# Implementing Binary Search

**Here's an implementation of binary search, named `bfind()`:**

```
int bfind(const Person data[], int first, int last,
          const string& key)
{
  cout << ".";                          // display comparisons
  if (last < first) {return -1;}        // not found
  int mid = (first + last) / 2;        // mid point

  if (key == data[mid].name.substr(0, key.size())) return mid;
  if (key < data[mid])
    return bfind(a, first, mid - 1, key); // look in left
  else
    return bfind(a, mid + 1, last, key);  // look in right
}
```

Add this code **before** your `find()` function, and then comment out the body of `find()`, and add a call to bfind() in its place, like this:

```
return bfind(contacts, 0, size - 1, key);
```

Now, when you run the program, you'll see that even the worst case will take only about 10 or 11 comparisons, instead of 500 or 1000.