|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования 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 К ЛАБОРАТОРНОЙ РАБОТЕ №4 ПО «ТИПАМ И СТРУКТУРАМ ДАННЫХ»**

**Тема: «Работа со стеком»**

Группа: ИУ7-32Б

|  |  |  |
| --- | --- | --- |
| Студент: |  | Сидоров Максим Михайлович |

Преподаватель: Никульшина Татьяна Александровна

**Цель работы:** научиться работать со стеком в различных его представлениях.

**Условие задачи:** создать программу работы со стеком, выполняющую операции добавление, удаления элементов и вывод 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.

При реализации стека массивом располагать два стека в одном массиве. Один стек располагается в начале массива и растет к концу, а другой располагается в конце массива и растет к началу. Заполнять и освобождать стеки произвольным образом с экрана. Элементами стека являются вещественные числа. Списком реализовать один стек.

**Входные данные:** номер команды из меню, терминальный ввод.

Команды:

1. Добавить элемент в стек-список
2. Добавить элемент в первый стек стека-массива
3. Добавить элемент во второй стек стека-массива
4. Удалить элемент из стека-списка
5. Удалить элемент из первого стека стека-массива
6. Удалить элемент из второго стека стека-массива
7. Вывести стек-список
8. Вывести стек-массив
9. Вывести освобожденные адреса списка
10. Вывести текущие адреса списка
11. Очистить стек-список
12. Очистить первый стек стек-массива
13. Очистить второй стек стек-массива
14. Выйти

**Выходные данные:**

- текущее состояние стека

- сравнение массива и списка

- освобожденные адреса списка

- текущие адреса списка

**Аварийные ситуации:**

- ввод неправильной команды

- некорректный ввод элементов стека

- удаление элементов из пустого стека

**Описание СД:**

1. Для хранения ячейки стека-списка было выбрано следующее представление:

struct stack\_cell

{

double num;

int pos;

stack\_cell \*prev;

};

*Листинг 1. Представление стека-списка.*

1. Для хранения стека-массива было выбрано следующее представление:

#define STACK\_SIZE 20

typedef struct

{

double stack[STACK\_SIZE \* 2];

int first\_len;

int second\_len;

} stack\_arr;

*Листинг 2. Представление стека-массива.*

**Функции:**

stack\_arr init\_stack() – инициализация стека-массива

bool first\_is\_empty(stack\_arr \*a) – проверка на то, пустой ли первый стек стека-массива

bool second\_is\_empty(stack\_arr \*a) - проверка на то, пустой ли второй стек стека-массива

bool is\_full\_arr(stack\_arr \*a) – проверка на то, заполнен ли стек-массив

void push\_first(stack\_arr \*a, double num) – добавление значения в первый стек

void push\_second(stack\_arr \*a, double num) – добавление значения во второй стек

void pop\_first(stack\_arr \*a) – удаление элемента из первого стека

void pop\_second(stack\_arr \*a) – удаление элемента из второго стека

void print\_stack\_arr(stack\_arr \*a) – вывести стек-массив

void clear\_first(stack\_arr \*a) – очистка первый стек стека-массива

void clear\_second(stack\_arr \*a) – очистка второй стек стека-массива

bool is\_empty(stack\_cell \*a) – проверка на то, пустой ли стек-список

bool is\_full\_list(stack\_cell \*a) – проверка на то, заполнен ли стек-список

stack\_cell \*push(stack\_cell \*a, double num) – добавление элемента в стек-список

stack\_cell \*pop(stack\_cell \*a) – удаление элемента из стека-списка

void clear(stack\_cell \*\*a) – очистка стека-списка

void print\_stack\_list(stack\_cell \*a) – вывод стека-списка

void print\_addresses(stack\_cell \*a) – вывод текущих адресов списка

void print\_free\_addresses(free\_address \*a) – вывод освобожденных элементов списка

**Тестовые данные:**

|  |  |  |
| --- | --- | --- |
| **Описание** | **Вход** | **Выход** |
| Некорректная команда | А, -1, 15 | Некорректная команда |
| Некорректный ввод элементов стека | Ф, - | Некорректный элемент |
| Добавление элемента в заполненный стек | 2.3 | Стек переполнен |
| Удаление элемента из пустого стека | - | Стек пуст |
| Вывод стека на экран | - | 2.00000 6.40000 |

**Сравнение эффективности:**

* Добавление элемента в первый стек-массив выполняется за O(1). Добавление элемента в стек-список выполняется за O(1).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Размер** | **Первый стек-массив** | **Второй стек-массив** | **Список** | **Процент** |
| 1 | 22 | 22 | 42 | -90.1% |
| 10 | 22 | 22 | 44 | -100% |
| 100 | 22 | 22 | 39 | -77.3% |

* Удаление элемента из первого стека-массива выполняется за О(1). Удаление элемента из стека-списка выполняется за О(1).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Размер** | **Первый стек -массив** | **Второй стек-массив** | **Список** | **Процент** |
| 1 | 12 | 12 | 56 | -366.7% |
| 10 | 12 | 12 | 50 | -317.7% |
| 100 | 12 | 12 | 53 | -341.7% |

* Замеры памяти

|  |  |  |  |
| --- | --- | --- | --- |
| **Размер** | **Массив** | **Список** | **Процент для списка относительно массива** |
| 1 | 408 | 24 | 94.1% |
| 10 | 408 | 240 | 41.1% |
| 17 | 408 | 408 | 0% |
| 100 | 408 | 2400 | -488.2% |

В программе для двух стеков используется статический массив на 100 элементов. Выигрыш массива по памяти происходит при количестве элементов >17.

Стек-массив при добавлении работает в среднем в 2 раза быстрее, чем список, а при удалении в среднем в 3 раза быстрее.

**Вывод:** реализация стека-массива, работает быстрее, чем список, но массив проигрывает по памяти при количестве элементов <17. Стек-массив выигрывает список по памяти при количестве элементов >17. После просмотра адресов элементов списка была обнаружена фрагментация памяти.

**Ответы на вопросы:**

1. **Что такое стек?**

Стек – последовательный список переменной длины, в котором обработка элементов происходит с одного конца – с вершины.

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

Если хранить стек как список, то память выделяется в куче. Память ля каждого элемента стека-списка рассчитывается так: объем памяти для данных + адрес предыдущего элемента (4 или 8 байт).

Если хранить стек как массив, то память выделяется либо на стеке (статический массив), либо в куче (динамический массив). Память выделяется один раз и большое количество(с запасом).

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

Если стек хранится как статический массив, то указатель на последний элемент сдвигается на одну позицию к началу.

Если стек хранится как динамический массив, то указатель на последний элемент сдвигается на одну позицию к началу. Память освобождается при окончании работы с массивом.

Если стек хранится как список, то последний элемент удаляется при помощи освобождения памяти из-под него и указатель на последний элемент смещается на адрес элемента, записанного в освобожденном элементе.

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

При просмотре элементов стека есть возможность посмотреть только на последний элемент стека. Чтобы получить доступ к элементу необходимо удалить все предыдущие.

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

Эффективнее всего реализовывать стек с помощью динамического массива, так как он выигрывает и по времени обработки, и по занимаемой памяти.

Хранение стека в виде списка нужно использовать, если не известно количество элементов, для которого нужно выделять память.