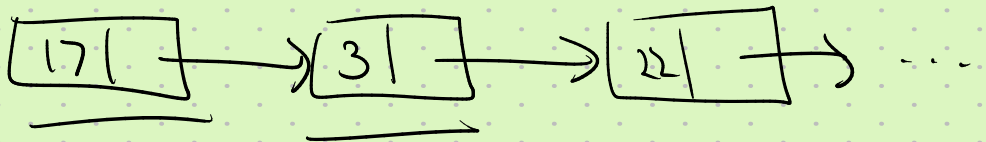
Linked Lists (l4.pdf, 2nd half)

Idea, in pictures:



Store a list... but each element tells you where to find the next one.

(contrast w/ vector: next element is @ (&e)+1...)

Why?

Pros

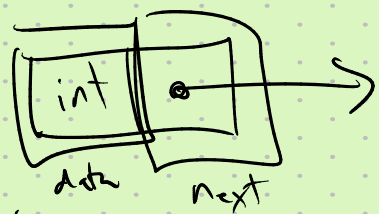
+ can add a new element anywhere in constant # of steps
+ removing elements from any location is also fast (does not depend on total # of elements)

Cons

- No "random access". i.e, no V[i]!
- Takes more space: each element takes an 8-byte pointer with it.
- Not friendly for CPU cache.

How to do it in C(++)?

Might be nice to have a datatype for the elements:



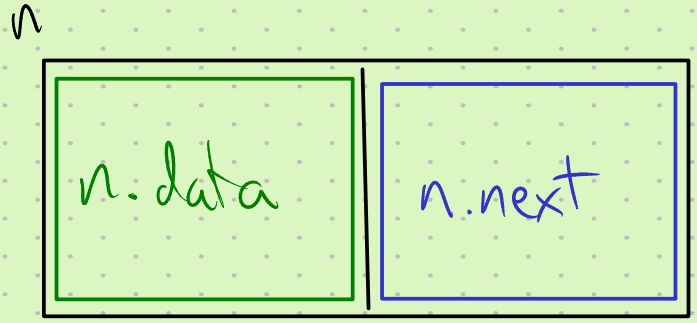
We can use structures:

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

Code

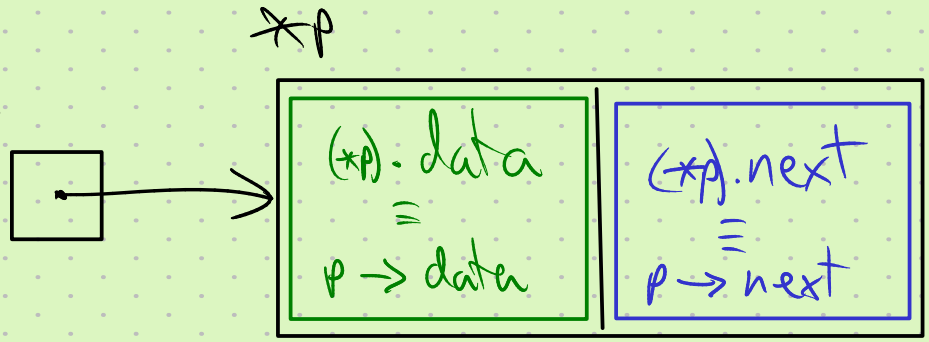
```
node n;
```

pictures



More commonly:

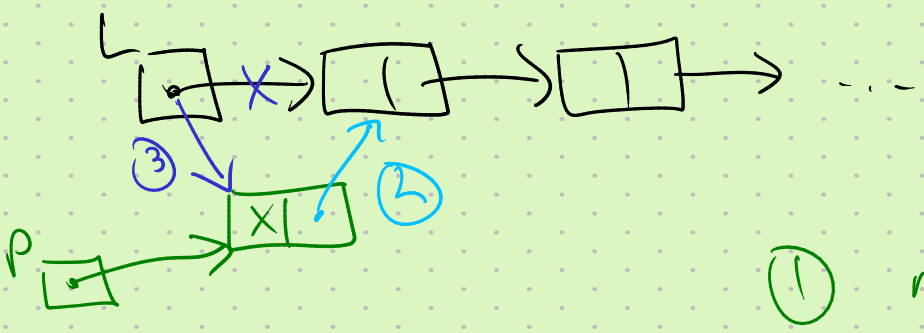
```
node * p = new node;
```



Exercise: add new node to beginning of an existing list.

Note: we'll keep track of a special pointer (for List) that always tells us where

the list begins.

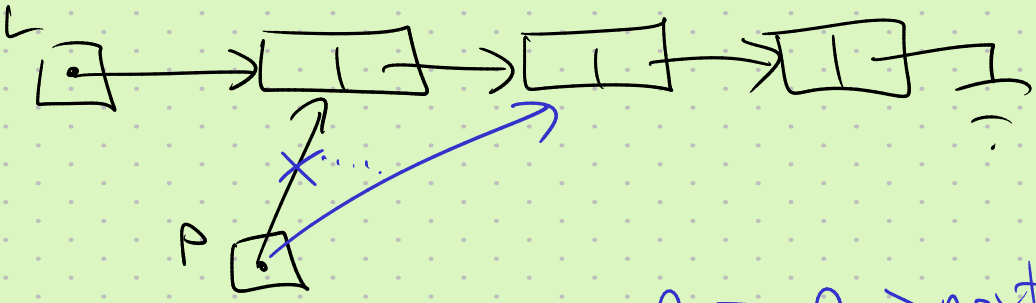


① node* p = new node;
p->data = x;

② p->next = L;

③ L = p;
// make L point to
// the same place as
// p points

Exercise: print contents of a list.



Notes:
⏚ ≡ NULL

p = p->next;