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

По дисциплине «Типы и структуры данных»

Название Работа со стеком

Студент Миленко Николай Викторович
фамилия, имя, отчество

Группа ИУ7-34Б

Тип лабораторной работы Учебная

Название предприятия

Студент _____ Миленко Н.В.
подпись, дата *фамилия, и.о.*

Преподаватель Барышникова М.Ю.
_____
подпись, дата *фамилия, и.о.*

2022 г.

Содержание

Цель работы *- реализовать операции работы со стеком, который представлен в виде массива (статического или динамического) и в виде односвязного линейного списка; оценить преимущества и недостатки каждой реализации: получить представление о механизмах выделения и освобождения памяти при работе со стеком.*

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.

Распечатать убывающие серии последовательности целых чисел в обратном порядке.

Входные данные:

Номер команды, отвечающий за определённое действие.

Команды:

1. Добавить элементы в стек (из консоли);
2. Добавить элементы в стек (из файла);
3. Удалить элементы из стека;
4. Вывести текущее состояние стека (в случае задания стека списком выводится в том числе и массив адресов удаленных элементов);
5. Распечатать убывающие серии последовательности целых чисел в обратном порядке;
0. Выйти из программы.

Выходные данные:

Результат выполнения одной из пяти приведенных выше команд, а также вывод оценки эффективности в случае выбора соответствующего пункта меню.

Требования по работе с программой:

- взаимодействие с программой осуществляется с помощью меню

- каждое новый элемент в текстовом файле должно быть отделен от других пробелом

- количество записей не должно превышать 10000

Возможные аварийные ситуации:

- программа печатает сообщение об ошибке при некорректном вводе, при этом в сообщении указывается тип ошибки

- поведение программы при введении пользователем с консоли большего количества элементов, чем было ранее указано. Например, пользователь указал, что вводит 5 элементов, а попытался ввести 6.

Структуры данных:

Для реализации программы были описаны следующие структуры:

```
#ifndef STRUCTURES_H
#define STRUCTURES_H

#include "constants.h"

typedef struct stack_array
{
    int *ptr;
    int size;
} stack_array_t;

typedef struct elem_stack_list
{
    int elem;
    int num_elem;
    struct elem_stack_list *next;
} elem_stack_list_t;

typedef struct array_of_freed_areas
{
    elem_stack_list_t **array;
    int size;
} array_of_freed_areas_t;

#endif
```

Структура **stack_array** задает собой стек, реализованный массивом.

1) int *ptr – указатель на текущий элемент стека;

2) int size – кол-во заполненных элементов стека.

Структура **elem_stack_list** задает собой стек, реализованный списком.

1) int elem –текущий элемент стека;

2) int num_elem – кол-во заполненных элементов стека;

3) struct elem_stack_list *next – указатель на следующий элемент стека.

Структура array_of_freed_areas задает собой массив удаленных адресов.

Оценка эффективности:

Добавление элементов (в тиках, секундах, в данной таблице для полного времени обработки всех элементов)

Размер	Массив	Список
10	3814 0.0000016583	5545 0.0000024148
100	45798 0.0000199122	67768 0.0000294643
1000	332844 0.0001447148	432434 0.0001880148
10000	3405753 0.0008076243	4754518 0.0020671817

Удаление элементов (в тиках, среднее по кол-ву удалений, операции проводятся в цикле)

Размер	Массив	Список
10	119	2032
100	1038	18789
500	5060	70534
1000	8319	131583

Печать убывающих серий последовательности целых чисел в обратном порядке (в тиках, среднее по кол-ву удалений, операции проводятся в цикле)

Размер	Массив	Список
10	7359	5240
100	32620	28362
500	128051	118246
1000	232917	215386

Сравнение памяти

Размер	Массив	Список
10	40016	160
100	40016	1600
500	40016	8000
1000	40016	16000

Ответы на контрольные 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. Стек функционирует по принципу: LIFO - последним пришел – первым ушел.

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

Если хранить стек как список, то память выделяется в куче. Если хранить как массив — либо в куче, либо на стеке (зависит от того, динамически или статический массив используется). Для каждого элемента стека, который хранится как список, выделяется на 4 байта больше, чем для элемента стека, который хранится как массив. Данные байты используются для хранения указателя на следующий элемент списка.

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

Если хранить стек как список, то верхний элемент удаляется при помощи операции освобождения памяти для него и смещением указателя, который указывает на начало стека. Если хранить стек как массив, то смещается только указатель на начало стека.

4. Что происходит с элементами стека при его просмотре?

Элементы стека удаляются, так как каждый раз достается верхний элемент стека, чтобы посмотреть следующий.

5. Каким образом эффективнее реализовывать стек? От чего это зависит?

Стек эффективнее реализовать с помощью массива, так как он выигрывает в количестве занимаемой памяти (если массив динамический) и во времени обработки стека (добавлении и удалении элементов).

Хранение с помощью списка может выигрывать, если только стек реализован с помощью статического массива, так как в данном случае размер памяти под список ограничен размером оперативной памяти (хранится в куче), а для статического массива — ограничена размером стека.

Вывод

Если стек реализован статическим массивом, то добавление в него новых элементов будет происходить быстрее (~ в 3-5 раз), чем в стек, реализованный списком. Это связанно с тем, что для хранения стека в виде списка требуется выделить память для указателей на следующие элементы списка. Удаление элементов из массива происходит в 13-20 раз быстрее, чем из списка, так как при этом затрачивается время на очищение памяти, динамически выделенной под элемент стека.

Для вывода убывающих серий последовательности целых чисел в обратном порядке стек-список справился быстрее (~ на 8 – 15%) списка массива.

Вариант хранения стека в виде списка может выигрывать в том случае, если стек реализован статическим массивом (массив должен быть заполнен менее, чем на 25%). Так же, если не известен размер стека, то в таком случае стоит использовать списки (или динамические массивы).