I/O to/from char arrays and strings

- sometimes we have text stored in a character array, but we want to convert it to a different type: e.g. from “37.5” to 37.5
- sometimes we have the reverse problem, we want to take a value of some type and store it as a string, e.g. 37.5 to “37.5”
- sometimes we have a long string and we want to read different parts of it in stages, e.g. take an address string like “900 Fifth St” and read “900”, “Fifth”, and “St” into three different variables
- cstdio allows us to do this with sprintf and sscanf (string print and string scan)
- sstream allows us to do this with in and cout

atoi, atof, stoi, stof

- in <cstdlib> we find functions atoi (ascii to int) and atof (ascii to float) to convert from char arrays to numbers

```
int i = atoi(“37”); // i now 37
float f = atof(“1.234”); // f now 1.234
```

  - (also works if the values are in char arrays)

- in <string> we find functions stoi (string to int) and stof (string to float) to convert from strings to numbers

```
string s = “37”;
int i = stoi(s);
s = “1.234”;
float f = stof(s);
```

sprintf, sscanf (in cstdio)

- sprintf prints into a char array instead of normal output

```
char text[SIZE];
int x = 3;
sprintf(text, “x is %d”, x);
// puts “x is 3” and null term into array
```

- snprintf also lets you specify max number of chars

```
snprintf(text, SIZE, “x is %d”, x);
```

- sscanf reads from a char array or text string

```
sscanf(“900 fifth”, “%d”, &x); // reads 900 into x
```

sstream library: reading from text

- istringstream can be used to read from char arrays/strings

```
string text = “900 Fifth St”;
istringstream strm(text); // create stream variable from text
int x;
strm >> x; // reads the 900 into x
```

- can also use getline to read entire lines

```
istringstream strm(“900 Fifth St\nNanaimo”);
string text;
getline(strm, text, '\n'); // reads “900 Fifth St” into text
```

sstream library: writing into strings

- can use ostringstream to write text into a stream, then use it's .str() method to get the data as a string

```
int x = 3;
ostringstream strm;
strm << “X is “ << x << endl;
string s = strm.str(); // s now holds “x is 3\n”
```

Converting string to char[]

- we can initialize a string from a character array or text literal easily

```
char text[6] = “abcde”; // 6 since needs space for '\0'
string s = text;
```

- to get content of string as an array we need .c_str()

```
strcpy(text, s.c_str());
// gets content of s as a null-terminated char array,
// since that is what the strcpy is expecting
```