*Государственное образовательное учреждение высшего профессионального образования*

*«Московский Государственный Технический Университет имени Н. Э. Баумана»*

ОТЧЕТ

По лабораторной работе №3

По курсу «Типы и структуры данных»

Тема: Стеки

Вариант №2

|  |  |
| --- | --- |
| Студент: | Щербатюк Д.С. |
| Группа: | ИУ7-34 |
| Преподаватель: | Силантьева А. В. |

**Техническое задание**

**Задача**: 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.

При реализации стека массивом располагать два стека в одном массиве. Один стек располагается в начале массива и растет к концу, а другой располагается в конце массива и растет к началу. Заполнять и освобождать стеки произвольным образом с экрана. Элементами стека являются вещественные числа. Списком реализовать один стек.

**Входные данные**: вещественные числа; команды добавления элемента в стек и снятия вершины.

**Выходные данные**: список элементов стека; список адресов, освобождённых в результате снятия вершины; среднее время, затрачиваемое на одну операцию.

**Функция программы**: реализация работы со стеком; демонстрация распределения памяти при работе со стеком; анализ времени, требующегося на добавление и удаление элементов при различной реализации.

**Аварийные ситуации**

1) Попытка добавить в стек элемент, не являющийся числом

Будет выведено сообщение об ошибке.

2) Попытка снять вершину пустого стека.

Будет выведена информация о том, что стек пуст.

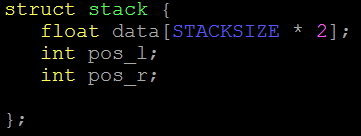
3) Переполнение стека

При реализации списком, максимальное число элементов в стеке: 20 (задано в программе через #define); Если при добавлении будет превышено число элементов, элемент не добавится в стек и выведется сообщение об ошибке.

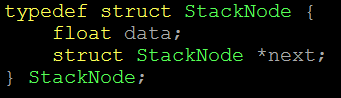
При реализации массивом, если индексы вершин стеков исчисляются из середины, при пустом стеке индексы равны нулю. Будет выведено сообщение о переполнении, элемент добавлен не будет.

**Используемые структуры**

Стек-массив



Стек-список



**Функции и алгоритмы**

* **Реализация стека-листа.**

**Добавление элемента в стек.**

**int push(StackNode \*\*top, float data);**

Возвращает код ошибки или 0

\*top – указатель на вершину

Data – добавляемые значения

Если стек полон

Вывести ошибку о переполнении

Выделить память под буферный узел

Буферный узел ссылается на data и указывает на вершину

\*top = буферный узел

**Снятие вершины стека.**

**float pop(StackNode \*\*top);**

Возвращает данные в вершине

\*top – указатель на вершину

Если стек пустой

Вывести ошибку

Буферный узел = \*top

Данные = данные в буферном узле

\*top указывает на следующий элемент после буфера (NULL)

- - - - - - - - -

* **Реализация двухстороннего стека**

**Добавление элемента в двусторонний стек.**

**int push\_l(float num, float stack[], int pos, int side);**

Возвращает новую позицию

Num – записываемое число

Stack[] - стек

Pos – позиция

Side - строна

Если side = правая

Если pos <= максимальной позиции справа

Выдать ошибку о переполнении

Иначе

Pos--

Стек[pos] = num

Макс позиция слева --

Если side = левая

Если pos <= - размер массива

Выдать ошибку о переполнении

Иначе

Pos++

Стек[pos] = num

Макс позиция справа ++

**Снятие вершины двустороннего стека.**

**int pop\_l(float \*num, float stack[], int pos, int side);**

Возвращает новую позицию.

Num – данные в вершине

Stack[] - стек

Pos – позиция

Side – сторона

Еесли pos = 0

Выдать ошибку о снятии вершины с пустого стека

Иначе

Если side = правая

num = Стек[позиция справа]

Стек[позиция справа] = 0

Pos—

Если side = левая

num = Стек[позиция слева]

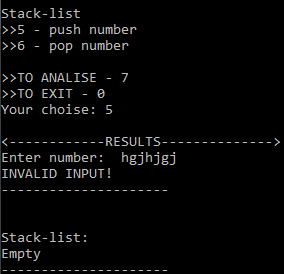
Стек[позиция слева] = 0

Pos++

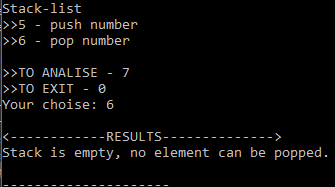
**Тесты**

**Реализация списком.**

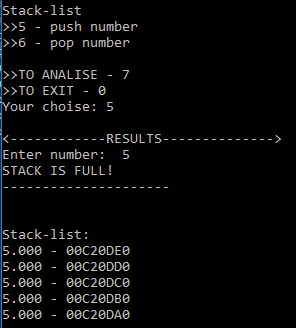
Попытка добавить в стек элемент, не являющийся числом.



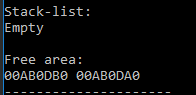
Попытка снять вершину пустого стека.

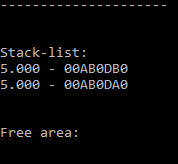


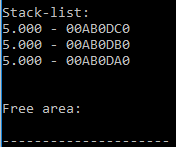
Переполнение стека

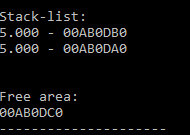


Наполнение зоны свободных адресов и выделение памяти из другой области.





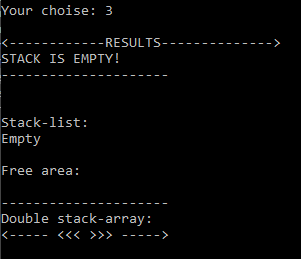




Фрагментации памяти не обнаружено.

**Реализация массивом.**

Переполнение одного из стеков.



**Сравнение реализации.**

Время для 100 000 элементов:



Делаем вывод, что реализация статическим массивом быстрее, однако она затратная по памяти.

Реализация статическим массивом выгодна, когда мы знаем точное количество элементов, а реализация списком подойдет для всего остального.



**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).

*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; 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.

*3. 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.

*4. 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.

*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 – 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.