Условие 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 массив свободных областей (адресов освобождаемых элементов) с выводом его на экран.

Техническое задание

Реализовать операции работы со стеком, который представлен в виде массива и в виде списка, оценить преимущества и недостатки каждой реализации. Найти путь в лабиринте с помощью стека.

Входные данные

Пункты меню (при ручной работе со стеком), количество добавляемых и удаляемых элементов, карта с лабиринтом.

Выходные данные

Текущее состояние стека, массив свободных областей, карта с лабиринтом и найденный путь в ней.

Возможные аварийные ситуации

Некорректный ввод, невозможность найти путь в лабиринте.

Структуры данных

Структура хранения лабиринта

```
typedef struct maze
{
   char **matrix;
   int x;
   int y;
   int i_enter;
   int j_enter;
} maze_t;
```

matrix – указатель на матрицу, хранящую сам лабиринт
x, y – количество столбцов и строк матрицы
i_enter, j_enter – индексы, откуда начинается поиск пути

Структура стека, реализованного связанным списком:

```
typedef struct list_element
{
    int i;
    int j;
    int direction;
    struct list_element *next_elem;
} list_element_t;
```

i, j – индексы клетки, из которой можно продолжить путь
direction – направление пути
next_elem – указатель на следующий элемент связанного списка

```
typedef struct list
{
    list_element_t *ptr;
} stack_list_t;
```

ptr – указатель на первый элемент списка

Структура стека, реализованного массивом:

```
typedef struct array_elem
{
    int i;
    int j;
    int direction;
} array_element_t;
```

i, j – индексы клетки, из которой можно продолжить путь
direction – направление пути

```
typedef struct array
{
    array_element_t *ptr;
    int size;
} stack_array_t;
```

ptr – указатель на начало массива
size – размер массива

Алгоритм

Сначала формируется матрица, в которой хранится лабиринт. Если клетка, в которой в данный момент находится алгоритм, является развилкой, то эта клетка помещается в стек. Если алгоритм попадает в тупик, то из стека достается верхний элемент, и движение продолжается оттуда, то есть из последней развилки. Если стек пуст, то это признак того, что найти путь к выходу невозможно.

Тесты

Время

Добавление элементов

Количество элементов	Список	Массив
10	12500	3650
100	85000	28000
1000	600000	290000

Удаление элементов

Количество элементов	Список	Массив
10	7500	3150
100	37000	25200
1000	380000	250000

Печать стека

Количество элементов	Список	Массив
10	240000	210000
100	1500000	1490000
1000	12500000	13000000

Занимаемая память

Количество элементов	Список	Массив
10	240	120
100	2400	1200
1000	24000	12000

Выводы по проделанной работе

Стек, реализованный связанным списком, проигрывает как по памяти, так и по времени обработки. Таким образом, можно сделать вывод, что если нужно реализовать такую структуру данных как стек, то лучше использовать массив, а не связанны список.

Контрольные вопросы

Что такое стек?

Стек – структура данных, в которой можно обрабатывать только последний добавленный элемент (верхний элемент). На стек действует правило LIFO — последним пришел, первым вышел.

Каким образом и сколько памяти выделяется под хранение стека при различной его реализации?

При хранении стека с помощью списка, то память всегда выделяется в куче. При хранении с помощью массива, память выделяется либо в куче, либо на стеке (в зависимости от того, динамический массив или статический). Для каждого элемента стека, реализованного списком, выделяется на 4 или 8 байт (на большинстве современных ПК) больше, чем для элемента массива. Эти дополнительные байты занимает указатель на следующий элемент списка. Размер указателя (4 или 8 байт) зависит от архитектуры.

Каким образом освобождается память при удалении элемента стека при различной реализации стека?

При хранении стека связанным списком, верхний элемент удаляется путь освобождением памяти для него и смещения указателя, указывающего на начало стека. При удалении из стека, реализованного массивом, смещается лишь указатель на вершину стека.

Что происходит с элементами стека при его просмотре?

Элементы стека уничтожаются, так как каждый раз достается верхний элемент стека.

Каким образом эффективнее реализовывать стек? От чего это зависит?

Реализовывать стек эффективнее с помощью массива. Он выигрывает как во времени обработки, так и в количестве занимаемой памяти. Вариант хранения списка может выигрывать только в том случае, если стек реализован статическим массивом. В этом случае, память для списка ограничена размером оперативной памяти (так как память выделяется в куче), а память для статического массива ограничена размером стека.