**РАЗДЕЛ 1. ОБЩИЕ ПОЛОЖЕНИЯ**

Связный список (linked list) – это структура данных, в которой объекты расположены в линейном порядке. Однако, в отличие от массива, в котором этот порядок определяется индексами, порядок в связанном списке определяется указателями на каждый объект. Связанные списки обеспечивают простое и гибкое представление динамических множеств и поддерживают все типичные операции над динамическими множествами (хотя и не всегда достаточно эффективно).

Однонаправленный (односвязный) список – это структура данных, представляющая собой последовательность элементов, в каждом из которых хранится значение и указатель на следующий элемент списка ([рис. 1.1](http://www.intuit.ru/studies/courses/648/504/lecture/11456?page=1#image.29.1)). В последнем элементе указатель на следующий элемент равен NULL.



Рис. 1.1. Линейный однонаправленный список

Описание простейшего элемента такого списка выглядит следующим образом:

struct имя\_типа { информационное поле; адресное поле; };

где

информационное поле – это поле любого, ранее объявленного или стандартного, типа;

адресное поле – это указатель на объект того же типа, что и определяемая структура, в него записывается адрес следующего элемента списка.

Например:

struct Node {

int key;//информационное поле

Node\*next;//адресное поле

};

Информационных полей может быть несколько. Например:

struct point {

char \*name; //информационное поле

int age; //информационное поле

point \*next;//адресное поле

};

Каждый *элемент списка* содержит *ключ*, который идентифицирует этот элемент. *Ключ* обычно бывает либо целым числом, либо строкой.

Основными операциями, осуществляемыми с однонаправленными списками, являются:

* создание списка;
* печать (просмотр) списка;
* вставка элемента в список;
* удаление элемента из списка;
* поиск элемента в списке
* проверка пустоты списка;
* удаление списка.

Особое внимание следует обратить на то, что при выполнении любых операций с линейным однонаправленным списком необходимо обеспечивать позиционирование какого-либо указателя на первый элемент. В противном случае часть или весь список будет недоступен.

В процессе реализации линейного односвязного списка будем придерживаться следующего порядка действий:

а) Прежде всего, определим две структуры данных:

1. Структура, содержащая характеристики окружности (координата x, координата y, радиус). Назовём эту структуру Data;
2. Структура, содержащая поле типа Data и поле – адрес последующего элемента next. Вторую структуру назовём List.

Исходный текст этих структур расположим в файле List\_Header.h.

struct Data

{

double coord\_x;

double coord\_y;

double rad;

};

struct List

{

Data d;

List \*next;

};

Такой подход позволит в дальнейшем изменять в широких пределах структуру с собственно данными, никак не затрагивая при этом основную структуру List.

б) В функции main() определим указатель на начало будущего списка:

List \*u = NULL;

Пока список пуст, и указатель явно задан равным константе NULL.

Для удобства заведём также вспомогательную переменную-указатель, которая будет хранить адрес последнего элемента списка:

List \*x;

Пока список пуст, последний элемент списка совпадает с его началом.

в) Выполним первоначальное заполнение списка.

Для заполнения списка в данной работе предусмотрены две функции – функция добавления узла в начало и функция добавления узла в конец списка. Возможные вызовы данных функций - Add\_top(&u, &x) и Add\_end(&u, &x) соответственно.

Дальнейшая работа с односвязным списком будет осуществляться посредством функций, реализованных в рамках данной курсовой работы. Для более удобной работы с ними используется дополнительная функция выбора действия.