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»

Название: Работа со стеком

Студент Пискунов Панте

Группа ИУ7 – 36Б

Тип лабораторной работы: Учебная

Вариант № 1

Преподаватель Барышникова Марина Юрьевна

2022 г.

Описание условия задачи

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.

Описание технического задания

Распечатайте убывающие серии последовательности целых чисел в обратном порядке.

Входные данные:

1. **Целое число, представляющее собой номер команды**:

   целое число в диапазоне от 0 до 6.

2. **Командно-зависимые данные:**

целочисленные значения (элементы стека)

Выходные данные:

Результат выполнения определенной команды.

Описание структуры данных

Реализация стека с помощью линейного односвязного списка.

```
// Структура стека в виде списка

typedef struct node_t
{
	int data; // данные стека
	struct node_t *next; // указатель на следующая ячейка памяти
}node_t;

// Структура стека в виде массива
typedef struct array_stack_t
{
	int *data; // массив
	size_t top; // верхный узел
	size_t size; // длина массива
}array_stack_t;
```

Меню программы:

1 - Добавить элемент в стек.

2 - Удалить элемент из стека.

3 - Распечатайте убывающие серии последовательности целых чисел в обратном порядке.

4 - Вывести стек.

5 - Сравнить время.

6 — Заполнить стек автоматически.

0 – Выйти из программы.

Обращение к программе:

Запускается через терминал с помощью команды ./app.exe.

Аварийные ситуации:

1. Некорректный ввод номера команды.

   На входе: число, большее чем 5 или меньшее, чем 0.

   На выходе: сообщение «Ошибка: Неправильный ввод!»

2. Совершение вывод последовательности при пустом стеке.

   На входе: целое число в диапазоне от 4 до 5 (номер команды).

   На выходе: сообщение «Стек пустой!»

3. Добавление элемента в стек, когда он заполнен полностью.

   На входе: элемент стека.

   На выходе: сообщение «Ошибка: Произошло переполнение стека»

Описание алгоритма

Исключение элемента для стека-списка реализуется переходом к предыдущему элементу и освобождением памяти из-под удаляемого элемента. Для стека-массива исключение элемента - сдвиг указателя стека на одну позицию назад.

Добавление элемента для стека-списка реализуется выделением памяти под новый элемент и связыванием его с концом списка. Для стека-массива добавление элемента реализуется сдвигом указателя стека на одну позицию вперед, а затем записью очередного значения добавляемого элемента.

НАБОР ТЕСТОВ

	Название теста	**Пользовательский ввод**	**Результат**
1	Некорректный ввод команды	-1	Ошибка: Неправильный ввод !
2	Добавление элемента в заполненный стек	Команда 1	Ошибка: Произошло переполнение стека !
3	Совершение вывод последовательности при пустом стеке	Команда 3	Стек пустой !
4	Добавление элемента в стек	Команда 1	Элемент успешно добавлен в стек
5	Удаление элементов из стека	Команда 2	Элементы успешно удалены из стека
6	Вывод текущего состояния стеков на экран	Команда 4	Все стеки выведены на экран + массив свободных указателей
7	Вывод результата, когда в последовательности нет последовательности	Команда 3	Нет последовательности!
8	Вывод сравнения памяти и объема	Команда 5	Статистика

Оценка эффективности

Измерения поиска убывающей последовательности будет производиться в секундах.

Размер	Массив	Список
10	0.000019	0.000031
100	0.000088	0.000100
250	0.000315	0.000338
500	0.000596	0.000686

Оценка эффективности по памяти

Размер	Массив	Список
10	40	160
100	400	1600
250	1000	4000
500	2000	8000

Из данных таблиц можно сделать вывод, что стек реализованный массивом всегда выигравает по памяти и по времени сортировки. По памяти лучше работает в 4 раза.

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 или 8 байт (если брать современные ПК) больше, чем для элемента стека, который хранится как массив.
Данные байты использованы для хранения указателя на следующий элемент списка. (из-за этого либо 4 либо 8 байт)

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.
При хранении стека как массив, память освобождается при завершении программы.

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 и во времени обработки стека.

В стеке, реализованном с помощью массива, легче просматривать элементы, сдвигая указатель, а также не надо хранить указатель на следующий элемент, как в списке.

Вывод

Если стек реализовать массивом, то он будет выигрывать. Это связанно с тем, что для хранения стека в виде списка требуется память, чтобы хранить указатели.
Так же при реализация стека в виде массива требуется меньше времени на обработку. Это связанно с тем, что при реализации массивом доступ к нужному элементу получить проще, требуется лишь передвинуть индекс. Если реализовать в виде списка, то требуется время для удаления верхнего элемента (верхушки стека) и перестановки указателя.

Можно сделать вывод, что для хранения стека лучше использовать массив, это выгоднее и по памяти и по времени.

Так же можно сделать вывод, что в результате тестирования фрагментации памяти не выявлено.