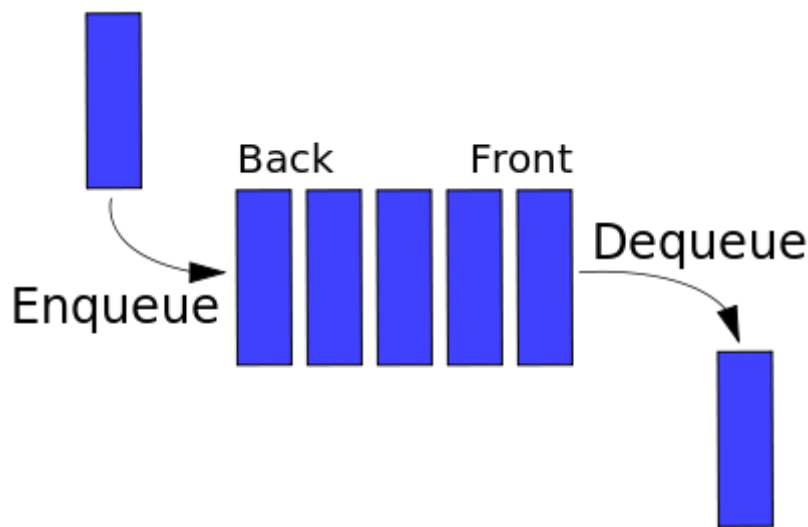
Drawing 3: Option selected incorrectly [own study]

What is needed for use?

The application does not require installation. It only needs the Windows operating system.

Some about queue

Queue - a linear data structure in which new data is added at the end of the queue, and data for further processing are retrieved from the beginning of the queue (buffer of FIFO type, First In, First Out; first in input, first in output).



Drawing 4: The idea of the queue

Queue operations are commonly called enqueue and dequeue. The queue works like a queue in a store and has a similar structure. It consists of a beginning (Front, Head) and an end (Back, Tail). A new element is always added at the end, similar to the client who always stands at the end of the queue. Removal from the queue corresponds to serving the customer in the store, i.e. the person who has waited the longest (i.e. is in front of the queue).

The priority queue is a special modification of the queue - each of the data contained in it is additionally assigned a priority (key), which is used to determine the order of individual elements in the set. This means that the first to be output will not necessarily be the data that is waiting the longest in the queue, but the data with the highest (or lowest) priority.[1]

Interface description

The interface is the console panel. The operation of the program is based on user communication. At the entrance, he chooses the desired option. If it adds an element, then it gives a value.

## Source code description

The project was made in the C++ programming language, in the Dev-C++ programming environment. All work was done on the Windows 10 operating system. The application's source code looks like this.

```
#include<iostream>
#include<windows.h>
#include<vector>
using namespace std;

int main(){
    SetConsoleTitleA("Queue");
    cout<<"Queue simulator"<<endl<<endl;
    vector<double> table;
    int option;
    while(true){
        cout<<"1. Add\t\t2. Delete\t3. Show\t\t4. Clear\t5.
End\r\n";
        cin>>option;
        cout<<endl;
        switch(option){
            case 1:{
                double item;
                cout<<"Enter the item: ";
                cin>>item;
                table.push_back(item);
                cout<<"Process has finished
succesfully."<<endl;
                break;
            }
            case 2:{
                vector<double> tmptable;
                for(int i=0;i<table.size();i++){
                    tmptable.push_back(table[i]);
                }
                table.clear();
                for(int i=0;i<tmptable.size()-1;i++)
                    table.push_back(tmptable[i+1]);
                tmptable.clear();
                cout<<"Process has finished
succesfully."<<endl;
                break;
            }
            case 3:{
                if(table.empty()) cout<<"Array is
empty."<<endl;
```

```

        else {
            for(int i=0;i<table.size();i++)
                cout<<"["<<table[i]<<"] ";
            cout<<endl;
        }
        break;
    }
    case 4:{
        table.clear();
        cout<<"Process has finished
successfully."<<endl;
        break;
    }
    case 5:{
        exit(0);
        break;
    }
}
cout<<endl;
}
return 0;
}

```

Listing 1: Source code [own study]

The option variable holds a value that specifies what action on the queue is to take. The user decides which option to choose. The vector table holds the values of the queue elements. Despite the lack of protection against the invalid option, the program does not exit, but performs no operation.

## List of drawings

Drawing 1: The beginning of the application's operation [own study].....	3
Drawing 2: Operation on the queue [own study].....	4
Drawing 3: Option selected incorrectly [own study].....	4
Drawing 4: The idea of the queue [2].....	5

## List of listings

Listing 1: Source code [own study].....	6
---	---

## Bibliography

- [1] [https://pl.wikipedia.org/wiki/Kolejka\\_\(informatyka\)](https://pl.wikipedia.org/wiki/Kolejka_(informatyka))
- [2] [https://pl.wikipedia.org/wiki/Plik:Data\\_Queue.svg](https://pl.wikipedia.org/wiki/Plik:Data_Queue.svg)