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 Равашдех Фадей Хешамович

Группа ИУ7 – 35Б

Преподаватель Никульшина Т. А.

2023 г.

Оглавление

Описание условия задачи

Цель работы: реализовать операции работы со стеком, который представлен в виде статического массива и в виде односвязного списка, оценить преимущества и недостатки каждой реализации, получить представление о механизмах выделения и освобождения памяти при работе с динамическими структурами данных.

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 отдельными подпрограммами. В случае реализации стека в виде списка при отображении текущего состояния стека предусмотреть возможность просмотра адресов элементов стека и создания дополнительного собственного списка свободных областей (адресов освобождаемой памяти при удалении элемента, который можно реализовать как списком, так и массивом) с выводом его на экран. Список свободных областей необходим для того, чтобы проследить, каким образом происходит выделение памяти менеджером памяти при запросах на нее и убедиться в возникновении или отсутствии фрагментации памяти.

Используя стек, определить, является ли строка палиндромом.

Описание технического задания

Входные данные:

Числовое значение номера пункта меню, строка добавляемых в стек символов, число удаляемых элементов из стека, проверяемая строка.

Выходные данные:

Текущее состояние стека на массиве и списке, адреса освобождённых областей памяти, является ли строка палиндромом.

Описание задачи, реализуемой программой

Реализация стека (представление в виде статического массиве и в виде списка) и операций со стеком (добавление, удаление, вывод), вывод освобожденных областей памяти, проврека строки на палиндром, сравнение представлений стека по эффективности.

Обращение к программе:

Запускается через терминал командой: ./app.exe.

Функции меню программы:

Информация о программе

1. Добавить элементы в стек на массиве
2. Удалить элементы из стека на массиве
3. Вывести текущее состояние стека на массиве
4. Добавить элементы в стек на списке
5. Удалить элементы из стека на списке
6. Вывести текущее состояние стека на списке
7. Вывести массив освобожденных областей
8. Определить, является ли строка палиндромом
9. Сравнение подходов
10. Выйти

Аварийные ситуации:

1. Ввод некорректного пункта меню;
2. Ошибка выделения памяти;
3. Введен символ окончания ввода;
4. Некорректный ввод.
5. Удаление элементов из пустого стека.
6. Ввод пустой строки в качестве паллиндрома.
7. Кол-во добавляемых на стек элементов превышает максимальное кол-во элементов на стеке.

Описание структур данных

```
// Длина статического массива
#define MAX_ARR_LEN 1000

// Структура стека на статическом массиве
typedef struct
{
     char array[MAX_ARR_LEN];// Статический массив символов
длиной
     char *pe;               // Указатель за конец массива
     char *sp;               // Указатель за текущий элемент
стека
} sa_stack_t;

// Структура стека на односвзяном списке
// Список
typedef struct list_t list_t;
struct list_t
{
    list_t *next;   // Указатель на следующий элемент списка
    char value;     // Значение
};
// Стек
typedef struct
{
    list_t *sp; // Указатель на текущий элемент стека
} ll_stack_t;
```

Описание функций

```
/**
 * @brief Контроллер
 *
 * @param option выбранный пользователем пункт меню
 * @param filename имя файла
 * @param table таблица
 * @param key_table таблица ключей
 * @return int код возврата
 */
int controller(int option, sa_stack_t *sa_stack, ll_stack_t
*ll_stack, fma_t *fma);
```

Операции со стеком

```
int sa_push(sa_stack_t *stack, char element);

int sa_pop(sa_stack_t *stack, char *element);

void sa_print(sa_stack_t *stack);
```

Операции со стеком

```
int ll_push(ll_stack_t *stack, char *element);

int ll_pop(ll_stack_t *stack, list_t **list_node);

void ll_print(ll_stack_t *stack);

void ll_free(ll_stack_t *stack);
```

Операции с массивом освобожденных областей

```
int da_add(fma_t *fma, list_t *p);
```

```
void da_print(fma_t *fma);

void da_free(fma_t *fma);
```

Функции, проверяющие, что строка в стеке является палиндромом.

```
bool sa_is_palindrome(sa_stack_t *word, size_t word_len);

bool ll_is_palindrome(ll_stack_t *word, size_t word_len);
```

Функции интерфейса

```
// Функции вывода меню и информации о программе
/**
 * @brief Вывод меню программы
 *
 */
void print_menu(void);

/**
 * @brief Вывод информации о программе
 *
 */
void print_info(void);

/**
 * @brief
 *
 * @param str
 * @return int
 * Указатель *str нужно будет очистить
 */
int get_input(char **str);

// Функции чтения из командной строки
/**
```

```
 * @brief Чтение строки определенной длины из стандартного потока ввода
 *
 * @param pch указатель на принимаемую строку
 * @param n длина строки
 * @return int код возврата
 */
int get_n_input(char pch[], size_t n);
/**
 * @brief Получение выбранного пользователем пункта
 *
 * @param option выбранный пункт
 * @return int код возврата
 */
int get_option(size_t *option);

// Функции вывода сообщений об ошибке
/**
 * @brief Вывод сообщения, поясняющего ошибку
 *
 * @param rc код возврата
 */
void print_error_msg(int rc);

/**
 * @brief Вывод сообщения, поясняющего ошибку, и возврат того же кода возврата
 *
 * @param rc код возврата
 * @return int код возврата
 */
int print_error(int rc);
```

Описание алгоритма

Добавление элемента в стек на массиве: элементу, на который указывает указатель стека, присваивается новое значение, сам указаетль перемещается вправо на размер элемента.

Удаление элемента из стека на массиве: указатель перемещается влево на размер элемента, в указатель на элемент помещается значение «удаленного» элемента.

Вывод стека на массиве: создается второй стек, в который перекладываются элементы из первого стека, выводя каждый перекладываемый элемент, затем элементы перекладываются обратно, чтобы привести первый стек к исходному состоянию (по значениям).

Добавление элемента в стек на списке: выделяется память под новый узел списка, в который помещается новое значение и указатель стека, указатель стека меняется на указатель на новый узел списка.

Удаление элемента из стека на списке: указателю на указаетль узла присваевается указатель на удаляемый элемент, указатель стека меняется на указтель на следующий элемента списка.

Вывод стека на списке: создается второй стек, в который перекладываются элементы из первого стека, выводя каждый перекладываемый элемент, затем элементы перекладываются обратно, чтобы привести первый стек к исходному состоянию (по значениям).

НАБОР ТЕСТОВ

№	Название теста	Пользовательский ввод	Вывод
1	Некорректный пункт меню	10	Некорректный номер пункта меню. Действия не существует. Ошибка: Действие с введенным пунктом отсутствует.
2	Некорректный пункт меню	abacaba	Ошибка: Некорректный ввод.
3	Удаление элементов из пустого стека на массиве	2	Стек пуст.
4	Удаление элементов из пустого стека на списке	5	Стек пуст.
5	Переполнение стека на массиве	1 (Кол-во элементов >1000)	Стек заполнен.
6	Переполнение стека при проверки слова на паллиндром	8 (Кол-во элементов >1000)	Стек заполнен.
7	Корректная проверка слова на паллиндром	8 123454321	За время t тактов было определено, что введенная строка 123454321 ЯВЛЯЕТСЯ палиндромом.
8	Отсутсвие освобождённых областей	7	Массив освобожденных областей памяти пуст.
9	Вывод освобождённых областей памяти	7	0x5555f34fed20 0x5555f34fed00 0x5555f34fece0

Оценка эффективности

Время в тактах за 100 повторений и память в байтах

Кол-во элементов	Операция (время в тактах)	Стек на массиве	Стек на списке
10	Добавление	100	133
	Удаление	95	124
	Палиндром	154	155
	Память (байт)	10016	168
100	Добавление	130	290
	Удаление	108	243
	Палиндром	196	413
	Память (байт)	10016	1608
1000	Добавление	169	640
	Удаление	151	474
	Палиндром	282	927
	Память (байт)	10016	16008
10000	Добавление	1604	6804
	Удаление	1420	4920
	Палиндром	2621	8920
	Память (байт)	10016	160008

Ответы на контрольные вопросы

1. Что такое стек? Стек – структура данных, работающая по принципу «последний пришёл – первый вышел». Т.е. включение и исключение элементов возможно только с одной стороны — вершины стека.

2. Каким образом и сколько памяти выделяется под хранение стека при различной его реализации? При реализации стека на статическом массиве выделяется статическая память, размер постоянен и не зависит от количества элементов в стеке, но зависит от установленного максимального количества элементов. При реализации стека на списке размер не фиксирован и новая память выделяется по мере необходимости для каждого нового элемента, а значит, что памяти выделится столько, сколько элементов в списке.

3. Каким образом освобождается память при удалении элемента стека при различной реализации стека? При удалении элемента со стека на статическом массиве длина статического массива уменьшается на 1, но сам «удаляемый» элемент остается на своем месте, в случае добавления нового элемента он будет перезаписан, т.е. память не освобождается. При удалении элемента со стека на списке указатель стека меняется на указатель на следующий узел списка, область памяти узла списка освобождается.

4. Что происходит с элементами стека при его просмотре? Стек не предусматривает возможность просмотра элементов, поэтому просмотр элементов стека выполняется при исключении их из стека, т. е. после просмотра стека он будет пуст.

5. Каким образом эффективнее реализовывать стек? От чего это зависит? Реализация стека на массиве быстрее реализации стека на списке в несколько раз. Стек на статическом массиве ограничен по количеству элементов, а стек на списке нет, но при этом стек на списке затрачивает больше памяти на единицу данных, т. к. каждый узел списка хранит не только данные, но и указатель на следующий узел. Поэтому, если заранее

известно, что количество элементов не превысит некий лимит статического массива, то эффективнее будет использовать стек на статическом массиве, иначе лучше использовать стек на списке, оперирующий динамической памятью и ограниченный лишь оперативной памятью компьютера.

Вывод

В результате выполнения этой лабораторной работы, я узнал о способах реалиации структуры стека на массиве и на списке, улучшил свои навыки работы со списками и контролем динамически выделенной памяти, узнал о различных методах аллоцирования памяти и фрагментации памяти.

Стек на статическом массиве в несколько раз быстрее стека на списке в опреациях добавления, удаления элементов из списка и проверки строки на палиндром. Так же можно заметить, что удаление элементов происходит быстрее чем их добавление на обеих реализациях стека. В случае стека на статическом массиве объем занимаемой памяти не зависит от количества элементов (т. к. память выделена статически), а в случае стека на списке объем занимаемой памяти зависит от количества элементов стека.

При этом в массиве данные типа «char» хранятся последовательно и каждый такой элемент занимает ровно 1 байт, а в списке каждый элемент типа «char» занимает целых 8 байт из-за выравнивания структур узлов списка в памяти, к этому еще добавляется 8 байт для хранения указателя, итого 16 байт. Если рассматривать общий случай типа данных, занимающего N байт, то массив будет занимать N*MAX_LEN (MAX_LEN — максимальное количество элементов в статическом массиве), а список будет занимать:

1. При N<8: 16 байт на элемент;
2. При N>8: 2*N байт на элемент (из-за выравнивания по большему полю).

Т. е. для получения экономии по памяти для стека на списке нужно, чтобы количество элементов было в два раза меньше предела массива для N>8 и в MAX_LEN*N/16 раз меньше предела массива для N<8.