CS 211

Chapter 9

C-Strings:

- C-String is an array of type char that stores strings of characters that end with the null character, ‘\0’
- C-String is inherited from the C programming language

Declare a C-String:

Model:

char variable_name[size];

char variable_name[] = “initial_value”;

Ex:

char firstname[ ] = “Andy”;

char firstname[5] = “Andy”; // leave an extra bucker for the null char

char lastname[6];

lastname = “Abreu”; // illegal

lastname[0] = ‘A’;

lastname[1] = ‘b’;

lastname[2] = ‘r’;

lastname[3] = ‘e’;

lastname[4] = ‘u’;

lastname[5] = ‘\0’

Copy a C-String:

Assign:

strcpy(lastname, “Abreu”);

- http://www.cplusplus.com/reference/cstring/strcpy/

strncpy(lastname, “Abreu-Fenandez”, 5)

- http://www.cplusplus.com/reference/cstring/strncpy/

Compare a C-String:

- To test equality:

strcmp(str1, str2)

Returns an integral value indicating the relationship between the strings:

return value	indicates
< 0	the first character that does not match has a lower value in *ptr1* than in *ptr2*
0	the contents of both strings are equal
> 0	the first character that does not match has a greater value in *ptr1* than in *ptr2*

Note: The comparison on the c-strings are done on individual characters in the c-string from left to right based on **ASCII** value.

Concatenate C-Strings:

- To concatenate two c-strings together, we use function strcat or strncat:

```
char s1[11] = “Hello”;
char s2[] = “World”;
strcat(s1, s2);
cout << s1 << endl; // HelloWorld
```

Note: There must be enough space allocated to the array to accept both c-strings and null character

Strings:

- String is a class built into the C++ library
  - http://www.cplusplus.com/reference/string/string/

- String has predefined functions contained within the class which we can use for our convenience to do string manipulations

Declare a String:

Model:

string variable_name;

Ex:

string firstname = “Bob”;

String lastname = “Smith”;

- a string object has been created.

Concatenate Strings Together:

- To concatenate two or more strings together we can use the addition operator:

```
string s1 = “Hello”;
string s2 = “World”;
string s3 = s1 + “ “ + s2;
cout << s3 << endl; //Hello World
```

Comparison on Strings:

- To compare two strings we can use the comparison operators we are used to seeing when programming in C++.
- The comparison on the strings are done on individual characters in the string from left to right based on ASCII value.

```
string s1, s2;
st1 == st2          st1 != st2
st1 > st2           st1 >= st2
st1 < st2           st1 <= st2
```

Thus, we can sort strings in C++ the same way as we sort numbers.

Note: In ASCII code, ‘A’ is not the same as ‘a’. ‘A’ == 65 and ‘a’ == 97

- http://www.asciitable.com/

Read in a character:

- To read in one character from console, we can do the following:

```
char nextChar;
cout << “Enter your first name: “;
do{
        cin.get(nextChar);
}while(nextChar != ‘\n’);
```

Note: the get function will read in a single character, including the newline character or space

Read in a word:

To read in a string from the console we do the following:

```
string word;
cout << “Enter a word: “;
cin >> word;
```

Note: using cin, only reads in one word at a time, which means it reads up to the whitespace

Read in a line:

- To read in a whole line from the console, we can do the following:

```
string fullname;
cout << “Enter your first name: “;
getline(cin, fullname);

char fullname[20]
cout << “Enter your first name: “;
cin.getline(fullname, 19);
```

Note: getline function will read in all the characters entered until it hits the newline character

Parts of the string: (String Class)

We may access each character of the string using [ ] or function at();

So a string defined as following:

string name = “MICHAEL”;

Could be thought of as:

0	1	2	3	4	5	6
M	I	C	H	A	E	L

Where

name[0] = ‘M’; name[1] = ‘I’;

name.at(0) = ‘M’; name.at(1) = ‘I’;

Char Manipulation:

<cctype>

- toupper()
- tolower()
- isupper()
- islower()
- isalpha()
- isalnum()
- isspace()

Length of a string: (String Class)

To identify the length of the string, we can use one of the following methods:

```
 string st = “I love C++”;
 cout << “length: “ << st.length();
OR
 cout << “length: “ << st.size();
```

Insert into a String: (String Class)

- String library also allows us to insert some text into part of the string, instead of append the text to the end of a string. We use the insert function.

Model:

string_variable.insert(index_pos, text_tobe_insert);

index_pos: the starting position in the string_variable
where you want the text to go, and push all
the text in the string_variable back

text tobe_insert: the text you want to insert in to the
string variable

Insert into a String: (String Class)

Ex:

```
 string st = “NY”;
 st.insert( 1, “ew ” );

 //insert into the end
 str.insert( st.size(), “ork” );

 cout << st << endl;
```

Substring of a string: (String Class)

To get a substring from the original string:

Model 1:

```
string_variable.substr( starting_index );
```

Model 2:

```
string_variable.substr( starting_index, num_of_chars);
```

Substring of a string: (String Class)

Ex:

```
string st = “ABCDEFG”;
cout << st.substr( 0 ) << endl; //ABCDEFG
cout << st.substr( 1 ) << endl; //BCDEFG
cout << st.substr( 2 ) << endl; //CDEFG
cout << st << endl; //ABCDEFG
st = st.substr( 3 );
cout << st << endl; //DEFG
cout << st.substr( 0, 1 ) << endl; //D
cout << st.substr( 1, 2 ) << endl; //EF
cout << st.substr( 2, 1 ) << endl; //F
```

Main Function Parameters:

So far, main function looks like

```
 int main( ) ← without parameters
```

What if we need some information when we run the program?

```
 int main( int argc, char * argv[ ] ){

    cout << “number of arguments: ” << argc << endl;
    for ( int c = 0; c < argc; c++)
      cout << argv[ c ] << endl;
    return 0;
}
```