Практическое занятие № 11 «Стек»

Учебные цели:

- получение умений и навыков работы со стеком.

Воспитательные цели:

- воспитание познавательного интереса, активности, целеустремленности, настойчивости, активности, наблюдательности, интуиции, сообразительности;
- формировать диалектико-материалистическое мировоззрение;
- формировать навыки самостоятельности и дисциплинированности;
- стимулировать активную познавательную деятельность обучаемых, способствовать формированию у них творческого мышления.

Категория слушателей: 2,3 курс.

Время: 90 мин.

Место проведения: дисплейный класс.

Материально-техническое обеспечение:

1) персональный компьютер *IBM PC* с операционной системой Windows XP; 2) среда разработки приложений *Visual C++.NET*.

ПЛАН ЛАБОРАТОРНОГО ЗАНЯТИЯ

Учебные вопросы	Время, мин
Вступительная часть	5
1. Структура данных − стек. Основные сведения	15
2. Выполнение индивидуального задания	65
Заключительная часть	5

Элементы теории

Стек − такой последовательный список с переменной длиной, включение и исключение элементов из которого выполняются только с одной стороны списка, называемого вершиной стека. Применяются и другие названия стека - магазин и очередь, функционирующая по принципу LIFO (Last - In - First- Out - "последним пришел - первым исключается"). Примеры стека: винтовочный патронный магазин, тупиковый железнодорожный разъезд для сортировки вагонов.

Основные операции над стеком - включение нового элемента (английское название push - заталкивать) и исключение элемента из стека (англ. pop - выскакивать).

Полезными могут быть также вспомогательные операции:

- определение текущего числа элементов в стеке;
- очистка стека;
- неразрушающее чтение элемента из вершины стека, которое может быть реализовано, как комбинация основных операций:

x:=pop(stack); push(stack,x);

Некоторые авторы рассматривают также операции включения/исключения элементов для середины стека, однако структура, для которой возможны такие операции, не соответствует стеку по определению.

Для наглядности рассмотрим небольшой пример, демонстрирующий принцип включения элементов в стек и исключения элементов из стека. На рис. 1 (а-ж) изображены состояния стека:

- а). пустого;
- б-г). после последовательного включения в него элементов с именами 'A', 'B', 'C';
- д, е). после последовательного удаления из стека элементов 'C' и 'B';
- ж). после включения в стек элемента 'D'.

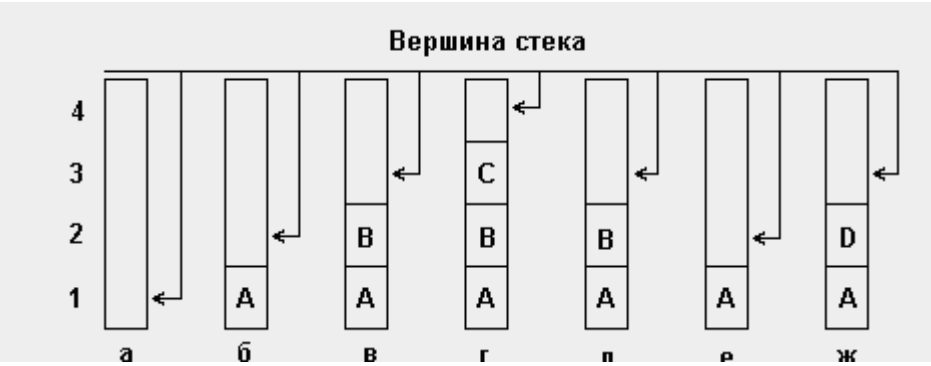

Рис. 1: Включение и исключение элементов из стека

Как видно из рис. 1, стек можно представить, например, в виде стопки книг (элементов), лежащей на столе. Присвоим каждой книге свое название, например A,B,C,D... Тогда в момент времени, когда на столе книг нет, про стек аналогично можно сказать, что он пуст, т.е. не содержит ни одного элемента. Если же мы начнем последовательно класть книги одну на другую, то получим стопку книг (допустим, из n книг), или получим стек, в котором содержится n элементов, причем вершиной его будет являться элемент n+1. Удаление элементов из стека осуществляется

аналогичным образом т. е. удаляется последовательно по одному элементу, начиная с вершины, или по одной книге из стопки

Вопросы для самопроверки

1. Дайте определение понятию стек?
2. Перечислите основные операции над стеком.
3. На базе каких структур может быть организован стек?
4. Приведите из жизни примеры организации чего-либо по принципу стека.
5. В чем сходство и отличие динамических структур данных типа список и стек?
6. Можно ли добраться до середины или конца («дна») стека, минуя его начало («вершину»)?

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ОТРАБОТКЕ УЧЕБНЫХ ВОПРОСОВ

Задание 1. Реализуйте структуру данных «стек», реализовав все указанные здесь методы. Напишите программу (функцию `main()`), содержащую описание стека и моделирующую работу стека. Функция `main()` считывает последовательность команд и в зависимости от команды выполняет ту или иную операцию. После выполнения одной команды программа должна вывести одну строчку. Возможные команды для программы:

- **1 - Push n -** добавить в стек число n (значение n задается после команды). Программа должна вывести `ok`.
- **2 - Pop -** удалить из стека последний элемент. Программа должна вывести его значение.
- **3 - Back -** программа должна вывести значение последнего элемента, не удаляя его из стека.
- **4 - Size -** программа должна вывести количество элементов в стеке.
- **5 - Clear -** программа должна очистить стек и вывести *ok*.
- **6 - Exit -** программа должна вывести *bye* и завершить работу.

Гарантируется, что набор входных команд удовлетворяет следующим требованиям: максимальное количество элементов в стеке в любой момент не превосходит 100, все команды `pop_back` и `back` корректны, то есть при их исполнении в стеке содержится хотя бы один элемент.

Пример протокола работы программы:

Команда	Ввод с клавиатуры	Вывод

```
push 2          1 2          ok
push 3          1 3          ok
push 5          1 5          ok
back            3 0           5
size            4 0           3
pop             2 0           5
size            4 0           2
push 7          1 7          ok
pop             2 0           7
clear           5 0          ok
size            4 0           0
exit            6 0         bye
```

Указания к выполнению работы.

Интерфейс программы должен быть понятен неподготовленному пользователю. При разработке интерфейса программы следует предусмотреть:

- указание формата и диапазона вводимых данных;
- блокирование ввода данных, неверных по типу;
- указание операции, производимой программой;
- наличие пояснений при выводе результата.

Кроме того, нужно вывести на экран время выполнения программы при реализации стека списком и массивом, а также указать требуемый объем памяти.

При тестировании программы необходимо:

- проверить правильность ввода и вывода данных (в том числе, отследить попытки ввода данных, неверных по типу);
- обеспечить вывод сообщений при отсутствии входных данных («пустой ввод»);
- проверить правильность выполнения операций;
- обеспечить вывод соответствующих сообщений при попытке удаления элемента из пустого стека;
- отследить переполнение стека.

При реализации стека в виде списка необходимо:

- ограничить доступный объем оперативной памяти путем указания:
- максимального количества элементов в стеке;
- максимального адреса памяти, превышение которого будет свидетельствовать о переполнении стека;
- следить за освобождением памяти при удалении элемента из стека.

Контрольные вопросы

1. Что такое стек?
2. Каким образом и сколько памяти выделяется под хранение стека при различной его реализации?
3. Каким образом освобождается память при удалении элемента стека при различной реализации стека?
4. Что происходит с элементами стека при его просмотре?
5. Каким образом эффективнее реализовывать стек? От чего это зависит?
6. Оформите в виде процедуры Push занесение в стек значения.
7. Оформите в виде процедуры Pop извлечение значения из стека.