Отчет по лабораторной работе № 4 по дисциплине

“Типы и структуры данных”

Сушина А.Д. ИУ7-31б

Работа № 4

Работа со стеком

Цель работы: реализовать операции работы со стеком, который

представлен в виде массива (статического или динамического) 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.

# 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.

Элементы стека - слова.Распечатать слова в обратном порядке.

# Описание ТЗ

### Общая концепция системы

Программа выполняет операции над стеком, реализованным двумя способами. Стек вводится пользователем. Сначала вводится количество элементов стека, затем сам стек.

Основная задача - вывести стек в обратном порядке. Элементами стека являются слова.

### Требования к функциональным характеристикам

Программа должна выполнять следующие функции:

* Добавление элемента в стек для двух реализаций
* удаление элемента из стека для двух реализаций
* печать стека в обратном порядке
* вывод времени работы программы
* указание формата и диапазона вводимых данных,
* блокирование ввода данных, неверных по типу,
* указание операции, производимой программой:

1. добавление элемента в стек,
2. удаление элемента из стека,
3. вычисление (обработка данных);

* наличие пояснений при выводе результата.

На **вход** программа получает длину стека, затем № действя. При необходимости слово, как составляющую стека.

**Выход** должен быть представлен в виде строки. Также программа должна вывести данные о времени выполнения алгоритма.

Программа должна выдавать корректный ответ при вводе любых данных. Если произошла ошибка ввода, программа должна сообщить об этом.

## Аварийные ситуации

* Ввод несуществующего пункта меню  
  Программа выведет сообщение “Input error”.
* Ввод значений, не удовлетворяющих условиям.

(Длина стека больше 100, длина строки больше 30)

Программа выведет сообщение “Input error”.

* Переполнение стека

программы выведет FULL STACK

* Пустой стек   
  программа выведет EMPTY STACK

## Способ обращения к программе

Программа представляет собой файл app.exe. Запускается в консоли. для запуска достаточно команды ./app.exe. Если файл отсутствует можно собрать его с помощью утилиты make.

# Описание структур данных

struct j - структура единицы стека

{

char i[30+1]; - слово

struct j \*next; - указатель на предыдущий

};

struct j \*head = NULL; - голова стека

struct j \*headmax = NULL; - максимальный адрес головы

long int tb, te; - переменные для замера времени

long int t1, t2; - время выполнения первого и второго алгоритмов

int action, rc; - переменные кода ошибки и выбора действия

char buf[31]; - переменная для записи слова

int nmax, n = 0; - число элементов стека

char stek1[100][31]; - стек в виде массива

char (\*p)[31]; - указатель на элемент стека в массиве

struct j \*full[100] = {NULL}; - массив заполненных адресов

struct j \*empty[100] = {NULL}; - массив пустых адресов

# Описание алгоритма

* Пока не выход

1. Добавить элемент   
   Если стек не полон:   
   Добавить элемент в стек   
   Вывести массив занятых адресов
2. Удалить элемент  
   Если стек не пуст:  
   Удалить элемент из стека  
   вывести массив свободных адресов
3. Напечатать в обратном порядке  
   Переписать элементы в другой стек   
   напечатать полученный стек  
   напечатать время выполнения операции
4. Выход  
   выйти из цикла

# Тесты

1.Несуществующий пункт меню  
Вход:

Способ: 4

Вывод: Input error

2. Некорректный ввод n (n < 0)

Вход:

n: 0

Вывод: Input error

3. Переполнение стека   
Вход:

n : 0

1 ooo

1

Вывод: FULL STACK

4. удаление элемента из пустого стека

Вход:

2

Вывод: EMPTY STACK

5. нормальный вход

вход:

aa

bb

cc

Выход:

cc

bb

aa

# Оценка эффективности

|  |  |
| --- | --- |
| Добавление элемента Масиив t, такты | Добавление элемента  Список t, такты |
| 123 | 3609 |
| Удаление элемента  Массив  t, такты | Удаление элемента  Список  t, такты |
| 43 | 3179 |
| Память на 1 эл стека Массив | Память на 1 эл стека Список |
| 31 | 36 |

# Выводы по проделанной работе

Была реализована программа, печатающая слова в обратном порядке с использованием стека.  
Реализованы функции добавления и удаления элемента для двух способов реализации стека.

Для массива функции удаления(на 98%) и добавления(на 96%) элемента работают быстрее, чем для списка. Для хранения элемента массива используется меньше памяти(на 16%), чем для хранения структуры. Таким образом, функции работы с стеком, реализованным как массив, эффективнее по памяти и по времени, чем функции для стека-списка.

# Ответы на 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.

2. Каким образом и сколько памяти выделяется под хранение стека при

различной его реализации?

При реализации стека массивом, память выделяется только под элемент массива. (Например, 30 байт для строки из 30 элементов). При реализации стека списком, память выделяется не только под сам элемент, но и под указатель на предыдущий элемент стека.

3. Каким образом освобождается память при удалении элемента стека при

различной реализации стека?

При реализации стека с помощью списка, освобождается память из-под элемента списка. При реализации списка массивом, смещается указатель, память освобождается только в конце работы со стеком.

4. Что происходит с элементами стека при его просмотре?

Элементы стека удаляются при его просмотре.

5. Каким образом эффективнее реализовывать стек? От чего это зависит?

Стек эффективнее реализовывать с помощью массива. Этот способ эффективнее по памяти и по времени.