Stack

Software Documentation

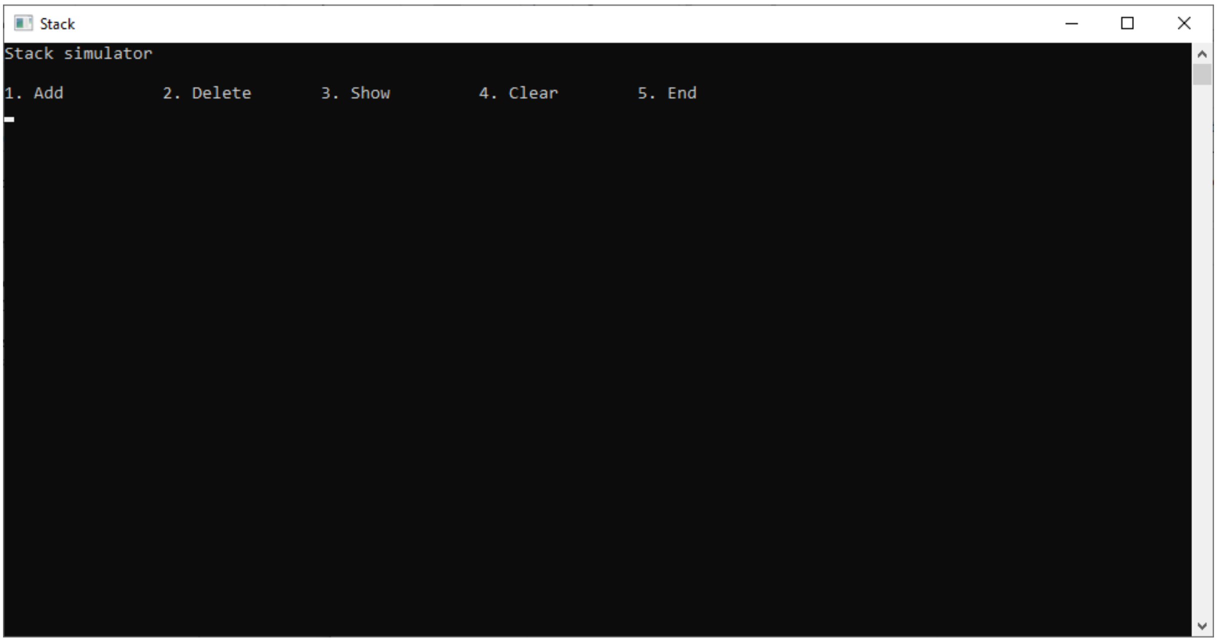
Author: matiwa

Table of contents

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 stack.

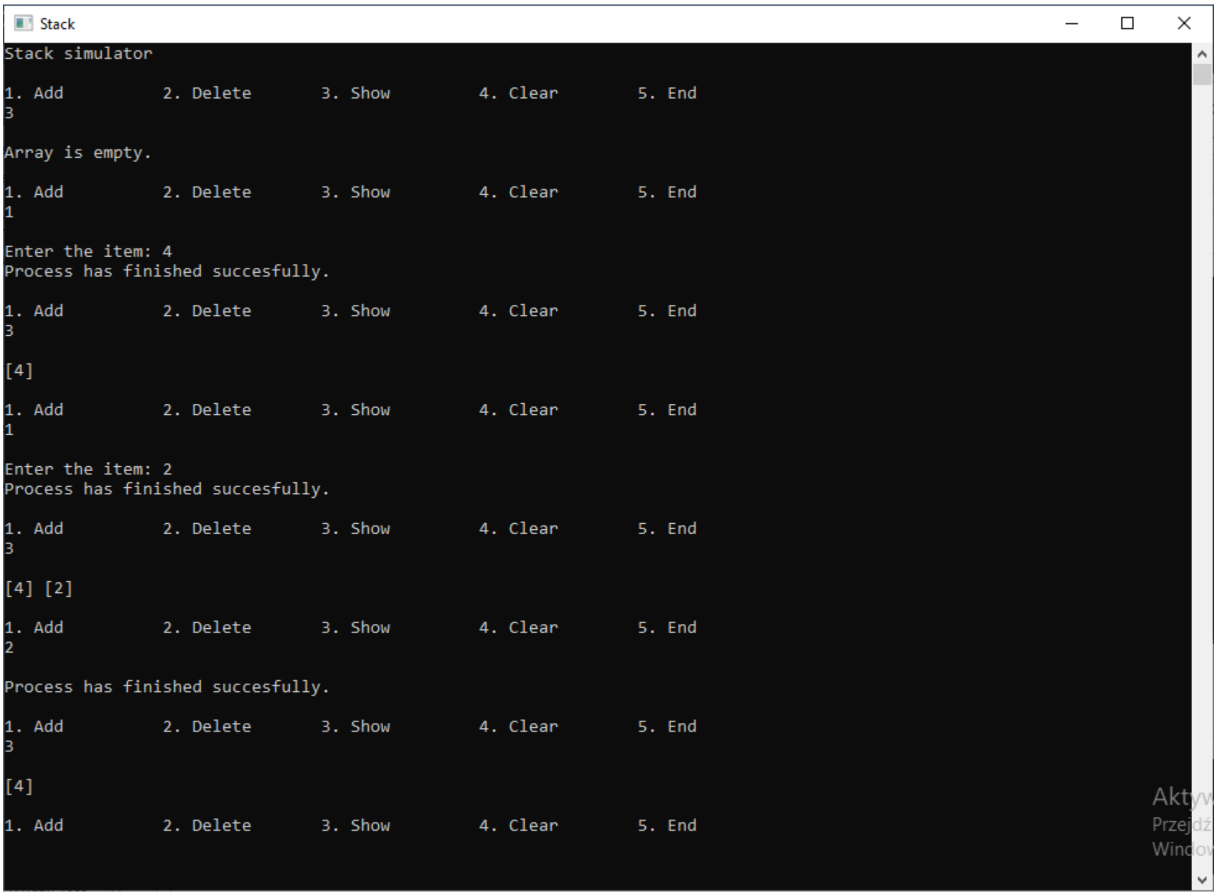
Describing of the application’s operation



```
Stack simulator

1. Add          2. Delete       3. Show         4. Clear        5. End
```

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

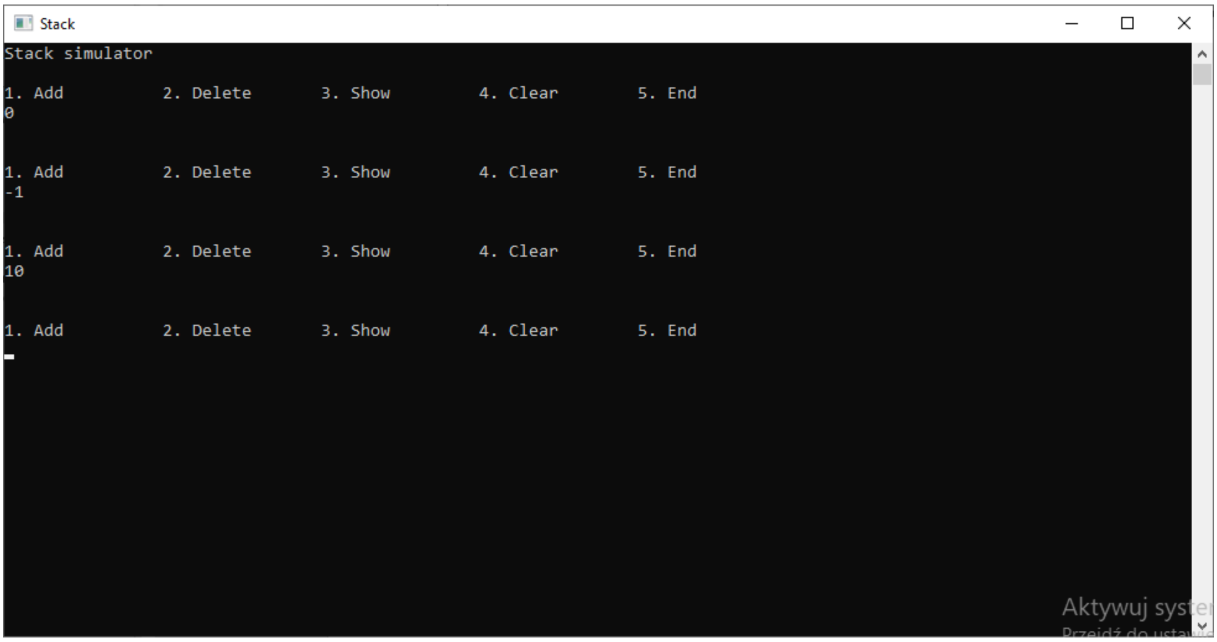
The user has to choose whether to add, remove an item from the stack, view, clear the contents of the stack, or close the application. For this, the user must enter a number between 1 and 5. Initially, the contents of the stack are empty.



```
Stack simulator

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: 4
Process has finished succesfully.

1. Add          2. Delete       3. Show         4. Clear        5. End
3

[4]

1. Add          2. Delete       3. Show         4. Clear        5. End
1

Enter the item: 2
Process has finished succesfully.

1. Add          2. Delete       3. Show         4. Clear        5. End
3

[4] [2]

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

[4]

1. Add          2. Delete       3. Show         4. Clear        5. End
```

Drawing 2: Operation on the stack [own study]

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



```
Stack 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
10


1. Add          2. Delete       3. Show         4. Clear        5. End
```

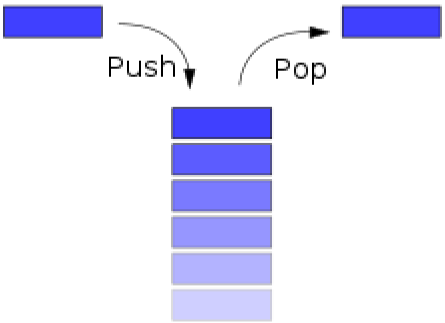
Drawing 3: Option selected incorrectly [own study]

What is needed for use?

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

Some about stack

Stack - a linear data structure in which data is added to the top of the stack and is retrieved from the top of the stack (LIFO buffer, Last In, First Out; last in, first out). The idea of a data stack can be illustrated as a stack of books stacked on top of each other - a new copy is placed on top of the stack and more copies are removed from the top of the stack. Items of the stack below the top can only be viewed, in order to download them, you must first download what is above them one by one.[1]



Drawing 4: The idea of the stack [2]

The basic operations that can be performed on the stack are formally written:

- push (object) - that is, pushing the object onto the stack;
- pop () - pull an object from the stack and return its value;
- isEmpty () - check if there are any objects on the stack.

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("Stack");
     cout<<"Stack 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:{
                    table.pop_back();
                    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 
succesfully."<<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 stack is to take. The user decides which option to choose. The vector table holds the values of the stack elements. Despite the lack of protection against the invalid option, the program does not exit, but performs no operation.

List of drawings

List of listings

Bibliography

[1] https://pl.wikipedia.org/wiki/Stos_(informatyka)
[2] https://upload.wikimedia.org/wikipedia/commons/thumb/2/29/Data_stack.svg/220px-Data_stack.svg.png