|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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 ПО ЛАБОРАТОРНОЙ РАБОТЕ №4**

**«РАБОТА СО СТЕКОМ»**

Студент\_\_\_\_\_\_Чыонг Ван Хао\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*фамилия, имя, отчество*

Группа\_\_\_\_\_*ИУ7И-31Б\_\_\_\_\_\_\_\_\_\_\_\_\_\_*

Принял : Барышникова М. Ю.

**Описание условия задачи**

Создать программу работы со стеком, выполняющую операции добавление, удаления элементов и вывод текущего состояния стека. Реализовать стек:

а) массивом; б) списком.

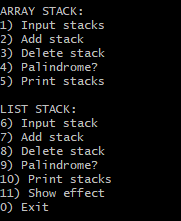
Все стандартные операции со стеком должны быть оформлены подпрограммами. При реализации стека списком в вывод текущего состояния стека добавить просмотр адресов элементов стека и создать свой список или массив свободных областей (адресов освобождаемых элементов) с выводом его на экран.

Используя стек, определить, является ли строка палиндромом.

**Техническое задание**

**Входные данные**

Программа для получения данных из вручную.



1. Реализация стека в виде массива
2. Добавить элемент в стек.
3. Удалить элемент из стека.
4. Определить, является ли строка палиндромом.
5. Показать стек.
6. Реализация стека в виде списка.
7. Добавить элемент в стек. (для стека-списка)
8. Удалить элемент из стека. (для стека-списка)
9. Определить, является ли строка палиндромом. (для стека-списка)

10) Показать текущее состояние стека с адресами и освобожденные адреса (для стека-списка).

11) Вывести эффективность.

0) выйти из программы

**Выходные данные**

1. Символы — текущие элементы стека.

2. Адреса — адреса текущих элементов стека или адреса свободных областей.

3. Временная характеристика — время работы функций.

4. Объемная характеристика — объем структуры данных

**Функция программы**

Операции работы со стеком — добавление элемента, удаление элемента, печать текущего состояния и адресов элементов, печать свободных областей, проверка строки на палиндром при помощи стека.

**Обращение к программе**

Программа запускается из терминала командой «./app.exe» в директории с  
программой.

**Возможные аварийные ситуации и ошибки пользователя**

1. Ввести неправильный выбор.

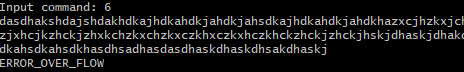
На выходе сообщение: «ERROR\_COMMAND.»



2. Переполнение стека.

На выходе сообщение:

« ERROR\_OVER\_FLOW»



3. Удаление элемента из пустого стека

На выходе сообщение:

« Stacks is empty!»

4. Вывод пустого стека

На выходе сообщение:

« ERROR\_EMPTY\_STACK »



5. Проверка на палиндром пустого стека.

На выходе сообщение:

« Stacks is empty!



6. Ввод уже созданного стека.

На выходе сообщение: «Stacks was created before!»





**Внутренняя структура данных**

Для реализации стека при помощи массива используется структура:

typedef struct

{

    int len;

    char array[MAX\_STACK + 1];

} stack\_arr\_t;

Её поля:

int len — текущее число элементов;  
char array[MAX\_STACK] — стек, где MAX\_STACK — его максимальный  
размер, равный 100.

Для реализации стека при помощи односвязного списка используется  
структура:

typedef struct node

{

    int len;

    char sym;

    struct node \*next;

} stack\_list\_t;

Её поля:  
int len — индекс элемента;  
char syml — символ элемента;  
struct node \*next — указатель на следующий элемент;

Для массива свободных областей используется структура:

typedef struct

{

    size\_t \*arr;

    int size;

} locate\_t;

Её поля:  
size\_t \*arr — массив адресов;  
int size — текущее число адресов;

**Функции программы**

void menu(void);

Описание: печать на экран параметров выполнения.

void message(int rc);

Описание: показать сообщение об ошибке

int input\_symbol(char \*sym);

Описание: введите 1 символ.

**Функции для работы со стеком, реализованном при помощи массива:**

int add\_symbol\_to\_arr(stack\_arr\_t \*array, const char sym);

Описание: добавить элемент в стек.

int array\_input(stack\_arr\_t \*array);

Описание: Реализация стека в виде массива.

char pop\_stack\_array(stack\_arr\_t \*arr);

Описание: Удалить элемент из стека.

int print\_array(stack\_arr\_t \*arr);

Описание: Показать стек.

int array\_is\_palindrome(stack\_arr\_t \*arr);

Описание: Определить, является ли строка палиндромом.

**Функции для работы со стеком, реализованном при помощи односвязного  
списка, и с массивом свободных областей:**

stack\_list\_t \*create\_node(const char sym);

Описание: инициализировать 1 элемент.

int add\_node\_to\_list(stack\_list\_t \*\*head, const char sym);

Описание: добавление элемента в стек.

int list\_input(stack\_list\_t \*\*head);

Описание: Реализация стека в виде списка.

char pop\_stack\_list(stack\_list\_t \*\*head, size\_t \*locate);

Описание: Удалить элемент из стека.

int print\_list(stack\_list\_t \*\*head, locate\_t address);

Описание: Показать стек.

int list\_is\_palindrome(stack\_list\_t \*\*head);

Описание: Определить, является ли строка палиндромом.

void free\_list(stack\_list\_t \*head);

Описание: освободить память стеков.

locate\_t \*create\_address(void);

Описание:создать список, содержащий освобожденные адреса.

void print\_address(const locate\_t address);

Описание: вывести список освобожденных адресов.

int free\_array\_address(locate\_t \*address);

Описание: освободить список, содержащий адрес.

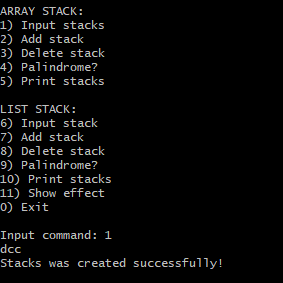
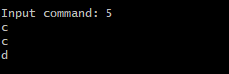
int time\_effect();

Описание: эффективность.

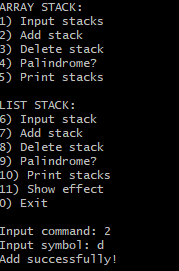
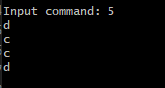
**Тесты**

а) массивом :

*Реализация стека в виде массива*

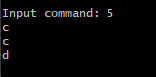
*Добавить элемент в стек.*

*Определить, является ли строка палиндромом.*

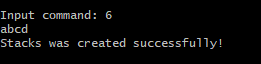
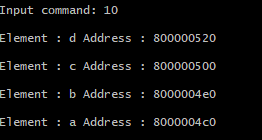
**

*Удалить элемент из стека.*

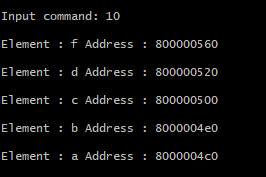
 

б) списком.

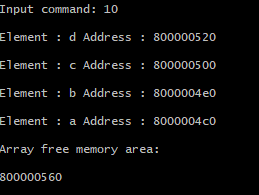
*Реализация стека в виде списка.*

*Добавить элемент в стек.*

*Удалить элемент из стека.*

*Определить, является ли строка палиндромом.*



**Оценка эффективности**

Время добавления элемента в стек (microsecs)

|  |  |  |
| --- | --- | --- |
| Кол-во элементов | Массив | Список |
| **400** | **32** | **62** |
| **700** | **46** | **98** |
| **1000** | **114** | **196** |

Время удаления элемента из стека

|  |  |  |
| --- | --- | --- |
| Кол-во элементов | Массив | Список |
| **400** | **1** | **2** |
| **700** | **2** | **3** |
| **1000** | **3** | **7** |

Время проверки строки на палиндром

|  |  |  |
| --- | --- | --- |
| Кол-во элементов | Массив | Список |
| **400** | **1** | **16** |
| **700** | **3** | **30** |
| **1000** | **4** | **52** |

Объем памяти(байты):

|  |  |  |
| --- | --- | --- |
| Кол-во элементов | Массив | Список |
| **400** | **1004** | **6400** |
| **700** | **1004** | **11200** |
| **1000** | **1004** | **16000** |

**Выводы**

Реализация стека с помощью массива на 2 раз выгоднее по времени и во много раз выгоднее по памяти, потому что в стеке-списке помимо данных char хранятся указатели. Тем не менее, реализация стека списком удобна, когда мы не знаем количество элементов в стеке.

**Контрольные вопросы**

1. **Что такое стек?**

Стек – последовательный список с переменной длиной, в котором включение и исключение элементов происходит только с одной стороны. Функционирует по Last In - First Out (LIFO).

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

При реализации стека списком, память выделяется динамически по мере добавления новых элементов; число элементов в стеке ограничено только количеством доступной ОП.

При реализации стека массивом, выделяется фиксированный участок памяти; в стеке не может быть больше заданного числа элементов. Добавление нового элемента происходит путём смещения индекса вершины.

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

При реализации списком память из-под элемента освобождается после его удаления.

При реализации массивом память из-под элемента не освобождается, происходит лишь изменение значения индекса вершины.

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

В общем случае доступ есть только к вершине стека; при просмотре она удаляется из стека, а указатель смещается далее. Для отображения состояния стека требуется последовательно проходить по всем его элементам, не «снимая» их.

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

Реализация стека массивом даёт огромный выигрыш во времени, поскольку не нужно каждый раз заново выделять и освобождать память. Тем не менее, в этом случае количество элементов в стеке жёстко ограничено – возможно либо переполнение стека, либо постоянный «излишек» памяти, отведённой под него.

Способ реализации напрямую зависит от условий решаемой задачи – нужно знать примерное число элементов, которые могут храниться в стеке; можно ли пренебрегать переполнением; ограничен ли объём памяти.