CSC 215

Math and Computer Science

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Drawbacks of C Strings

- Size of C string
- Comparisons
  - Don't support Boolean operators < <= > >= == !=

C++ Strings

- What are they?
  - They are an adt (Abstract Data Type) which contains data and functions that work on the data.
  - Make working with strings easier.
  - Provide an excellent set of functions to manipulate the data.
- They grow and shrink as needed.

Getting the Library

```
#include <string>
```

- Declaring a string:

```
string firstName;
string myName;
string fileName;
```

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

What I just did

- I now have a variable named firstName, myName and fileName.
- These variable names are actually called objects.
- They will contain data which is a collection of characters, but has no null terminator '\0'.
- It has a bunch of function that allow me to access and manipulate the data.
- Allows me to treat a string like a primitive data type(int, float,…).

Initializing

- Uses a special initializer list
  - Must be a C string, a c++ string, or a literal constant string.

Examples:

```
char cMyName[100] = "Roger";
string firstName1 = "Roger";
string firstName2 = cMyName;
string firstName3 = firstName1;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Input

- Read in a word or token

```
cin >> firstName;
fin >> firstName;
```

- Read in a line (contain spaces)

```
cin.getline will not work
```

Must use the stl function getline, no size needed.

```
getline( cin, myName );
getline( fin, myName );
getline( cin, myName, ':');
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Output

- Treat the string just like a variable
- Output to file and monitor is the same.

Examples:

```
cout << "My name is: " << firstName << endl;
cout << "My full name is: " << myName << endl;
fout << "My name is: " << firstName << endl;
Fout << "My full name is: " << myName << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Manipulating the Strings

- Use member functions
- The object gives us many functions that work on the data
- To access these function you use the '.' operator after the variable name.

Example:

```
string firstName;
firstName.memberFunctionName(…(arguments)…);
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

=, assign() Member Functions

- Both will resize the data set if needed. May use them to assign c++ strings, or c strings. Only the = can use a character.
- Assign function has 9 different ways of usage.

Examples:

```
str1 = str2;          str1.assign( str2 );
str1 = "Roger";       str1.assign( "Roger" );
str1 = cstr;          str1.assign( cstr );
str1 = 'r';
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Swap Member Function

- Will exchange the values within two c++ strings.

```
string str1 = "Roger", str2 = "Schrader";
str1.swap( str2 );
cout << str1 << endl;   // Schrader
cout << str2 << endl;   // Roger
```

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

+= Member Functions

- Appends Characters to the end of a string
  - Can append a C string, C++ String or single Character

```
string str1 = "Hello";
string str2 = "world";
str1 += ' ';
str1 += str2;

cout << str1 << endl;   // outputs Hello world
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Append and Push Back Member Functions

- append, concatenates characters to the end of a string.
  - Can append a C string, C++ String
  - 8 ways of calling the append
- push_back, concatenates a single character to the end of a string.

```
string str1 = "Hello";
string str2 = "world";
str1.push_back (' ');
str1.append( str2 );
cout << str1 << endl;   // outputs Hello world
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Insert Member Function

- Insert will position one or more characters into an existing string.
  - Works with c strings, c++ strings, and single characters.
  - 11 ways to call the insert function

```
string str1 = "FlintstoneFred";
char cstr2[100] = ", Barney";
str1.insert( 10, ", ");       // "Flintstone, Fred"
str1.insert( 10, 3, ' ');     // "Flintstone   Fred"
str1.insert( 10, cstr2, 2 ); // "Flintstone, Fred"
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Erase Member Function

- Erase, removes characters from a string object
  - 4 ways to call the erase function

```
string str1 = "FlintstoneFred";
str1.erase();       // str1 is now empty
str1.erase(5);      // str1 is now "Flint"
str1.erase(5,5);    // str1 is now "FlintFred"
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Clear Member Function

- Clear, Removes all characters from the string object.

```
string str1 = "FlintstoneFred";
str1.clear();          // str1 is now empty
```

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Resize Member Function

- Resize, changes the number of characters.
  - Shrinks or increases the capacity of the string.
  - 2 ways to call it.

```
string str1 = "FlintstoneFred"; // holds 14 chars
str1.resize(10);   // str1 can only hold 10 chars
          // "Flintstone" (10 chars)
str1.resize(20,'-');// increase to 20 padding
          // with '-''s "FlintstoneFred------"
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Replace Member Function

- Replace, substitues characters for other characters
  - Works with c strings, c++ strings and characters.
  - 14 ways to call the this function

```
string str1 = "FlintstoneFred";
string str2 = " – Barney";
str1.replace(5,5,1,' '); // "Flint Fred"    start at 5th char,
                         // replace the next 5 with 1 space

str1.replace(10,4,str2); // "Flintstone - Barney"  start at
                   // 10th char, replace the next 4 with str2
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

+, Member function

- +, joins two strings together.
  - Works with c string, c++ strings and characters.
  - One of the operands must be a c++ string object.

```
string str1 = "Fred";
char lastname[100] = "Flintstone";
string result;
result = "I like " + str1 + " " + lastname + "!";
cout << result << endl;
          // outputs "I like Fred Flintstone!";
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Boolean Member Functions

- ==, != , <, <=, >, >=
  - Does a comparison based on the ascii chart.
  - Works with C strings and C++ strings. One must be a c++ string.
  - Does a case sensitive compare like strcmp but returns a true or false.

```
string str1 = "quit";
string str2;
cin >> str2;
if( str1 == str2 )
    cout << "You entered quit" << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Compare Member Function

- Compare, sees what is different between 2 strings
  - Works with c strings and c++ strings.
  - Returns the same values as the strcmp function. Does ascii comparison.
    - 0 they are equal
    - < 0 comes before string2
    - > 0 comes after string2
  - 6 different ways to call the function.

Compare Examples

```
string str1 = "FlintstoneFred";
string str2 = "stone";
if( str1.compare( str2 ) == 0 ) // no output
    cout << "They are equal" << endl;
if( str1.compare( 5, 5, str2 ) == 0 ) // output
    cout << "They are equal" << endl;
// start a position 5 and compare the next 5
// characters to str2
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Size and Length Member Functions

- Size and length both return the number of characters.

```
string str1 = "FlintstoneFred";
string str2 = "stone";

cout << str1.size() << endl;          // 14
cout << str2.length() << endl;        // 5
```

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Empty Member Function

- Empty, returns a true false value based on if the string contains any data.

```
string str1 = "FlintstoneFred";
if( !str1.empty() )
    cout << "Str1 not empty." <<endl;
str1.erase();
if( str1.empty() )
    cout << "Str1 is empty." << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Max Size and Capacity Member Functions

- max_size, returns an integer representing how large the string can become.
- capacity, returns an integer representing the number of characters the string can hold be for it needs to resize.

```
string str1;
cout << "Capacity: " << str1.capacity();
cout << " Max Size: " << str1.max_size() << endl;
// Capacity: 15 Max Size: 4294967294
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

[] and at Member functions

- Both allow access to individual characters.
- Neither will increase the size of the string
- At function will do boundary checking. Safely exits program.

```
string str1 = "Fred flintstone";
str1[5] = 'F';                  cout << str1[3];
str1.at(5) = 'F';               cout << str1.at(3);
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Copy Member Function

- Copy, transfers the data to a C string. Works like strncpy and is not guaranteed to null terminate the c string.

```
string str1 = "Fred Flintstone";
char cstr1[100];
str1.copy(cstr1, 10);
cstr1[10] = '\0';                    ← make sure.
cout << cstr1 << endl;  // Fred Flint

str1.copy(cstr1, 10, 5);
cstr1[10] = '\0';                    ← make sure.
cout << cstr1 << endl;  // Flintstone
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

C_str Member Function

- Returns a pointer to a constant character arrays that is null terminated.
- Valid for that instance only.
- Can be used in C string functions where const char * is an argument.

```
strcpy( cstr1, str1.c_str() );
fin.open( str1.c_str() );
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

data Member Function

- Returns a pointer to a constant character arrays that is **NOT** null terminated. (no guarantee of '\0')
- Valid for that instance only.

```
str1.data();
```

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Substr Member function

- Substr, returns a substring that is a portion of the string

```
string str1 = "Fred Flintstone";
string str2;
str2 = str1.substr(5);      //str2 is Flintstone

str2 = str1.substr(6,4);    //str2 is lint
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String size type

- Basically an index into the string object.
- You must include the string library.
- Can declare a variable like

```
std::string::size_type idx;
string::size_type idx;
```

npos

- Special constant, used to indicate no match (4294967295, UL Max)

```
std::string::npos;
string::npos;
```

```
if( str1.findfunction( "sub" ) == string::npos)
        cout << "Sub string not found" << endl;

idx = str1.findfunction( "Sub" );
if( idx == string::npos )
    cout << "Sub string not found" << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Find Member Function

- Find locates a string within a string object moving forward
  - Works with C Strings and C++ strings.
  - 4 ways to call the find function.

```
string str1 = "Flintstone, Fred";
idx = str1.find("e");
if( idx != string::npos )          // 9
    cout << "position: " << idx << endl;

idx = str1.find("e", idx+1);       // 14
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Rfind Member Function

- Rfind locates a string within a string object moving backwards
  - Works with C Strings and C++ strings.
  - 4 ways to call the find function.

```
string str1 = "Flintstone, Fred";
idx = str1.rfind("e");
if( idx != string::npos )          // 14
    cout << "position: " << idx << endl;

idx = str1.rfind("e", idx-1);      // 9
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Find First of Member Function

- Locates the first occurrence of any character within the argument string passed in moving forwards in the string object.
  - Works with C Strings and C++ strings.
  - 4 ways to call the find function.

```
string str1 = "Flintstone, Fred";
idx = str1.find_first_of("aeiou");
if( idx != string::npos )                    // 2
    cout << "position: " << idx << endl;

idx = str1.find_first_of("aeiou", idx+1); // 7
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Find Last of Member Function

- Locates the first occurrence of any character within the argument string passed in moving backwards in the string object.
  - Works with C Strings and C++ strings.
  - 4 ways to call the find function.

```
string str1 = "Flintstone, Fred";
idx = str1.find_last_of("aeiou");
if( idx != string::npos )                    // 14
    cout << "position: " << idx << endl;

idx = str1.find_last_of("aeiou", idx-1);     // 9
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Other finds

- Find First not of
- Find last not of

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Passing Strings to Functions

- Pass by Value

  void func(string s);

- Pass by Reference

  void func(string &s);

- Return value

  string str;

  str = string func( );