Об’єктно орієнтоване програмування

**Лабораторна робота №2**

# Стек

# Теоретичні відомості

Стек (англ. Stack – стопка) – структура даних, в якій доступ до елементів організований за принципом LIFO (англ. Last in – first out, «останнім прийшов – першим вийшов»). Найчастіше принцип роботи стека порівнюють зі стопкою тарілок: щоб взяти другу зверху, потрібно зняти верхню.

Додавання елемента, зване також проштовхуванням (push), можливо тільки в вершину стека (доданий елемент стає першим зверху). Видалення елемента, зване також виштовхуванням (pop), теж можливо тільки з вершини стека, при цьому другий зверху елемент стає верхнім (рис. 1).

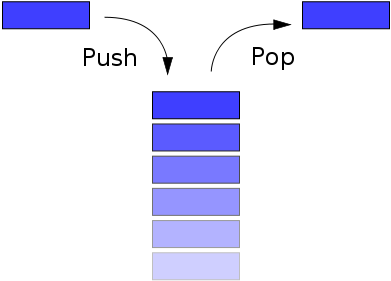


Рисунок 1 – Графічне представлення стека. (Зображення взято з <http://uk.wikipedia.org/wiki/Стек> )

Стеки широко застосовуються в обчислювальній техніці. Наприклад, для відстеження точок повернення з підпрограм використовується стек викликів, який є невід'ємною частиною архітектури більшості сучасних процесорів. Мови програмування високого рівня також використовують стек викликів для передачі параметрів при виклику процедур.

Для візуалізація ходу процесу виконання в Qt часто використовується компонент QProgressBar. Відображення ходу процесу можна здійснювати, задаючи значення позиції c допомогою функції SetValue, а діапазон можливий значень за допомогою функції SetRange. Наприклад, якщо повна тривалість процесу характеризується значенням цілої змінної Count (обсяг всіх копіюються файлів, число налаштувань, кількість циклів якогось процесу), а виконана частина – цілої змінної Current, то ставити позицію діаграми в разі, якщо використовуються значення мінімальної і максимальної позиції за замовчуванням (тобто 0 і 100), можна операторами:

ui->progressBar->SetValue(100 \* Current / Count);

Можна діяти інакше: задати спочатку значення максимальної величини рівним Count, а потім в ході процесу задавати позицію рівної Current. наприклад:

ui->progressBar->SetRange(0, Count);

ui->progressBar->SetValue(Current);

Таблиця 1 – Методи класу QProgressBar

|  |  |
| --- | --- |
| **Метод** | **Опис** |
| void setMaximum(int maximum) | Максимальне значення |
| void setMinimum(int minimum) | Мінімальне значення |
| void setOrientation(Qt::Orientation) | Орієнтація (вертикально/горизонтально) |
| void setRange(int minimum, int maximum) | Задати діапазон значень |
| void setValue(int value) | Задати поточне значення |

**Приклад.** Написати програму, яка реалізує стек для зберігання цілих чисел. Користувач може поміщати і отримувати дані з стека. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан.

Лістинг 1 – Файл stack.h

#ifndef STACK\_H

#define STACK\_H

#include <QTableWidget>

const int stacksize = 10;

const int EMPTY = -1;

class **Stack**

{

int arr[stacksize]; // Массив для хранения данных

int top; // Вершина стека

QTableWidget \*qtable; // Указатель на StringGrid

public:

**Stack**(); // Конструктор

void **push**(int c); // Добавление элемента

int **pop**(); // Выталкивание элемента

void **clear**(); // Очистка стека

bool **isEmpty**(); // Проверка на наличие элементов в стеке

bool **isFull**(); // Проверка на заполнение всего стека

int **getCount**(); // Количество элементов в стеке

void **setQTable**(QTableWidget \*qtable);

};

#endif // STACK\_H

Лістинг 2 – Файл stack.cpp

#include "stack.h"

#include <QString>

Stack::**Stack**()

{

top = EMPTY; // Изначально стек пуст

qtable = nullptr;

}

void Stack::**clear**()

{

top = EMPTY;

}

bool Stack::**isEmpty**()

{

return top == EMPTY;

}

bool Stack::**isFull**()

{

return top == stacksize - 1;

}

int Stack::**getCount**()

{

return top + 1; // Количество элементов в стеке

}

void Stack::**push**(int c)

{

if(!isFull())

{

top++;

arr[top] = c;

if(qtable)

{

qtable->insertRow(0);

qtable->setItem(0, 0, new QTableWidgetItem(QString::number(c)));

}

}

}

int Stack::**pop**()

{

if(!isEmpty())

{

top--;

if(qtable) qtable->removeRow(0);

return arr[top + 1];

}

else // Нечего извлекать

return 0;

}

void Stack::**setQTable**(QTableWidget \*qtable)

{

this->qtable = qtable;

}

Лістинг 3 – Файл mainwindow.cpp

#include <QMessageBox>

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "stack.h"

Stack stack;

MainWindow::**MainWindow**(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

// Наші налаштування

ui->tableWidget->setColumnCount(1);

ui->tableWidget->setHorizontalHeaderItem(0, new QTableWidgetItem("Stack data"));

stack.setQTable(ui->tableWidget);

}

MainWindow::~***MainWindow***()

{

delete ui;

}

void MainWindow::**on\_pushButton\_clicked**()

{ // Кнопка push

stack.push(ui->lineEdit->text().toInt());

}

void MainWindow::**on\_pushButton\_2\_clicked**()

{ // Кнопка pop

if(!stack.isEmpty())

{

int val = stack.pop();

ui->lineEdit->setText(QString::number(val));

}

else

{

QMessageBox::information(this, "stack ", "Stack is empty!!");

}

}

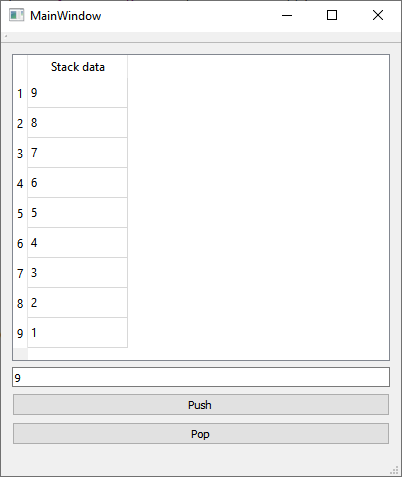


Рисунок 2 – Результат роботи програми

**Лабораторна робота №2 (3-й семестр)**

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п.3;
  3. У класі реалізований метод для обчислення мінімального значення елементів стека;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 3;
  3. У класі реалізований метод для обчислення максимального значення елементів стека;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 3;
  3. У класі реалізований метод для обчислення суми елементів стека;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 3;
  3. У класі реалізований метод для обчислення добутку елементів стека;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 3;
  3. У класі реалізований метод для обчислення мінімального позитивного значення елементів стека;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 3;
  3. У класі реалізований метод для обчислення максимального від'ємного значення елементів стека;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 3;
  3. У класі реалізований метод для обчислення мінімального за модулем значення елементів стека ,;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 3;
  3. У класі реалізований метод для обчислення максимального по модулю значення елементів стека;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

Написати програму, в якій

* 1. Реалізовано клас, що описує стек для зберігання дійсних чисел;
  2. Вміст стека зв'язується з деякими компонентом таким чином, щоб користувач міг бачити його актуальний стан. Користувачеві також виводиться результат виконання п. 3;
  3. У класі реалізований метод для обчислення суми позитивних елементів стека;
  4. Величина заповнення стека візуалізується за допомогою компонента QProgressBar.

**Варіант \* (Додаткове завдання)**

* 1. Реалізовано клас, що описує стек для зберігання символів (char);
  2. За допомогою стеку реалізувати можливість перевірки рядка на відповідність відкритих та закритих дужок. Перевіряються дужки (), {}, [].
  3. У випадку невідповідності, користувачу виводиться повідомлення, про відсутність відповідної закриваючої дужки.