Vectors

- Vector Introduction
  - Recall: arrays are fixed size
  - Vectors: "arrays that grow and shrink"
    - During program execution
  - Formed from Standard Template Library (STL)
    - Using template class

Vector Basics

- Similar to array:
  - Has base type
  - Stores collection of base type values
- Declared differently:
  - Syntax: vector<Base_Type>
    - Indicates template class
    - Any type can be "plugged in" to Base_Type
    - Produces "new" class for vectors with that type
  - Example declaration:
    vector<int> v;

Vector Use

- vector<int> v;
  - "v is vector of type int"
  - Calls class default constructor
    - Empty vector object created
- Indexed like arrays for access
- But to add elements:
  - Must call member function push_back( )
- Member function size( )
  - Returns current number of elements

Vector Example: Using a Vector (1 of 2)

Display 7.7 Using a Vector

```
1   #include <iostream>
2   #include <vector>
3   using namespace std;

4   int main( )
5   {
6       vector<int> v;
7       cout << "Enter a list of positive numbers.\n"
8            << "Place a negative number at the end.\n";

9       int next;
10      cin >> next;
11      while (next > 0)
12      {
13          v.push_back(next);
14          cout << next << " added. ";
15          cout << "v.size( ) = " << v.size( ) << endl;
16          cin >> next;
17      }
```

Vector Example: Using a Vector (2 of 2)

```
18        cout << "You entered:\n";
19        for (unsigned int i = 0; i < v.size( ); i++)
20            cout << v[i] << " ";
21        cout << endl;

22        return 0;
23    }
```

SAMPLE DIALOGUE

Enter a list of positive numbers.
Place a negative number at the end.
2 4 6 8 –1
2 added. v.size = 1
4 added. v.size = 2
6 added. v.size = 3
8 added. v.size = 4
You entered:
2 4 6 8

Vector Efficiency

- Member function capacity( )
  - Returns memory currently allocated
  - Not same as size( )
  - Capacity typically > size
    - Automatically increased as needed
- If efficiency critical:
  - Can set behaviors manually
    - v.reserve(32); // sets capacity to 32
    - v.reserve(v.size()+10); // sets capacity to 10 more than size