Parasol
Smarter computing.
Texas A&M University

Chapter 4
Computation

Bjarne Stroustrup

www.stroustrup.com/Programming

Vector

Parasol
Smarter computing.
Texas A&M University

- Think of it as an array of any type you set
  - Let's look at vector_ex.cpp
  - Methods that might be of use
    - push_back – insert new element at the end of the array
    - front – provides first element of array
    - back – provides last element of array
    - erase – remove one or a range of elements
    - at

https://www.tutorialspoint.com/cpp_standard_library/vector.htm

Data for Iteration - Vector

Parasol
Smarter computing.
Texas A&M University

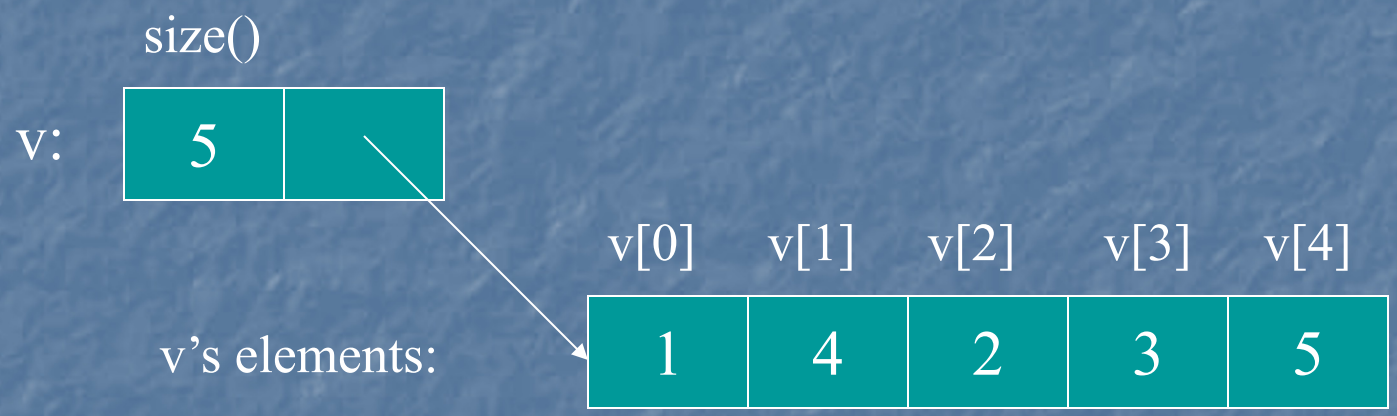
- To do just about anything of interest, we need a collection of data to work on. We can store this data in a **vector**. For example:

```
// read some temperatures into a vector:
int main()
{
    vector<double> temps;       // declare a vector of type double to store
                                // temperatures – like 62.4
    double temp;                // a variable for a single temperature value
    while (cin>>temp)           // cin reads a value and stores it in temp
        temps.push_back(temp);      // store the value of temp in the vector
    // ... do something ...
}
// cin>>temp  will return true until we reach the end of file or encounter
// something that isn't a double: like the word "end"
```

Vector

- Vector is the most useful standard library data type
  - a **vector<T>** holds a sequence of values of type **T**
  - Think of a vector this way

    A vector named **v** contains 5 elements: {1, 4, 2, 3, 5}:



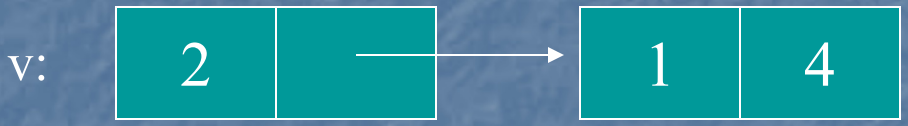
Vectors

Parasol
Smarter computing.
Texas A&M University

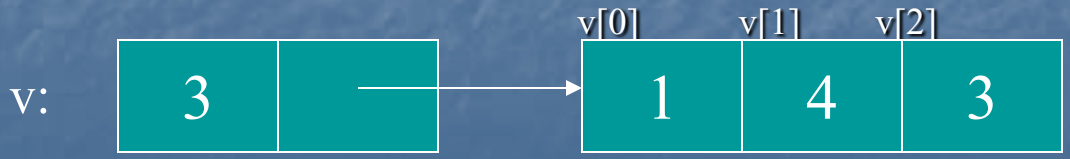
vector<int> v; // start off empty

v.push_back(1); // add an element with the value **1**

v.push_back(4); // add an element with the value **4** at end (“the back”)



v.push_back(3); // add an element with the value **3** at end (“the back”)



Vectors

- Once you get your data into a vector you can easily manipulate it

```
// compute mean (average) and median temperatures:
int main()
{
    vector<double> temps;       // temperatures in Fahrenheit, e.g. 64.6
    double temp;
    while (cin>>temp)  temps.push_back(temp); // read and put into vector

    double sum = 0;
    for (int i = 0; i< temps.size(); ++i) sum += temps[i];  // sums temperatures

    cout << "Mean temperature: " << sum/temps.size() << '\n';
    sort(temps);          // from std_lib_facilities.h
                          // or sort(temps.begin(), temps.end());
    cout << "Median temperature: " << temps[temps.size()/2] << '\n';
}
```

Traversing a vector

Parasol
Smarter computing.
Texas A&M University

- Once you get your data into a vector you can easily manipulate it
- Initialize with a list
  - **vector<int> v = { 1, 2, 3, 5, 8, 13 };** *// initialize with a list*
- *often we want to look at each element of a vector in turn:*

```
for (int i = 0; i< v.size(); ++i) cout << v[i] << '\n';  // list all elements

// there is a simpler kind of loop for that (a range-for loop):

for (int i : v) cout << x << '\n';  // list all elements

// for each x in v ...
```

Example – Word List

Parasol
Smarter computing.
Texas A&M University

```
// "boilerplate" left out

    vector<string> words;
    for (string s; cin>>s && s != "quit"; )          // && means AND
            words.push_back(s);

    sort(words);                                     // sort the words we read

    for (string s : words)
            cout << s << '\n';

    /*
       read a bunch of strings into a vector of strings, sort
       them into lexicographical order (alphabetical order),
       and print the strings from the vector to see what we have.
    */
```

Word list – Eliminate Duplicates

Parasol
Smarter computing.
Texas A&M University

// Note that duplicate words were printed multiple times. For
// example "the the the". That's tedious, let's eliminate duplicates:

```
vector<string> words;
for (string s; cin>>s && s!= "quit"; )
        words.push_back(s);

sort(words);

for (int i=1; i<words.size(); ++i)
        if(words[i-1]==words[i])
                "get rid of words[i]"     // (pseudocode)
for (string s : words)
        cout << s << '\n';
```

how to do this?

wordsdup.cpp

// there are many ways to "get rid of words[i]"; many of them are messy
// (that's typical). Our job as programmers is to choose a simple clean
// solution – given constraints – time, run-time, memory.