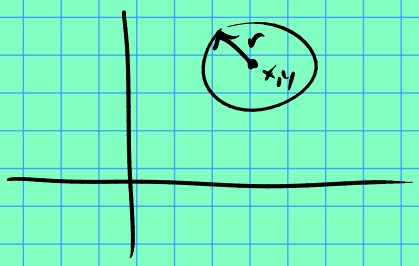
Structs / classes.

≈ like an array, but

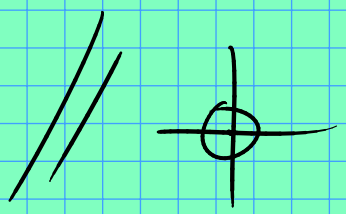
① heterogeneous datatypes for elements

② use names not numbers for elements.

E.g. to store a circle in the plane:



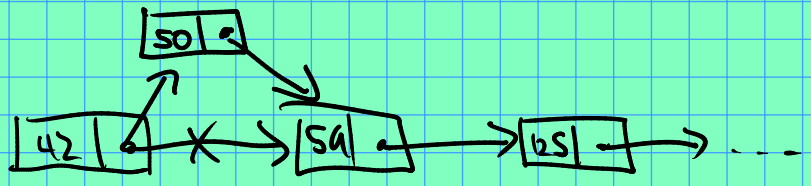

```
struct circle {
    double x; // x coord
    double y; // y coord
    float r;  // radius
};

// Now circle is a new
//                datatype.

circle c;
c.x = 0;
c.y = 0;
c.r = 1;
```



```
circle C[10]; // 10 circles!
C[0].x = 0;
```

C[0] type is circle.

Lists.



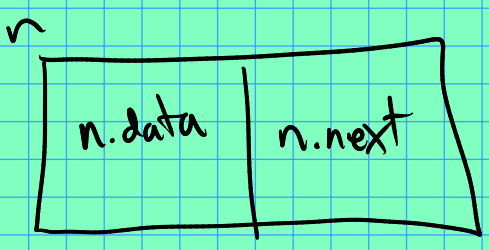
Trade offs w/ arrays or vectors:

Vector/Array	List
fast "random access" V[i]	Bad at random access: finding ith element requires linear scan
Insert at front or middle is slow (costs ≈ n steps if n = # elements)	Fast to insert at front.

How to implement in C/C++?

```
struct node {
    int data;
    node* next;
};
```

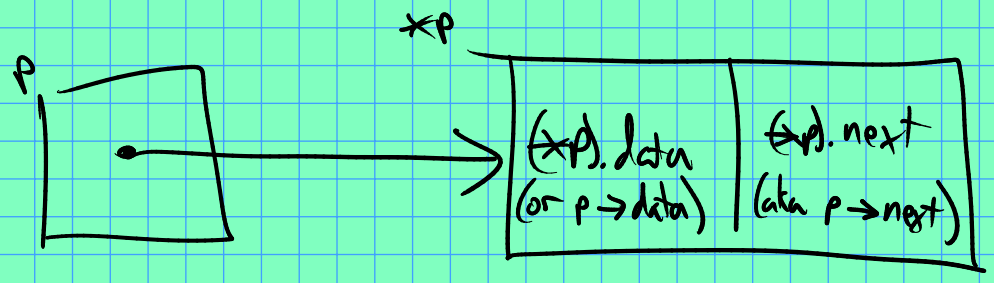
If we declare one as `node n;`, this is the picture:



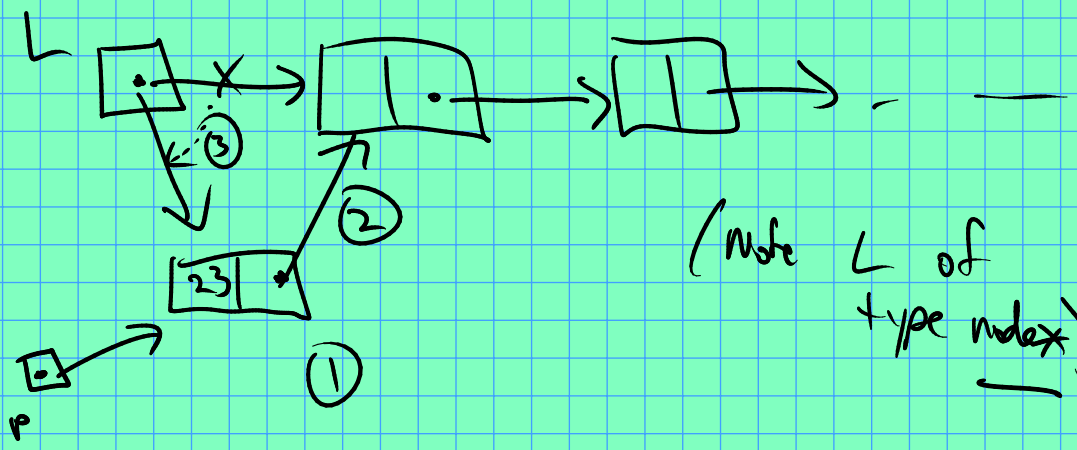
More common to dynamically allocate:

```
node* p = new node;
```

picture:



Add node to beginning:



(Node L of type node*)

```
node* p = new node; // ①
p->data = 23;  // also ①
p->next = L; // ②
L = p; // ③
```