Using Objects

Object-Oriented Programming with C++

Safe way to manipulate strings?

`std::string`

The string class

- You must add this at the head of you code

```
#include <string>
```

- Define variable of string like other types

```
string str;
```

- Initialize it with string contant

```
string str = "Hello";
```

- Read/write string with cin/cout

```
cin >> str;
```

```
cout << str;
```

Assignment

```
char cstr1[20];
char cstr2[20] = "jaguar";

string str1;
string str2 = "panther";

cstr1 = cstr2; // illegal
str1 = str2;   // legal
```

Concatenation

```
string str3;
str3 = str1 + str2;
str1 += str2;
str1 += "a string literal";
```

Constructors (Ctors)

```
string (const char *cp, int len);

string (const string& s2, int pos);

string (const string& s2, int pos, int len);
```

Sub-string

```
substr (int pos, int len);
```

Modification

```
assign (...);

insert (...);
insert (int pos, const string& s);

erase (...);

append (...);

replace (...);
replace (int pos, int len, const string& s);

...
```

Search

```
find (const string& s);
```

File I/O

```
#include <ifstream>  // read from file
#include <ofstream>  // write to file

ofstream File1("C:\\test.txt");
File1 << "Hello world" << std::endl;

ifstream File2("C:\\test.txt");
std::string str;
File2 >> str;
```

A Quick Tour of C++

Make them sorted!

```
int main()
{
  int arr[] = {64, 25, 12, 22, 11};
  int n = sizeof(arr)/sizeof(arr[0]);

  selection_sort(arr, n);
  return 0;
}
```

- how to write a *practical* sorting algorithm?
  - overloading, template, comparator...
  - native type, user-defined type, inheritance...