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 ИУ

КАФЕДРА ИУ7

ОТЧЕТ ПО ЛР4 ТИПОВ И СТРУКТУР ДАННЫХ, Вариант 3

Студент Андреев Александр Алексеевич
фамилия, имя, отчество

Группа ИУ7-34Б

Тип практики учебная

Студент _____ _____
подпись, дата фамилия, и.о.

Преподаватель _____ _____
подпись, дата фамилия, и.о.

Оценка _____

9 октября 2020

Оглавление

1. Цель работы

В качестве цели работы ставится приобретение навыков реализации 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.

2. 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

3. Описание ТЗ, включающее внешнюю спецификацию

a. Описание исходных данных

Исходные данные программа получает из командной строки, где

- каждым элементом стека является цифра

Ограничения на записи во входном файле:

1. Максимальный размер стека 256 символов

b. Описание задачи, реализуемой программой

Программа должна уметь совершать следующие действия:

- При вводе выводить описание своей работы
- Выводить основное меню при входе / выходе в / из него, где расположен блоки выбора работы с массивом или стеком и завершения работы программы
- Выводить подпрограммное меню при при входе в него, где расположены блоки выхода в основное меню, добавление элемента в стек, удаление элемента из стека, проверка стека на палиндром, печать содержимого стека

- Указывать информацию, с каким типом памяти программа сейчас работает
- Выводить причина, почему не может быть произведена та или иная выбранная операция пользователем

с. Способ обращения программы

Скомпилированная программа запускается командой **“./a.out”** на Unix и **“./a.exe”** на Windows.

При запуске программа должна вывести описание и основное меню:

Рис. 1

```
===
Программа работы со стеком выполняет операции:
  - Добавления / Удаления элементов
  - Вывода текущего состояния стека
  - Проверку на палиндром числа
===
Основное меню программы:
  0 - Выход из программы
  1 - Работа с массивом
  2 - Работа со списком
```

Далее после ввода соответствующей команды программа должна выполнять команды пользователя.

0 - Программа завершает свою работу
1 - Программа переходит в режим работы со стеком, хранящимся массивом, выводит информацию о переводе и соответствующее меню (см. Рис. 2)
2 - Программа переходит в режим работы со стеком, хранящимся списком, выводит информацию о переводе и соответствующее меню (см. Рис. 3)

Рис. 2 (Вывод информации о переводе и меню при выборе работы со стеком, хранящимся массивом)

```
Выбрана работа со списком (2)...

        Меню подпрограммы работы со списком:
        0 - Вернуться в основное меню
        1 - Добавить символ в стек
        2 - Удалить символ из стека
        3 - Проверить на палиндром
        4 - Вывести содержимое
```

Пояснение к функциональности меню Рис. 2:
0 - Возвращает пользователя к основному меню программы
1 - Запрашивает у пользователя цифру для добавления или выводит информацию о том, то добавить цифру в стек невозможно
2 - Удаляет цифру из стека или выводит информацию о том, что это сделать невозможно.
3 - Выводит информацию о том, является ли текущий стек палиндромом, или указывает, что это сделать невозможно.
4 - Выводит содержимое стека или пишет, что это сделать невозможно.

Рис. 3 (Вывод информации о переводе и меню при выборе работы со стеком, хранящимся списком)

Выбрана работа с массивом (1)...

Меню подпрограммы работы с массивом:
0 - Назад
1 - Добавить символ
2 - Удалить символ
3 - Проверить на палиндром
4 - Вывести

Пояснение к функциональности меню Рис. 3:
0 - Возвращает пользователя к основному меню программы
1 - Запрашивает у пользователя цифру для добавления или выводит информацию о том, то добавить цифру в стек невозможно
2 - Удаляет цифру из стека или выводит информацию о том, что это сделать невозможно.
3 - Выводит информацию о том, является ли текущий стек палиндромом, или указывает, что это сделать невозможно.
4 - Выводит содержимое стека с таблицей адресов и данных об освобождении памяти или пишет, что это сделать невозможно

При наличии аварийных ситуаций и ошибок пользователя программа должна вывести соответствующее сообщение и не должна завершить свою работу абортно.

d. Описание возможных аварийных ситуаций и ошибок пользователя

В представленной ниже **Таблице 1** отражены действия программы при различных возможных, допущенных пользователем, при использовании основного меню программы ошибках.

Табл.1

Действие программы при различных ошибках		
№	Действие пользователя	Реакция программы
1	Введен несуществующий номер или строка пункта	Программа выведет сообщение

	меню	об ошибке “Ошибка ввода пункта меню. Повторите попытку.”

В представленной ниже **Таблице 2** отражены действия программы при различных возможных, допущенных пользователем, при использовании меню подпрограмм ошибках.

Табл.2

Действие программы при различных ошибках		
№	Действие пользователя	Реакция программы
1	Введен несуществующий номер или строка пункта меню.	Программа выведет сообщение об ошибке “Ошибка ввода пункта меню. Повторите попытку.”
2	Пользователь пытается добавить в стек, размер которого максимальный, новый элемент (STACK_MAX_SIZE = 256).	Программа выведет сообщение об ошибке “Стек полон...”
3	Пользователь пытается удалить элемент пустого стека.	Программа выведет сообщение об ошибке “Стек пуст...”
4	Пользователь просит вывести элементы пустого стека	Программа выведет сообщение об ошибке “Стек пуст...”
5	Пользователь пытается проверить пустой стек на палиндром	Программа выведет сообщение об ошибке “Стек пуст....”

4. Описание внутренних СД

При выборе пользователем хранения стека, как массива, его элементы хранятся следующим образом. (см. Рис. 4) Все элементы хранятся в структуре v_stack_s, где расположен целочисленный массив, и количество уже добавленных в него элементов.
STACK_MAX_SIZE = 256;

Рис. 4

```
struct v_stack_s
{
        int data[STACK_MAX_SIZE];
        int size;
};
```

При выборе пользователем хранения стека, как списка, элементы хранятся следующим образом. (см. Рис. 5) Каждый элемент последующего элемента списка представляет из себя структуру s_stack_s с значением и ссылкой на ранее добавленный элемент.

Рис. 5

```
struct s_stack_s
{
        int data;
        struct s_stack_s *next;
};
```

5. Алгоритм

При запуске программа выводит описание и основное меню (Рис. 6):

Рис. 6

```
===
Программа работы со стеком выполняет операции:
  - Добавления / Удаления элементов
  - Вывода текущего состояния стека
  - Проверку на палиндром числа
===
Основное меню программы:
  0 - Выход из программы
  1 - Работа с массивом
  2 - Работа со списком
```

Далее, пользователю выводится приглашение для ввода пункта меню (Рис. 7):

Рис. 7

```
Укажите цифру из доступных (0-2):
```

После чего пользователь выбирает пункт меню. В случае некорректного выбора пункта - пользователь получает сообщение об ошибке.

Далее пользователь работает с выбранным меню.

Функции

Работа со стеком, как массивом

void v_push(struct v_stack_s *v_stack)
Добавляет в конец стека элемент с увеличением размера на единицу, если стек имеет меньше максимально допустимого размера.

void v_delete(struct v_stack_s *v_stack)
Удаляет последний элемент стека и уменьшает размер на единицу, если стек не пуст.

void v_is_palindrom(struct v_stack_s *v_stack)
Проверяет стек на палиндром, если стек не пуст.

void v_print(struct v_stack_s *v_stack)
Создает дублирующий массив стека, который заполняет с конца, после чего последовательно выводит элементы массива без пробела, если стек не пуст.
Если стек пуст выводит сообщение Стек пуст…”.

int scanf_v_menu_code()
Выводит приглашение о вводе и считывает цифру выбранного пункта меню, передавая сообщение об ошибке в v_menu(...);

int v_menu(struct v_stack_s *v_stack)
Получает сообщение о чтение и выбирает, какая функция должна дальше выполняться в соответствии с указанной пользователем цифрой.

void v_operations()
Выводит сообщение о том, что выбран массив, как хранение, далее до получения ключа о возврате в основное меню выполняет в цикле работе v_menu(...);

Работа со стеком, как списком

int get_s_size(struct s_stack_s* head)
Получени размер существующего стека.

void append_node(struct s_stack_s headRef, int num)**
Добавляет в существующий стек новую запись.

void s_push(struct s_stack_s *s_stack)
Верифицирует размер стека и выполняет, если стек не полон, операцию **void append_node(struct s_stack_s** headRef, int num)**

int del(struct s_stack_s ** head, int index)
Непосредственное удаление последнего элемента стека.

void s_delete(struct s_stack_s *s_stack)
Верификация размера стека с дальнейшим удалением последнего элемента, если стек не пуст.

int get_elem(struct s_stack_s **headRef, int elem_index)
Получение значения i-го элемента стека.

void s_is_palindrom(struct s_stack_s *s_stack)
Проверка стека на палиндром, если он не пуст.

void s_print(struct s_stack_s **headRef)
Вывод содержимого стека на экран, если он не пуст.

int scanf_s_menu_code()
Выводит приглашение о вводе и считывает цифру выбранного пункта меню, передавая сообщение об ошибке в v_menu(...);

int s_menu(struct s_stack_s *s_stack)
Получает сообщение о чтение и выбирает, какая функция должна дальше выполняться в соответствии с указанной пользователем цифрой.

void s_operations()
Выводит сообщение о том, что выбран список, как хранение, далее до получения ключа о возврате в основное меню выполняет в цикле работе s_menu(...);

6. Набор тестов с указанием, что проверяется

В представленных ниже **Таблица 2** отражены тестирование устойчивости работы консольного меню программы и демонстрация устойчивости работы программы к различному типу выполняемых с ней операций пользователем соответственно.

Табл. 2

Тестирование устойчивости работы программы			
№	Ввод пользователя	Реакция программы	Что проверяется данной операцией?
1	Ввод НЕКОРРЕКТНОГО пункта основного меню или меню подпрограммы	Программа выводит “Ошибка ввода пункта меню. Повторите попытку...”	Устойчивость программы к вводу неверной команды меню
2	Вводе НЕ цифры в блоке добавления элемента	Программа выводит “Вы ввели некорректное число...”	Устойчивость программы к вводу некорректного значения
3	Пользователь пытается добавить в стек, размер которого максимальный, новый элемент (STACK_MAX_SIZE = 256).	Программа выводит “Введенный файл оказался пустым или не может быть прочитан. Исправьте и повторите попытку.”	Устойчивость к работе с полным стеком
4	Пользователь пытается удалить элемент пустого стека.	Программа выведет сообщение об ошибке “Стек полон...”	Устойчивость к работе с пустым стеком
5	Пользователь просит вывести элементы пустого стека	Программа выведет сообщение об ошибке “Стек пуст...”	Устойчивость к работе с пустым стеком
6	Пользователь пытается проверить пустой стек на палиндром	Программа выведет сообщение об ошибке “Стек пуст...”	Устойчивость к работе с пустым стеком
5	Укажите цифру из доступных (0-2): 1 Выбрана работа с массивом (1)... Меню подпрограммы работы с массивом: 0 - Назад 1 - Добавить символ 2 - Удалить символ 3 - Проверить на палиндром 4 - Вывести	Программа выведет стек и аналитику “12321 Время удаления всех элементов массива: 0.000003 сек Время добавления всех элементов массива: 0.000001 сек Память: 24 байт”	Проверка корректности отображений и получения данных, используя массив.

	Укажите цифру из доступных (0-4): 1 Введите любую цифры (Число от 0 до 9), чтобы добавить в Стек: 1 Меню подпрограммы работы с массивом: 0 - Назад 1 - Добавить символ 2 - Удалить символ 3 - Проверить на палиндром 4 - Вывести Укажите цифру из доступных (0-4): 1 Введите любую цифры (Число от 0 до 9), чтобы добавить в Стек: 2 Меню подпрограммы работы с массивом: 0 - Назад 1 - Добавить символ 2 - Удалить символ 3 - Проверить на палиндром 4 - Вывести Укажите цифру из доступных (0-4): 1 Введите любую цифры (Число от 0 до 9), чтобы добавить в Стек: 3 Меню подпрограммы работы с массивом: 0 - Назад 1 - Добавить символ 2 - Удалить символ 3 - Проверить на палиндром 4 - Вывести Укажите цифру из доступных (0-4): 1 Введите любую цифры (Число от 0 до 9), чтобы добавить в Стек: 2		

	Меню подпрограммы работы с массивом: 0 - Назад 1 - Добавить символ 2 - Удалить символ 3 - Проверить на палиндром 4 - Вывести Укажите цифру из доступных (0-4): 1 Введите любую цифры (Число от 0 до 9), чтобы добавить в Стек: 1 Меню подпрограммы работы с массивом: 0 - Назад 1 - Добавить символ 2 - Удалить символ 3 - Проверить на палиндром 4 - Вывести Укажите цифру из доступных (0-4): 3 Число в стеке - Палиндром. Меню подпрограммы работы с массивом: 0 - Назад 1 - Добавить символ 2 - Удалить символ 3 - Проверить на палиндром 4 - Вывести Укажите цифру из доступных (0-4): 4		
6	Укажите цифру из доступных (0-2): 2 Выбрана работа со списком (2)... Меню подпрограммы работы со списком: 0 - Вернуться в основное меню 1 - Добавить символ в стек 2 - Удалить символ из стека 3 - Проверить на палиндром 4 - Вывести содержимое	Программа выведет стек и аналитику “Освобождено и удалено: 0x7f9377000030, 1 0x7f9377000020, 2 0x7f9377000010, 1 Стек: 121 Время удаления всех элементов стека: 0.000007 сек Время добавления всех элементов стека: 0.000000 сек Память: 60 байт”	Проверка корректности отображений и получения данных, используя стек.

	Укажите цифру из доступных (0-4): 1 Введите любую цифры (Число от 0 до 9), чтобы добавить в Стек: 1 Меню подпрограммы работы со списком: 0 - Вернуться в основное меню 1 - Добавить символ в стек 2 - Удалить символ из стека 3 - Проверить на палиндром 4 - Вывести содержимое Укажите цифру из доступных (0-4): 1 Введите любую цифры (Число от 0 до 9), чтобы добавить в Стек: 2 Меню подпрограммы работы со списком: 0 - Вернуться в основное меню 1 - Добавить символ в стек 2 - Удалить символ из стека 3 - Проверить на палиндром 4 - Вывести содержимое Укажите цифру из доступных (0-4): 1 Введите любую цифры (Число от 0 до 9), чтобы добавить в Стек: 1 Меню подпрограммы работы со списком: 0 - Вернуться в основное меню 1 - Добавить символ в стек 2 - Удалить символ из стека 3 - Проверить на палиндром 4 - Вывести содержимое Укажите цифру из доступных		

	(0-4): 4		

7. Выводы по проделанной работе

По результатам проделанной работы я научился реализовывать стек двумя способами, стеком и списком.

Также стоит отметить, что работать со стеком-массивом эффективнее, чем работать со стеком-списком. По результатам тестирования видно, что при добавлении и удалении элемента массивы значительно быстрее списков из-за отсутствия необходимости выделения/освобождения памяти под каждый элемент.

При реализации стека списком элементы располагаются в определенных участках памяти. Также стоит отметить, что фрагментации памяти не наблюдается.

По данным анализа видно, что время, затрачиваемое при работе со списком на добавление увеличивается быстрее из-за того, что приходится аллоцировать память.

Удаление оказывается также быстрее, потому что при его выполнении приходится освобождать память.

Сравнительный анализ						
Кол-во элементов	Массив			Список		
	Добавление	Удаление	Память	Добавление	Удаление	Память
10	0.000002 сек	0.000002 сек	44 байт	0.000001 сек	0.000002 сек	200 байт
100	0.000002 сек	0.000003 сек	404 байт	0.000008 сек	0.000065 сек	2200 байт
1000	0.000004 сек	0.000004 сек	4004 байт	0.000065 сек	0.000006 сек	22200 байт

8. Контрольные вопросы

1. Что такое стек?
   Стек – это структура данных, в которой элементы поддерживают принцип LIFO (“Last in – first out”): последним вошел – первым вышел. Или первым вошел – последним вышел. **Стек** позволяет хранить элементы и поддерживает, обычно, базовые операции: добавление элемента на вершину стека и его снятие.

2. Каким образом и сколько памяти выделяется под хранение стека при различной его реализации?
   При массиве n * 2 Байта + 2 Байта
   При списке n * 10 Байт

3. Каким образом освобождается память при удалении элемента стека при различной реализации стека?
   В стеке удобнее, что можно освободить сразу последний элемент и не совершать переписывание элементов, как в целочисленном массиве.

4. Что происходит с элементами стека при его просмотре?
   Их приходится снимать с вершины и перекладывать в копию для того, чтобы потом снова положить в стек.

5. Каким образом эффективнее реализовывать стек? От чего это зависит?
   Если происходят постоянные незначительные по количеству от общего размера стека операции, то выгоднее использовать списки, потому что не будет прохода по всему массиву для полного снятия и последующего освобождения.
   Такие операции позволяют использовать меньше памяти.
   А если же операции над стеком происходят значительные относительно его размера, то лучше использовать массив.