

**Министерство образования и науки Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4**

### Студент Завойских Евгения Васильевна

*фамилия, имя, отчество*

### Группа ИУ7-33Б

Студент Завойских Е.В.

*подпись, дата фамилия и.о.*

### Руководитель практики Барышникова М.Ю.

*подпись, дата фамилия, и.о.*

### Оценка

*2021 г.*

**Условие задачи**

Разработать программу работы со стеком, реализующую операции добавления и удаления элементов из стека и отображения текущего состояния стека. Реализовать стек: а) массивом; б) списком. Все стандартные операции со стеком должны быть оформлены подпрограммами. При реализации стека списком в вывод текущего состояния стека добавить просмотр адресов элементов стека и создать свой список или массив свободных областей (адресов освобождаемых элементов) с выводом его на экран.

Распечатать убывающие серии последовательности целых чисел в обратном порядке.

**Внешняя спецификация программы**

1. Входные данные

При вызове программы выводится меню, каждый раз пользователь выбирает один из пунктов.

Стек заполняется целыми числами, числа вводятся с клавиатуры по одному.

1. Выходные данные

На экран выводятся текущие состояния стеков (текущее количество элементов, максимально возможное количество элементов в стеке, адрес и значение каждого элемента + для реализации списком адреса уже освобожденной памяти и их количество), замеры времени выполнения (push, pop) и выделяемой памяти для двух реализаций стека в виде таблицы, убывающие серии последовательности целых чисел в обратном порядке. При удалении элемента с вершины стека выводится значение этого элемента.

В случае некорректного ввода команды меню выводится сообщение об ошибке и предоставляется возможность выбрать команду еще раз. В случае некорректного ввода числа для добавления в стек также выводится соответствующее сообщение об ошибке. При ошибке выделения памяти, попытке удалить элемент из пустого стека или добавить элемент в уже заполненный стек выводится соответствующее сообщение об ошибке.

1. Задача, реализуемая программой

Данная программа предназначена для работы со стеком, реализованным массивом, и стеком, реализованным списком, и представляет собой приложение со следующими возможными операциями:

1. Добавить элемент в стек (массив).
2. Добавить элемент в стек (список).
3. Удалить элемент из стека (массив).
4. Удалить элемент из стека (список).
5. Посмотреть текущее состояние стека (массив).
6. Посмотреть текущее состояние стека (список).
7. Вывести замеры времени и памяти для двух реализаций стека.
8. Вывести убывающие серии чисел из стека в обратном порядке (массив).
9. Вывести убывающие серии чисел из стека в обратном порядке (список).

1. Способ обращения к программе

Программа может быть вызвана из консоли или запущена в любой среде разработки, поддерживающей язык С.

1. Возможные аварийные ситуации и ошибки пользователя

В случае аварийной ситуации программа выводит сообщение о произошедшей ошибке. Ошибка может возникнуть в результате некорректного ввода (не соответствующего формату), при попытке удалить элемент из пустого стека или добавить элемент в уже заполненный стек, выделении динамической памяти.

**Внутренние структуры данных**

В программе используются структуры для хранения стека в виде матрицы и списка. Для реализации списком используется структура для хранения элемента стека и структура для хранения элемента списка адресов освобожденной памяти.

// структура для определения стека в виде массива

struct stack\_array

{

    int arr[MAX\_ARRAY\_SIZE]; // массив

    int \*ps; // указатель стека

};

typedef struct stack\_array stack\_array\_s;

// структура для определения элемента стека

struct node

{

    int val; // значение элемента

    struct node \*prev; // указатель на предыдущий элемент

};

typedef struct node node\_s;

// структура для определения элемента в списке адресов освобожденной памяти

struct free\_node

{

    node\_s \*adr; // адрес освобожденной памяти

    struct free\_node \*prev; // указатель на предыдущий элемент

};

typedef struct free\_node free\_node\_s;

// структура для определения стека в виде списка

struct stack\_list

{

    node\_s \*ps; // указатель стека

    int cnt; // количество элементов в стеке

    free\_node\_s \*free\_nodes; // список адресов освобождаемой памяти

    int free\_cnt; // количество адресов освобожденной памяти

};

typedef struct stack\_list stack\_list\_s;

**Алгоритм**

Пока пользователь не решит завершить программу, выполняется действие, выбранное из меню:

1. Добавить элемент в стек (массив): с клавиатуры вводится целое число, если массив не заполнен, число добавляется на вершину стека (в конец массива, туда же смещается указатель стека)
2. Добавить элемент в стек (список): с клавиатуры вводится целое число, если не достигнут максимум количества элементов, число добавляется на вершину стека (выделяется память, адрес которой записывается в указатель стека)
3. Удалить элемент из стека (массив): если стек не пустой, возвращается значение элемента в вершины стека, указатель стека смещается к предыдущему элементу
4. Удалить элемент из стека (список): если стек не пустой, возвращается значение элемента в вершины стека, адрес вершины добавляется в список адресов освобожденной памяти и освобождается, указатель стека смещается к предыдущему элементу
5. Посмотреть текущее состояние стека (массив): выводится количество элементов в стеке, максимально возможное количество элементов в стеке, адрес и значение каждого элемента стека, начиная с вершины
6. Посмотреть текущее состояние стека (список): выводится количество элементов в стеке, максимально возможное количество элементов в стеке, адрес и значение каждого элемента стека, начиная с вершины, количество адресов освобожденной памяти и сами адреса
7. Вывести замеры времени и памяти для двух реализаций стека: выводится таблица: первый столбец – реализация (массив или список), второй – количество элементов в стеке, третий – время выполнения push, четвертый – время выполнения pop, пятый – количество памяти, выделенной под стек
8. Вывести убывающие серии чисел из стека в обратном порядке (массив): из стека по одному извлекаются числа, каждое число добавляется в стек-копию, если новое число больше предыдущего, оно добавляется в стек серии, иначе выводится стек серии с вершины (и освобождается), и новое число добавляется в пустой стек серии, и так до конца исходного стека; в конце стек восстанавливается из стека-копии
9. Вывести убывающие серии чисел из стека в обратном порядке (список): из стека по одному извлекаются числа, каждое число добавляется в стек-копию, если новое число больше предыдущего, оно добавляется в стек серии, иначе выводится стек серии с вершины (и освобождается), и новое число добавляется в пустой стек серии, и так до конца исходного стека; в конце стек восстанавливается из стека-копии

**Тесты**

Негативы:

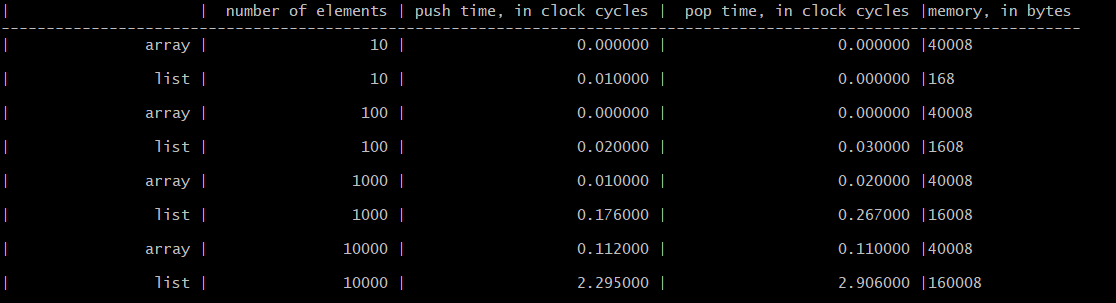
|  |  |  |
| --- | --- | --- |
| Входные данные | Вывод | Что проверяется |
| Возможные действия (введите номер нужной команды):  0. Завершить работу  1. Добавить элемент в стек (массив)  2. Добавить элемент в стек (список)  3. Удалить элемент из стека (массив)  4. Удалить элемент из стека (список)  5. Посмотреть текущее состояние стека (массив)  6. Посмотреть текущее состояние стека (список)  7. Вывести замеры времени и памяти для двух реализаций стека  8. Вывести убывающие серии чисел из стека в обратном порядке (массив)  9. Вывести убывающие серии чисел из стека в обратном порядке (список)  -10 | Нет команды с введенным номером | Неверно введена команда меню |
| 5  В стеке 0 элементов, максимально возможно 10000  3 | Стек пустой | Попытка удалить элемент из пустого стека (массив) |
| 6  В стеке 0 элементов, максимально возможно 10000  Освобождено 0 областей памяти  4 | Стек пустой | Попытка удалить элемент из пустого стека (список) |
| 5  В стеке 0 элементов, максимально возможно 10000  8 | Стек пустой | Вывод убывающих серий чисел из пустого стека в обратном порядке (массив) |
| 6  В стеке 0 элементов, максимально возможно 10000  Освобождено 0 областей памяти  9 | Стек пустой | Вывод убывающих серий чисел из пустого стека в обратном порядке (список) |
| 1  Введите целое число: q | Ошибка ввода | Некорректный ввод числа для добавления в стек (массив) |
| 2  Введите целое число: у | Ошибка ввода | Некорректный ввод числа для добавления в стек (список) |
| 5  В стеке 3 элементов, максимально возможно 3  Начиная с вершины:  Адрес элемента: 0000003acbdff928, значение: 0  Адрес элемента: 0000003acbdff924, значение: 22  Адрес элемента: 0000003acbdff920, значение: 1  1  Введите целое число: 3 | Стек уже заполнен | Попытка добавить число в уже заполненный стек (массив) |
| 6  В стеке 3 элементов, максимально возможно 3  Начиная с вершины:  Адрес элемента: 000001f8b4b22470, значение: -99  Адрес элемента: 000001f8b4b22450, значение: 100  Адрес элемента: 000001f8b4b22430, значение: 43  Освобождено 0 областей памяти  2  Введите целое число: 4 | Стек уже заполнен | Попытка добавить число в уже заполненный стек (список) |

Позитивы:

|  |  |  |
| --- | --- | --- |
| Входные данные | Вывод | Что проверяется |
| 1  Введите целое число: 3 | Добавлен  5  В стеке 1 элементов, максимально возможно 10000  Начиная с вершины:  Адрес элемента: 000000929cff6070, значение: 3 | Добавление числа в стек (массив) |
| 2  Введите целое число: 3 | Добавлен  6  В стеке 1 элементов, максимально возможно 10000  Начиная с вершины:  Адрес элемента: 00000244e7532430, значение: 3  Освобождено 0 областей памяти | Добавление числа в стек (список) |
| 3 | 3  5  В стеке 0 элементов, максимально возможно 10000 | Удаление числа с вершины стека (массив) |
| 4 | 3  6  В стеке 0 элементов, максимально возможно 10000  Освобождено 1 областей памяти  Адрес освобожденной области памяти: 00000244e7532430 | Удаление числа с вершины стека (список) |
| 5 | В стеке 11 элементов, максимально возможно 10000  Начиная с вершины:  Адрес элемента: 000000929cff6098, значение: 10  Адрес элемента: 000000929cff6094, значение: 2  Адрес элемента: 000000929cff6090, значение: 6  Адрес элемента: 000000929cff608c, значение: 6  Адрес элемента: 000000929cff6088, значение: 7  Адрес элемента: 000000929cff6084, значение: 14  Адрес элемента: 000000929cff6080, значение: 15  Адрес элемента: 000000929cff607c, значение: 5  Адрес элемента: 000000929cff6078, значение: 3  Адрес элемента: 000000929cff6074, значение: 24  Адрес элемента: 000000929cff6070, значение: 100 | Вывод состояния стека (массив) |
| 6 | В стеке 11 элементов, максимально возможно 10000  Начиная с вершины:  Адрес элемента: 00000244e7532590, значение: 10  Адрес элемента: 00000244e7532570, значение: 2  Адрес элемента: 00000244e7532550, значение: 6  Адрес элемента: 00000244e7532530, значение: 6  Адрес элемента: 00000244e7532510, значение: 7  Адрес элемента: 00000244e75324f0, значение: 14  Адрес элемента: 00000244e75324d0, значение: 15  Адрес элемента: 00000244e75324b0, значение: 5  Адрес элемента: 00000244e7532490, значение: 3  Адрес элемента: 00000244e7532470, значение: 24  Адрес элемента: 00000244e7532430, значение: 100  Освобождено 1 областей памяти  Адрес освобожденной области памяти: 00000244e7532430 | Вывод состояния стека (список) |
| 5  В стеке 11 элементов, максимально возможно 10000  Начиная с вершины:  Адрес элемента: 000000929cff6098, значение: 10  Адрес элемента: 000000929cff6094, значение: 2  Адрес элемента: 000000929cff6090, значение: 6  Адрес элемента: 000000929cff608c, значение: 6  Адрес элемента: 000000929cff6088, значение: 7  Адрес элемента: 000000929cff6084, значение: 14  Адрес элемента: 000000929cff6080, значение: 15  Адрес элемента: 000000929cff607c, значение: 5  Адрес элемента: 000000929cff6078, значение: 3  Адрес элемента: 000000929cff6074, значение: 24  Адрес элемента: 000000929cff6070, значение: 100  8 | 6 2  15 14 7 6  100 24 3 | Вывод убывающих серий чисел из стека в обратном порядке (массив) |
| 6  В стеке 11 элементов, максимально возможно 10000  Начиная с вершины:  Адрес элемента: 00000244e7532590, значение: 10  Адрес элемента: 00000244e7532570, значение: 2  Адрес элемента: 00000244e7532550, значение: 6  Адрес элемента: 00000244e7532530, значение: 6  Адрес элемента: 00000244e7532510, значение: 7  Адрес элемента: 00000244e75324f0, значение: 14  Адрес элемента: 00000244e75324d0, значение: 15  Адрес элемента: 00000244e75324b0, значение: 5  Адрес элемента: 00000244e7532490, значение: 3  Адрес элемента: 00000244e7532470, значение: 24  Адрес элемента: 00000244e7532430, значение: 100  Освобождено 1 областей памяти  Адрес освобожденной области памяти: 00000244e7532430  9 | 6 2  15 14 7 6  100 24 3 | Вывод убывающих серий чисел из стека в обратном порядке (список) |
| 7 | 1.PNG | Вывод таблицы по замерам времени и памяти для двух реализаций стека |

**Вывод**

Замеры времени (в тактах) и замеры памяти (в байтах) для двух реализаций стека:



При одинаковом количестве добавляемых и удаляемых из стека элементов видно, что для стека в виде массива операции push, pop работают в 10-20 раз быстрее, чем для стека в виде списка. Так как я использовала статический массив, память, выделяемая под стек в виде массива, всегда одинакова; она становится сравнима с памятью, выделяемой под стек в виде списка, при количестве элементов равном 1000, далее память, выделяемая под стек в виде списка, становится все больше памяти, выделяемой под стек в виде массива.

По итогу, когда количество элементов стека нам заранее неизвестно, есть смысл использовать реализацию стека списком, так как в этом случае объем стека ограничивается только объемом доступной оперативной памяти + при небольшом (менее 1000) количестве элементов список занимает меньше памяти, чем массив. Если же количество элементов в стеке известно заранее и оно превышает 1000 и по времени, и по памяти лучше использовать реализацию стека массивом.

**Ответы на вопросы**

1. **Что такое стек?**

Стек – это последовательный список с переменной длиной, в котором включение и исключение элементов происходит только с одной стороны – с его вершины.

1. **Каким образом и сколько памяти выделяется под хранение стека при различной его реализации?**

При хранении стека в виде массива обычно отводится непрерывная область памяти ограниченного размера, имеющая нижнюю и верхнюю границу. При хранении стека в виде списка объем памяти ограничен объемом доступной оперативной памяти, но может быть ограничен искусственно. Память под массив равна MAX\_SIZE \* sizeof(элемент). Память для списка выделяется как под сами элементы списка, так и под указатели на предыдущий элемент списка.

1. **Каким образом освобождается память при удалении элемента стека при различной реализации стека?**

При реализации стека массивом память освобождается в конце области видимости массива при уничтожении соответствующей переменной. При реализации стека списком память, выделенная под элемент, освобождается при каждом удалении элемента.

1. **Что происходит с элементами стека при его просмотре?**

При просмотре элемента, он удаляется с вершины стека, т.е. после просмотра всех элементов стек станет пустым.

1. **Каким образом эффективнее реализовывать стек? От чего это зависит?**

Эффективнее реализовывать стек массивом, когда нам заранее известно, сколько элементов будет в стеке и их количество велико, так как добавление и удаление элемента осуществляются быстрее, чем при реализации стека списком, а память выделяется один раз. Реализация списком может быть выгодна по памяти только при небольшом количестве элементов в стеке, и когда количество элементов в стеке заранее неизвестно.