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 (ИУ)

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии (ИУ7)

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №4

«Работа со стеком»

Студент, группа **Рядинский К.В., ИУ7-33Б**

2020 г.

Условие задачи

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.

Техническое задание

1) 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.
2) Распечатайте убывающие серии последовательности целых чисел в обратном порядке.

Входные данные

Пункт меню:

1. Ввод последовательности — целые числа
2. Печать массива-стека
3. Добавление элемента в массив-стек — целое число
4. Печать списка-стека
5. Добавление элемента в список-стек — целое число
6. Нахождение убывающих подпоследовательностей (см тз)
7. Печать помощи
8. Очистка экрана
9. Считывание последовательности из файла — путь к файлу
10. Выход из программы

Выходные данные

Подпоследовательности, выведенные в обратном порядке, время выполнение для разных стеков, потребление памяти.

Возможные аварийные случаи

1. Неправильный ввод
2. Пустой файл
3. Переполнение стека

Тесты

Ввод	Вывод
Ввод последовательности: 1 2 3	Не найдено подпоследотвальности
Неправильный размер: a	Введен неправильный размер
Неправильный элемент стека: b	Введен неправильный элемент
Превышен размер стека (100 элементов)	Переполнение стека

Структуры данных

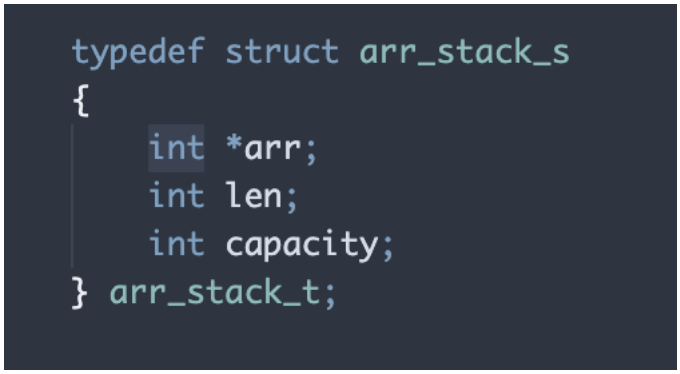
Структуры узла списка

```
5
6   typedef struct node_s
7   {
8       int val;
9       struct node_s *next;
10  } node_t;
11
```

Структура стека-списка

```
typedef struct list_stack_s
{
    node_t *head;
} list_stack_t;
```

Структура стека-массива



```
typedef struct arr_stack_s
{
    int *arr;
    int len;
    int capacity;
} arr_stack_t;
```

Функции

Int POP(stack) — удаление верхнего элемента из стека

Int APPEND(stack) — добавление целого числа в стек

Void CLEAR(stack) — очистка стека

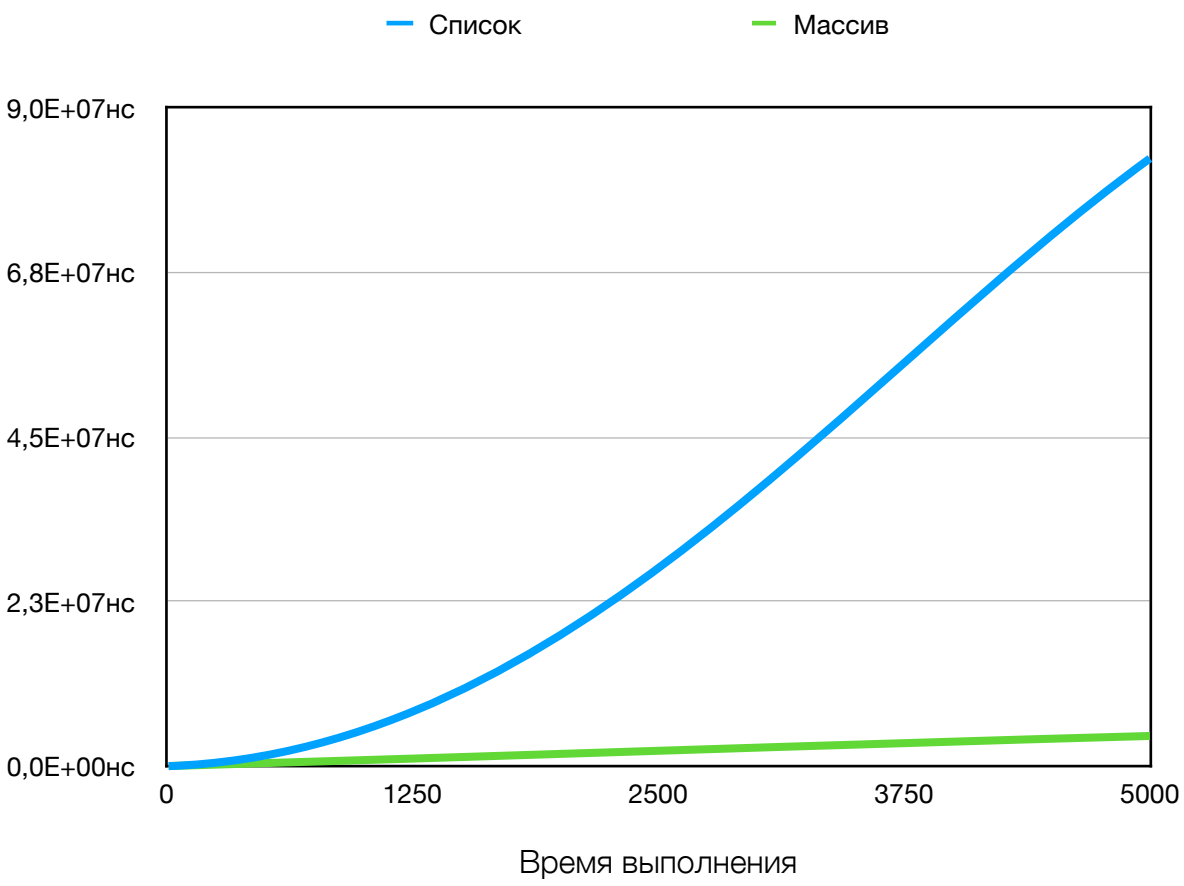
Void PRINT(stack) — печать стека

Void CPY(src, dst) — копирование стека *src* в *dst*

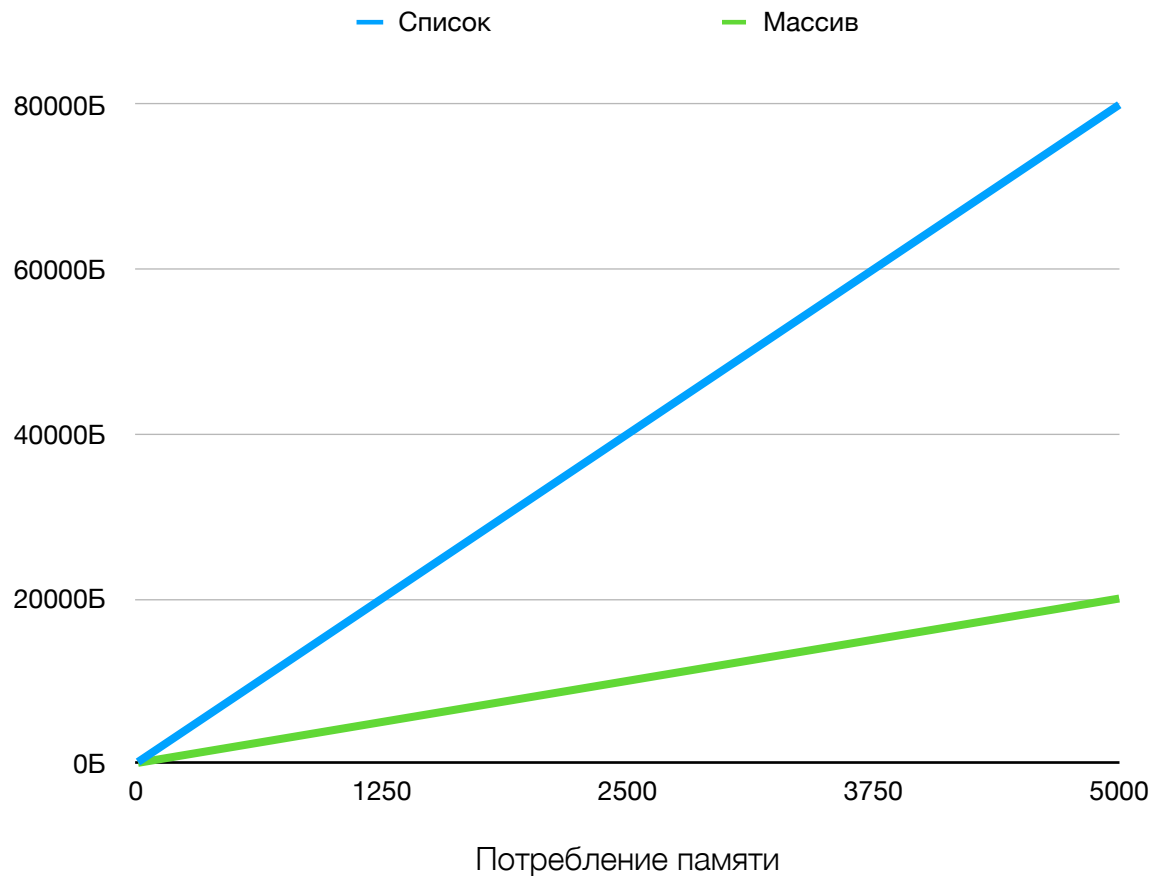
Сравнение времени

Размер	Стек-список нс	Память стек-список Б	Стек-массив нс	Память стек-массив Б
10	25000	144	34000	40
20	35000	304	42000	80
30	63000	464	67000	120
50	92000	784	73000	200
100	214000	1584	125000	400
500	1437000	7984	577000	2000
1000	4882000	15984	1189000	4000
5000	83065000	79984	4146000	20000

Ниже приведены графики



Время выполнения



Потребление памяти

Вывод

По проведенным тестам можно сделать вывод, что стек-массив примерно в несколько раз быстрее стека-списка на больших данных. При очень малом кол-ве элементов стек-список работает быстрее. Также список использует больше памяти. По графикам видно, что список требует примерно в 4 раза больше памяти.

Стек-список работает медленнее потому, что приходится выделять память каждый раз, когда добавляется новый «узел» в список и очищать память, когда удаляется «узел». В стеке-массиве же мы единожды выделяем память и далее только смещаем указатель на текущий элемент стека.

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 принципу «последний зашел – первый вышел».

2. **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.

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

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

При реализации списком память из-под элемента освобождается при его удалении.

При реализации массивом память из-под элемента не освобождается, происходит лишь изменение значения указателя на последний элемент.

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

Мы храним только указатель на последний элемент, чтобы пройтись по стеку нужно обойти все элементы стека по указателям на предыдущий элемент.

5. **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.
При реализации стека списком возникает меньше забот с памятью.