Оберган Татьяна ИУ7-35Б

Вариант 18

**Лабораторная работа №4**

Стек

**Цель работы:** 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.

**Условие задачи (вар 4):**

Проверить правильность расстановки скобок трех типов (круглых, квадратных и фигурных) в выражении

**Входные данные:**Элементы стека или выражение.

**Выходные данные:**Флаг, указывающий на корректность выражения.

**Функция программы:**Реализация операции работы со стеком; определение правильности расстановки скобок в выражении.

**Возможные ошибки:**

1. Переполнение стека
2. Ввод элемента неправильного типа

**Обращение к программе** осуществляется через консоль.

**Структуры данных:**

Структура узла для списка:

**struct** node  
{  
 **char** value;  
 **struct** node \*previous\_el;  
};

Реализация массива-стека:

**char** arrStack[**N**];

**Функции:**

Взаимодействие со стеком-массивом:

**int** push\_arr(**char** \*\*p\_curr, **char** \*p\_end, **char** value);  
**int** pop\_arr(**char** \*\*p\_curr, **char** \*p\_beg);  
**void** print\_arr(**char** \*p\_beg, **char** \*p\_curr);

Взаимодействие со стеком-списком:

**struct** node\* pop\_list(**struct** node \*last\_el, **int** fl\_an);  
**struct** node\* push\_list(**struct** node \*last\_el, **char** value, **int** fl\_an);  
**void** print\_list(**struct** node \*last\_el);

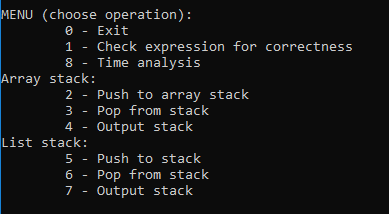
Анализ времени:

**void** analise();

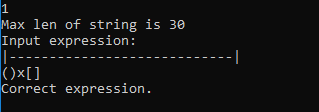
Анализ расстановки скобок:

**int** check\_parentheses();

**Интерфейс:**Главное меню:



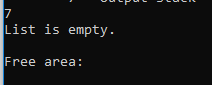
Проверка расстановки скобок:



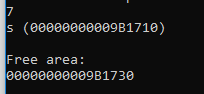
Ввод в стек:



Вывод пустого стека:

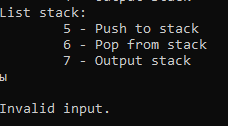


Вывод списка-стека:



**Тесты:**

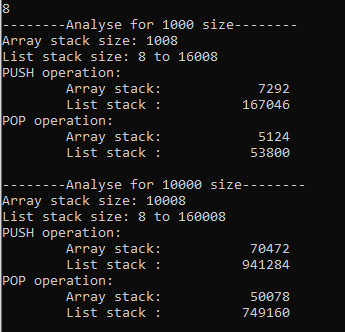
Вызов некорректного пункта меню:



Переполнение стека:



**Анализ эффективности (по памяти и времени):**



Работа с массивом происходит в 10 раз быстрее, чем со списком.

При этом в наихудшем случае список будет занимать в 16 раз больше памяти, чем массив, в наилучшем – 8 байт (место под нулевой указатель).

**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.

**Вывод:**

Для хранения маленьких стеков целесообразнее использовать массив т.к. он дает заметный выигрыш по времени. При размытом количестве элементов или когда максимальное количество элементов гораздо больше среднего целесообразнее использовать список.