CS 102
Introduction to Programming Using C++

Chapter 7

Pointers

Homework

- Written homework
- R7.1, 2, 4, 7, 8, 15, 16, 19, 21 to 27

- Programs
- p. 344, choose one of P7.1, 2, or 4.
- Also, implement a linked list.

Pointers on Their Own

- When you use a pointer, it must already point to something
- Until now, they always pointed to variables
- New idea: They can point to other things, not just variables
  - We can let our pointers point to a space in memory
  - This is a lot more flexible than the examples we have seen before

Managing the Computer's Resources

- Suppose you have five programs running
  - Perhaps Word, a game, a browser, Windows Explorer, your C++ IDE or an editor
- You click on one of them
- How should the program you click on know that you clicked on it?
- Should every program watch every mouse movement, every mouse click, every keystroke?

The Operating System

- That would make everything incredible slow
- That would be incredibly bug-prone
  - What if a program accidentally interpreted another program's mouse click as its own?
- Instead, there is a program that's in charge
  - It's the operating system (OS)
- The operating system is software that controls the computer's resources
  - Some of these resources are memory, the mouse, the hard disk, your flash drive

Running a Program: The OS in Action

- When you run a program
  - The OS finds some memory for the program
  - The OS loads the program from the disk into that memory
  - It "connects" the program to its resources
  - It starts the program
  - Then it lets the program run
    - It still monitors the program
    - It keeps monitoring the program all the time it's in memory

Managing Memory

- The OS controls everything
- When your program is running, you can ask the OS for more memory
- It will give you some if there is extra memory available
  - You should think of memory as huge
  - Requesting memory is common
  - The OS lends out memory often

Back to C++

- Look at this code fragment

```
int* int_ptr;
int_ptr = new int;
*int_ptr = 100;
cout << *int_ptr;
```

- It creates a pointer (using new)
  - The pointer is a pointer to an int
- It asks the OS for some memory
  - How much does it ask for?
  - How much does it get?
- It stores 100 in an int in that memory
- It prints that value to verify that things worked

A New Idea

- Notice that this is an entirely new situation
- There is no variable here that the pointer can get the address of and then point to
- Instead, the pointer points off into memory
- This is just a more-or-less random piece of memory from the program's view
  - The program thinks it was just some area of memory big enough to hold an int
- Of course, from the OS's view, it was the next available area in memory

Dynamic Allocation

- We say that the memory given to the program was dynamically allocated
  - It was obtained while the program was running
- This memory comes from a place called the heap
  - That's where the OS gets its spare memory from
- The OS manages the heap

Pointers Again

- A pointer can point to anything, even a struct

```
struct PersonInfo
{
        string name;
        string street;
        string city;
        string state;
        string zip_code;
};
PersonInfo* person_ptr;
```

Using the Pointer

- To use the pointer, you first have to get some memory
  - You do this with new

    person_ptr = new PersonInfo;
- Then you store some information into that memory

  person_ptr -> name = "Cay";
    - Remember, the name member of the struct is of type string

A Dynamic Array

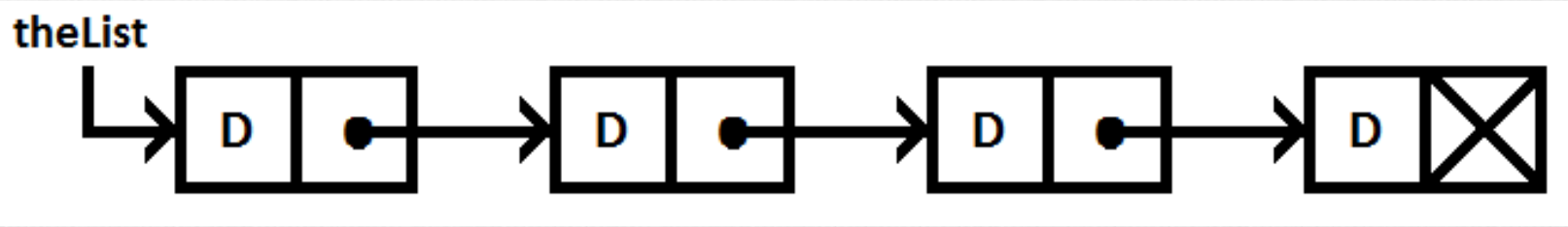
- Why would you want to do that?
- You have now created essentially, a dynamic array
  - If course, your array now has only one element
- Old facts:
  - An array cannot change its size
  - If you wish to delete a location from an array, you can't do that
    - You can fake it, though
  - If you wish to insert a new location into an array, you can't do that either
    - You can also fake this too

A New World

- You now can create an "array" (of sorts) that does allow deletions and insertions
- We call this idea a linked list
  - A linked list is a kind of data structure

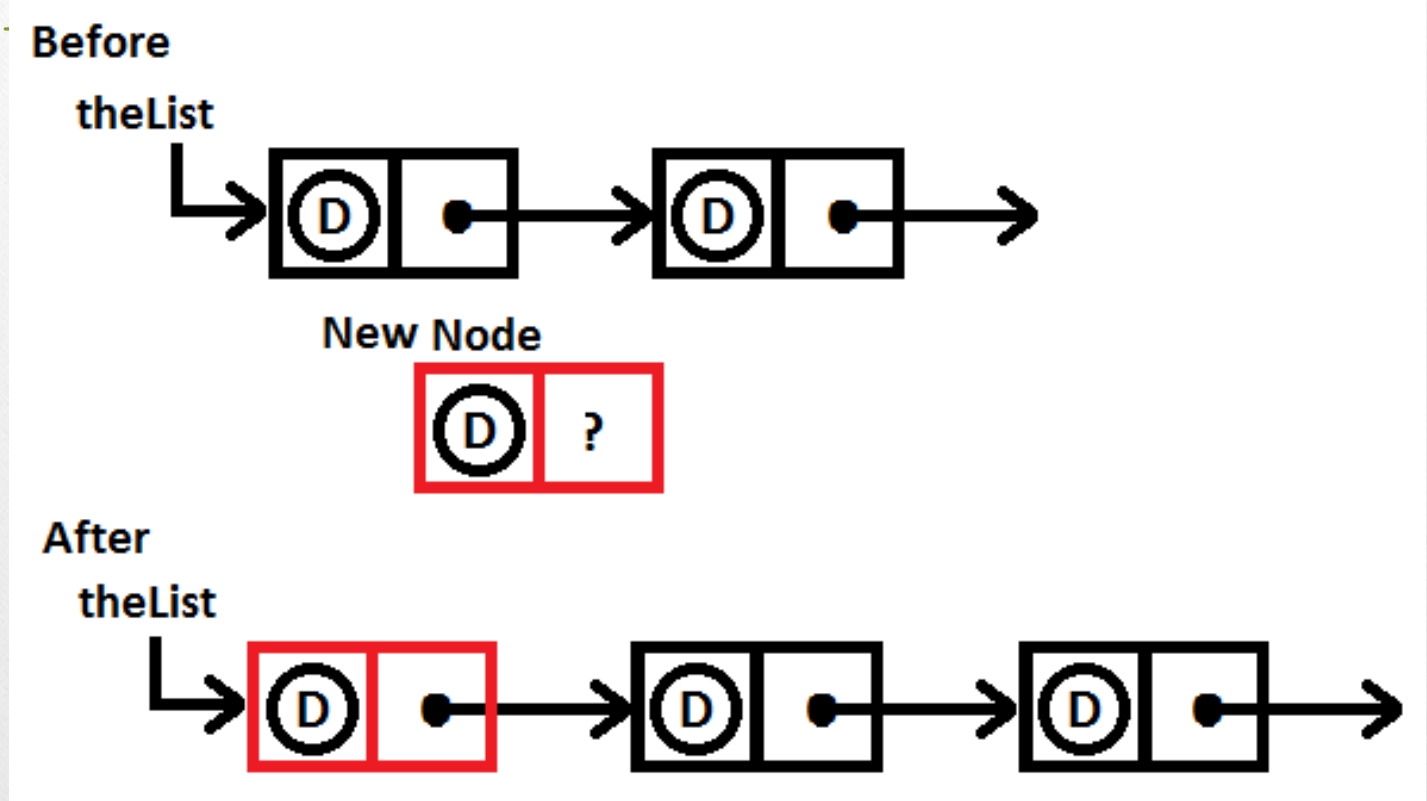
Pursuing the Idea of a Dynamic Array

- This is called a linked list
- Each of the four items in the list is called a node

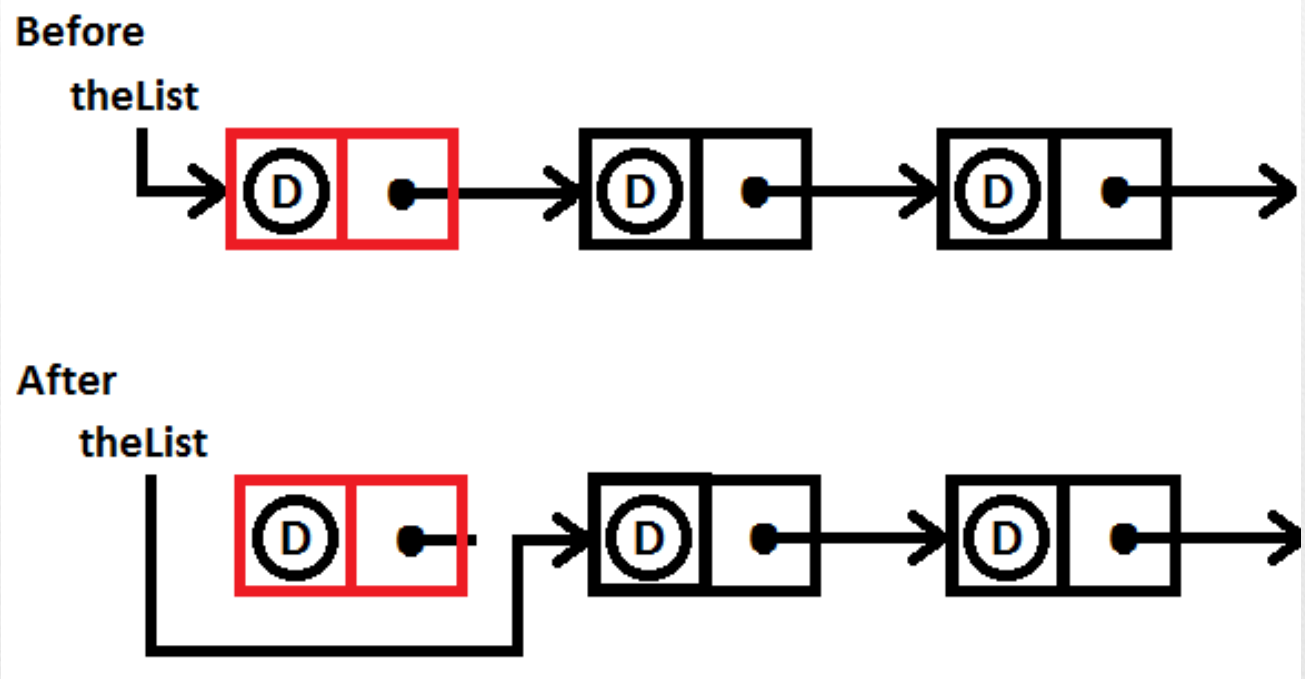


- We need a pointer to the start of the list
- Each node, then, points to the next node in the list

Adding a Node at the Beginning of the List



Deleting a Node from the Beginning of the List



Questions?

- Are there any questions?