МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**« МИРЭА Российский технологический университет»**

**РТУ МИРЭА**

Институт Информационных технологий

Кафедра Вычислительной техники

# УЧЕБНОЕ ЗАДАНИЕ

по дисциплине

« Объектно-ориентированное программирование»

Наименование задачи:

## « Задача 3_2_9_1 »

| | | |
|---|---|---|
| С тудент группы | ИНБО-15-20 | Ло В.Х. |
| Руководитель практики | Ассистент | Рогонова О.Н. |
| Работа представлена | «___»_____ 2021 г. | _____ |
| | | *(подпись студента)* |
| Оценка | | _____ |
| | | *(подпись руководителя)* |

Москва 2021

## Постановка задачи

Создать класс для объекта стек. Стек хранит целые числа. Имеет характеристики: наименование (строка, не более 10 символов) и размер (целое). Размер стека больше или равно 1. Функционал стека:
- добавить элемент и вернуть признак успеха (логическое);
- извлечь элемент (НЕ вывести!) и вернуть признак успеха (логическое);
- получить имя стека (строка);
- получить размер стека (целое);
- получить текущее количество элементов в стеке (целое).

В классе определить параметризированный конструктор, которому передается имя стека и размер. При переполнении стека очередной элемент не добавлять и определяется соответствующий признак успеха.
В основной программе реализовать алгоритм:
1. Ввести имя и размер для первого стека.
2. Создать объект первого стека.
3. Ввести имя и размер для второго стека.
4. Создать объект второго стека.
5. В цикле:
   5.1. Считывать очередное значение элемента.
   5.2. Добавлять элемент в первый стек, при переполнении завершить цикл.
   5.3. Добавлять элемент во второй стек, при переполнении завершить цикл.
6. Построчно вывести содержимое стеков.

## Описание входных данных

Первая строка: «имя стека 1»«размер стека»
Вторая строка: «имя стека 2»«размер стека»
Третья строка: Последовательность целых чисел, разделенных пробелами, в количестве не менее чем размер одного из стеков + 1.

## Описание выходных данных

Первая строка: «имя стека 1»«размер»
Вторая строка:

«имя стека 2»«размер»
Третья строка:
«имя стека 1»«имя стека 2»
Каждое имя стека в третьей строке занимает поле длины 15 позиции и прижата к левому краю.
Четвертая строка и далее построчно, вывести все элементы стеков:
«значение элемента стека 1»«значение элемента стека 2»
Вывод значений элементов стеков производиться последовательным извлечением. Каждое значение занимает поле из 15 позиции и прижата к правому краю.

## Метод решения

Потоки ввод/вывод cin/cout

Класса объекта: Stack

Описание класса: Stack

Свойства:

+ наименование (строка, не более 10 символов) и размер (целое)

+ Размер стека больше или равно 1.

Методы:

- Stack(string name,int size)- конструктор,примающий в аргументах значение имени стека и её максимальный размер
- ~Stack()- деструктор стека для очистки памяти
- bool input(int element)- добавляет элемент в стек, если это возможно
- bool output()- удаляет последний элемент стека,если это возможно
- int get(index)-возвращает элемента стека с данным индексом
- string get_name()-Возвращает имя стека
- int get_max_size()-Возвращает максимальный размер стека доступного для данного объекта
- int get_size()-возвращает размер стека

## Описание алгоритма

Класс объекта: Stack

Модификатор доступа: public

Метод: Stack()

Функционал: конструктор,примающий в аргументах значение имени стека и её максимальный размер

Параметры: name , size

Возвращаемое значение: нет

| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | конструктор,примающий в аргументах значение имени стека и её максимальный размер | ∅ | |

Класс объекта: Stack

Модификатор доступа: public

Метод: ~Stack()

Функционал: деструктор стека для очистки памяти

Параметры: нет

Возвращаемое значение: нет

| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | деструктор стека для очистки памяти | ∅ | |

Класс объекта: Stack

Модификатор доступа: public

Метод: input()

Функционал: добавляет элемент в стек, если это возможно

Параметры: element

Возвращаемое значение: int код,возврата

| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | добавляет элемент в стек, если это возможно | ∅ | |

Класс объекта: Stack

Модификатор доступа: public

Метод: output()

Функционал: удаляет последний элемент стека,если это возможно

Параметры: нет

Возвращаемое значение: нет

| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | удаляет последний элемент стека,если это возможно | ∅ | |

Класс объекта: Stack

Модификатор доступа: public

Метод: get(int index)

Функционал: Вернет значение элемента по данному индексу

Параметры: index

Возвращаемое значение: int код,возврата

| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | Вернет значение элемента по данному индексу | ∅ | |

Класс объекта: Stack

Модификатор доступа: public

Метод: get_name()

Функционал: Возвращает имя стека

Параметры: нет

Возвращаемое значение: string - имя стека

| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | Возвращает имя стека | ∅ | |

Класс объекта: Stack

Модификатор доступа: public

Метод: get_max_size()

Функционал: Возвращает значение максимального возможно размер стека

Параметры: нет

Возвращаемое значение: int код,возврата

| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | Возвращает значение максимального возможно размер стека | ∅ | |

Класс объекта: Stack

Модификатор доступа: public

Метод: get_size()

Функционал: Возвращает размер стека

Параметры: нет

Возвращаемое значение: int-код,возврата

| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | Возвращает размер стека | ∅ | |

Функция: main()

Функционал: Основнная программа

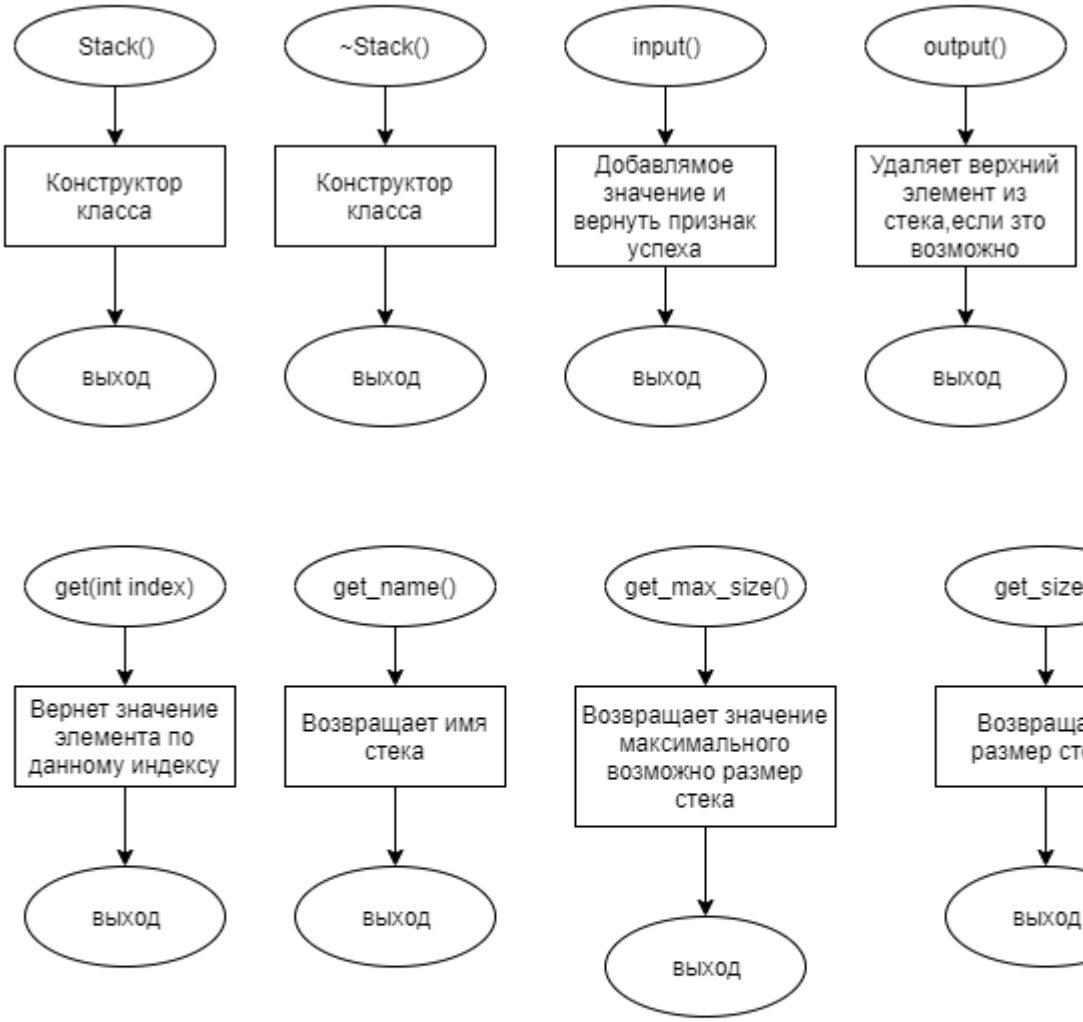
Параметры: нет

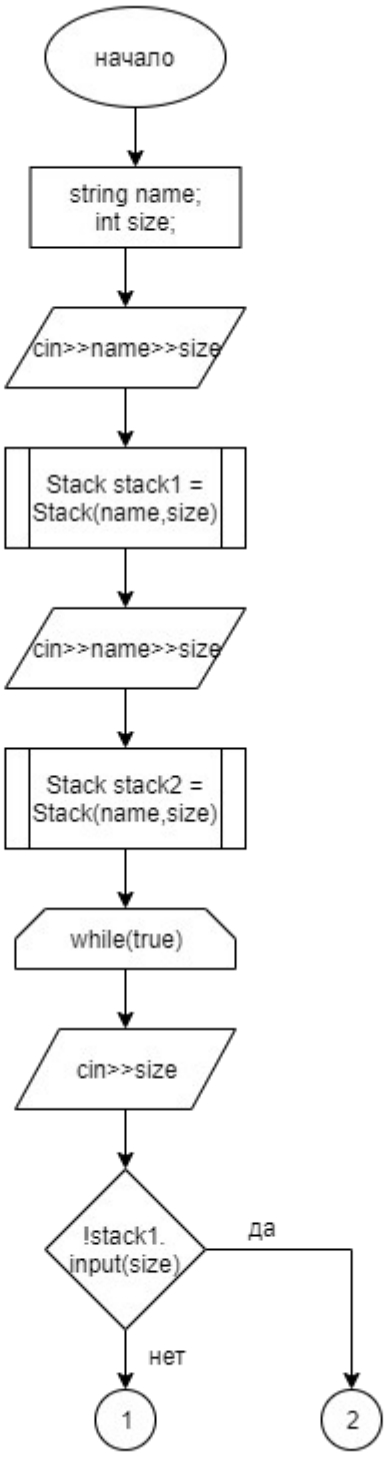
Возвращаемое значение: нет

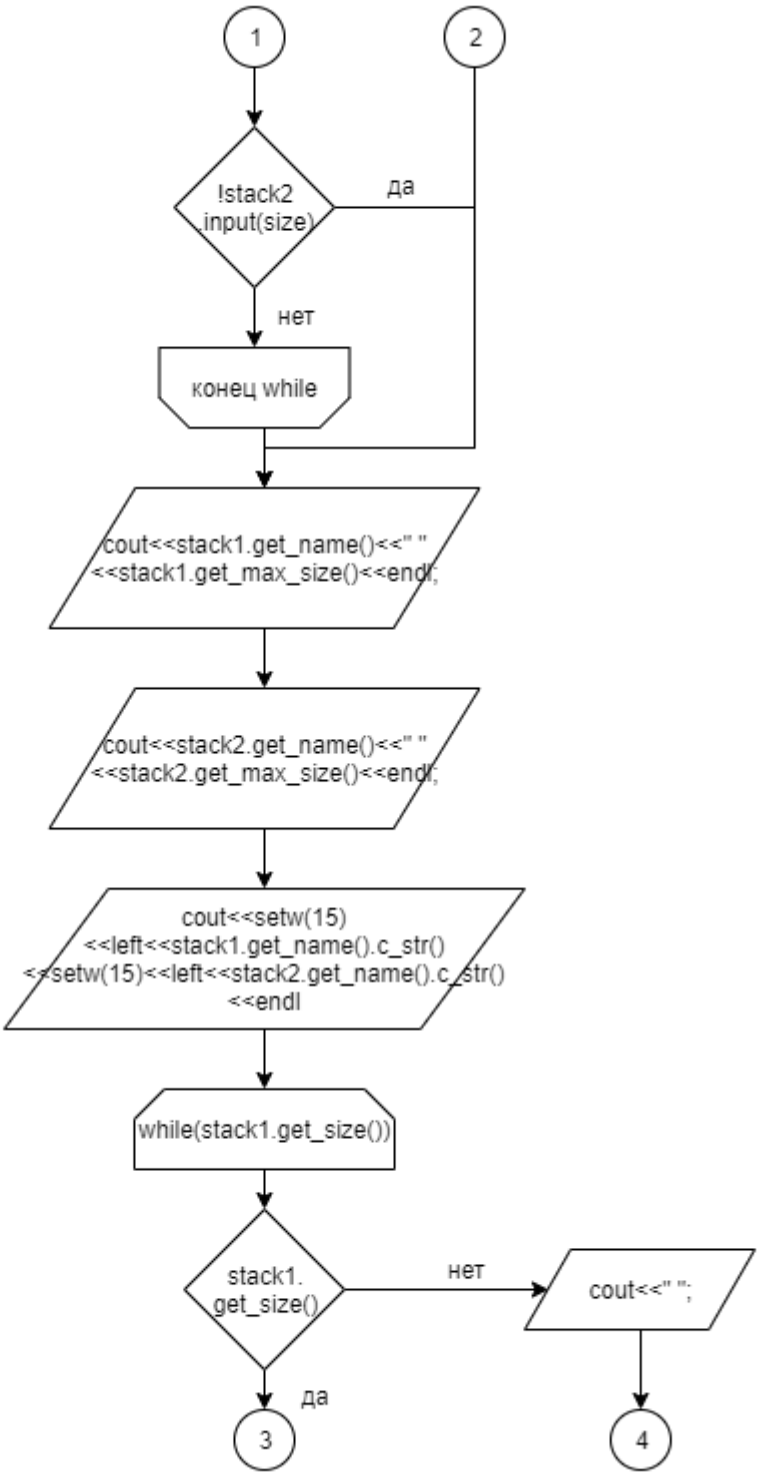
| № | Предикат | Действия | № перехода | Комментарий |
|---|---|---|---|---|
| 1 | | string name;<br>int size; | 2 | |
| 2 | | cin>>name>>size | 3 | |
| 3 | | Stack stack1 = Stack(name,size) | 4 | |

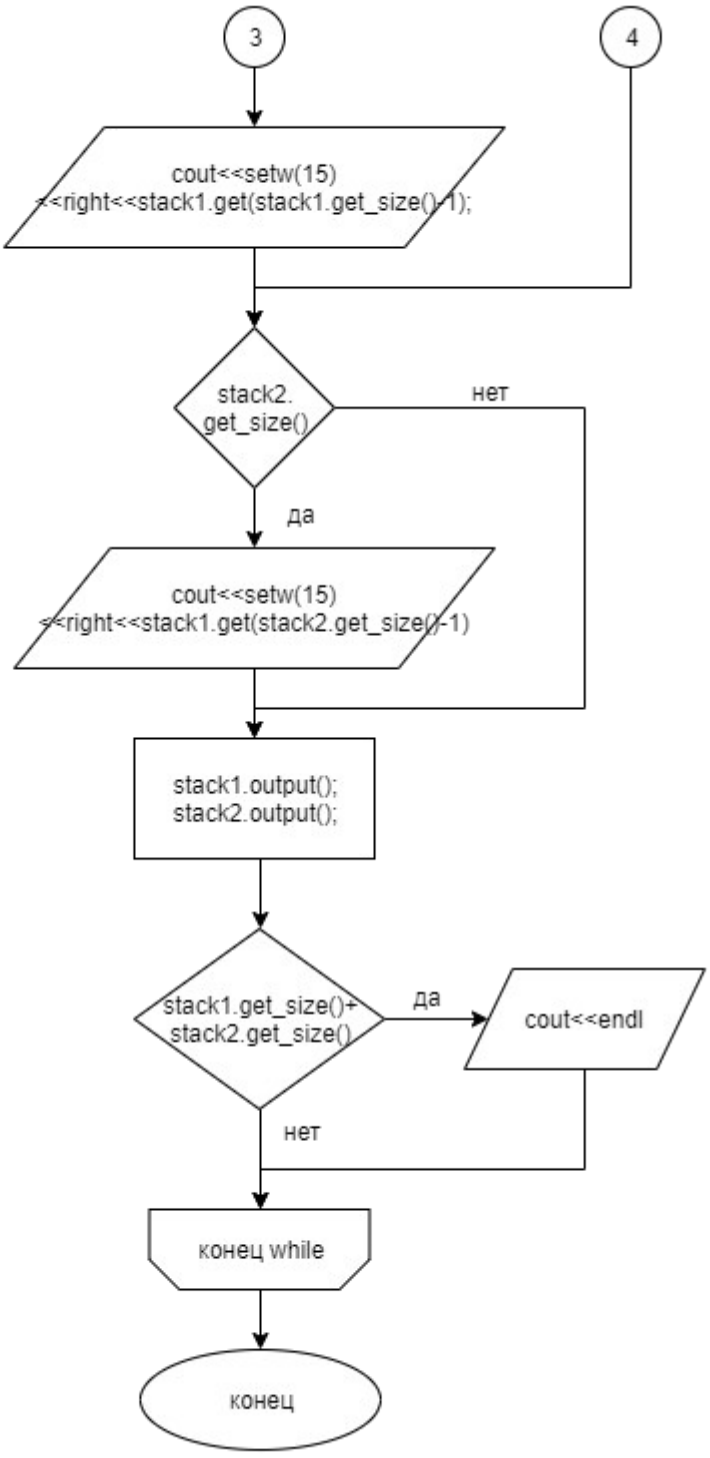
| | | | | |
|---|---|---|---|---|
| 4 | | cin>>name>>size | 5 | |
| 5 | | Stack stack2 = Stack(name,size) | 6 | |
| 6 | true | cin>>size | 7 | |
| | | | 17 | |
| 7 | !stack1.input(size) | | 9 | |
| | | | 8 | |
| 8 | !stack2.input(size) | | 9 | |
| | | | 6 | |
| 9 | | cout<<stack1.get_name()<<" "<<stack1.get_max_size()<<endl; | 10 | |
| 10 | | cout<<stack2.get_name()<<" "<<stack2.get_max_size()<<endl; | 11 | |
| 11 | | cout<<setw(15)<<left<<stack1.get_name().c_str()<<setw(15)<<left<<stack2.get_name().c_str()<<endl; | 12 | |
| 12 | stack1.get_size() | | 13 | |
| | | | 17 | |
| 13 | stack1.get_size() | cout<<setw(15)<<right<<stack1.get(stack1.get_size()-1) | 14 | |
| | | cout<<" "; | 14 | |
| 14 | stack2.get_size() | cout<<setw(15)<<right<<stack1.get(stack2.get_size()-1); | 15 | |
| | | | 15 | |
| 15 | | stack1.output();<br>stack2.output(); | 16 | |
| 16 | stack1.get_size()<br>+stack2.get_size() | cout<<endl | 12 | |
| | | | 12 | |
| 17 | | return 0 | ∅ | |

## Блок-схема алгоритма


Stack()
Конструктор класса
выход
~Stack()
Конструктор класса
выход
input()
Добавлямое значение и вернуть признак успеха
выход
output()
Удаляет верхний элемент из стека,если это возможно
выход
get(int index)
Вернет значение элемента по данному индексу
выход
get_name()
Возвращает имя стека
выход
get_max_size()
Возвращает значение максимального возможно размер стека
выход
get_size()
Возвращает размер стека
выход


начало
string name;
int size;
cin>>name>>size
Stack stack1 =
Stack(name,size)
cin>>name>>size
Stack stack2 =
Stack(name,size)
while(true)
cin>>size
!stack1.
input(size)
да
нет
1
2


1
2
!stack2
input(size)
да
нет
конец while
cout<<stack1.get_name()<<" "
<<stack1.get_max_size()<<endl;
cout<<stack2.get_name()<<" "
<<stack2.get_max_size()<<endl;
cout<<setw(15)
<<left<<stack1.get_name().c_str()
<<setw(15)<<left<<stack2.get_name().c_str()
<<endl
while(stack1.get_size())
stack1.
get_size()
нет
cout<<" ";
да
3
4




## Код программы

## Файл main.cpp

```
#include<iostream>
#include<string>
#include<iomanip>
#include "Stack.h"
int main()
{
	string name;
	int size;
	cin>>name>>size;
	Stack stack1 = Stack(name,size);
	cin>>name>>size;
	Stack stack2 = Stack(name,size);
	while(true)
	{
		cin>>size;
		if(!stack1.input(size))
		{
			break;
		}
		if(!stack2.input(size))
		{
			break;
		}
	}
	cout<<stack1.get_name()<<" "<<stack1.get_max_size()<<endl;
	cout<<stack2.get_name()<<" "<<stack2.get_max_size()<<endl;

cout<<setw(15)<<left<<stack1.get_name().c_str()<<setw(15)<<left<<stack2.get_na
me().c_str()<<endl;
	while(stack1.get_size())
	{
		if(stack1.get_size())
		{
			cout<<setw(15)<<right<<stack1.get(stack1.get_size()-
1);
		}

		else
			cout<<"               ";
		if(stack2.get_size())
		cout<<setw(15)<<right<<stack1.get(stack2.get_size()-1);
		stack1.output();
		stack2.output();
		if(stack1.get_size()+stack2.get_size())
		cout<<endl;
	}
return 0;
}
```

## Файл Stack.cpp

```
#include "Stack.h"
Stack::Stack(string name,int size)
{
        this->name=name;
        this->max_size=size;
        this->array = new int[this->max_size];
        this->size =0;
}
Stack::~Stack()
{
        delete this->array;
}
bool Stack::input(int element)
{
        if(this->size == this->max_size)
        {
                return false;
        }
        this->array[this->size]=element;
        this->size++;
        return true;
}
bool Stack::output()
{
        if(this->size>0)
        {
                this->size--;
                return true;
        }
        return false;
}
int Stack::get(int index)
{
        return this->array[index];
}
string Stack::get_name()
{
        return this->name;
}
int Stack::get_max_size()
{
        return this->max_size;
}
int Stack::get_size()
{
        return this->size;
}
```

## Файл Stack.h

```
#ifndef STACK_H
#define STACK_H
#define MAX 100
#include<iostream>
#include<string>
#include<iomanip>
using namespace std;
class Stack
{
                private:
                                int max_size;
                                int *array;
                                string name;
                                int size;
                public:
                                Stack(string name,int size);
                                ~Stack();
                                bool input(int element);
                                bool output();
                                int get(int index);
                                string get_name();
                                int get_max_size();
                                int get_size();
};
#endif
```

## Тестирование

| Входные данные | Ожидаемые выходные данные | Фактические выходные данные |
|---|---|---|
| stack1 3 stack2 6 1 2 3 4 5 6 | stack1 3 stack2 6 stack1 stack2 3 3 2 2 1 1 | stack1 3 stack2 6 stack1 stack2 3 3 2 2 1 1 |