CSC 215

Math and Computer Science

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Character Arrays & Strings

- Character Functions
- #include <cctype>
  - Includes functions for manipulating single characters
  - char ch1 = 'a', ch2, ch3;
  - ch2 = toupper (ch1);
  - ch3 = tolower (ch2);

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Character Array

- Just an array of characters
- Can initialize every element with an initializer list
  - char cArray[3] = { 'a', 'b', 'c'};
- Can output it character by character or perform operations on each character

```
for( i=0; i<3; i++)
    cout << cArray[i];                    // cArray[i] = toupper( cArray[i] );
```

- Nothing special about it yet!

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Strings (c-style String)

- #include <cstring> // note the c in cstring
- Allows us to use functions that have been written for the character array type.
- C string is an array of characters but the data ends with a special character, the null terminator. '\0'

String

- Is an array of characters that is null terminated '\0'
- The string functions require this character at the end of the data.
- Can initialize the arrays at initialization time.

```
char  cArray1[6] = { 'h', 'e', 'l', 'l', 'o', '\0' };          // clumsy
char  cArray2[6] = {'h', 'e', 'l', 'l', 'o' };                 // clumsy
        // the missing character is filled with a 0 which is the '\0' char

char cArray3[6] = "hello";          // much better, note 6 characters
                                    // needed to store 5 letter word
                                    // computer automatically puts in '\0'
                                    // for us.
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Initializer lists

```
char char_array[3] = "word ";     //too many characters
char char_array[4] = "word";      //still too many characters –
                                  //no place for \0
char char_array[5] = "word";      //just right
char char_array[10] = "word"; //fills first 5 characters
```

Word to the wise – need a string to hold 30 characters, create array to hold 60, space is cheap.

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Filling a string with a word

```
int i = 0;
char cArray[10];
i=0;
while( i<9 )
{
    cArray[i] = cin.get();
    if( isspace( cArray[i] ) )
    {
        cArray[i] = '\0';
        i = 10;
    }
    else
        i++;
}
```

But, let the istream work for you.

cin >> cArray;

>> will skip all Whitespace, copy data into the array, and stop when it hits whitespace

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

cin.getline

- cin.getline(char *str, int n, char delimiter);
  - default delimeter is '\n'
  - Reads up to n-1 characters, saves one character
  - Guarantees a null is appened onto str
  - Delimeter is removed from the input buffer and then discarded

```
cin.getline(strLine, 200, '\n');      //puts up to 199 characters into strLine
                                      //stopping at 199 or the \n character

Cin.getline( strLine, 100, ';');       //puts up to 99 characters into strLine
                                      //stopping at 99 or the ";" character
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

cin.get

- cin.get(char *str, int n, char delimiter);
  - default delimeter is '\n'
  - Reads up to n-1 characters, saves one character
  - Guarantees a null is appened onto str
  - Delimeter is left in the input buffer

```
cin.get(strLine, 200, '\n');  //puts up to 199 characters into strLine
                              //stopping at 199 or the \n character

Cin.get( strLine, 100, ';');  //puts up to 99 characters into strLine
                              //stopping at 99 or the “;" character
```

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

Reading in a whole line

```
char cLine[100];                    // make sure it is big enough

Example 1:      cin. getline( cLine, 100, '\n');
Example 2:      cin.getline( cLine,100);
        // reads in at most 99 characters, one character is reserved
        // for the '\0' to make it a C-string
```

Example 1: can change the delimiter to any character, default if not given is the newline character (see example 2);

Danger: it will read until it encounters what ever character is required. If you type 300 characters then hit enter, it will overstep the bounds of the array. CSC 150 and CSC250 will have limits of 80 chars unless otherwise specified.

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Outputting an array

```
int i=0;
char cArray[100];
:
while( cArray[i] != '\0' )
       cout << cArray[i++];
```

- Let the ostream do the work for you much easier

```
cout << cArray;          // it will output each character until '\0'
                         // if the first 100 spot have no '\0',
                         // overstep the array until if finds one
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions - strcpy

- String Copy (strcpy)

  char *strcpy(char *destination, char * source);

- Would like to do cArray1 = cArray2 but is illegal in c++

  strcpy(cArray1, cArray2); // function to copy contents over

```
int i = 0;
while( cArray2[i] != '\0' )
{       cArray1[i] = cArray2[i];
        i++;
}
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions - strncpy

- Copy n number of characters from one string to another
  - Quits when a '\0' is encountered or n characters have been copied

    char *strncpy(char *destination , char *source, n);
  - Not guaranteed to null terminate.

  Consider:

```
char cArray1[20] = ""; char cArray2[20] = "hello world";
strncpy( cArray1, cArray2, 5 );  // cArray1 now contains the string "hello"

char cArray1[20] = "friend"; char cArray2[20] = "hello world";
strncpy( cArray1, cArray2, 5 );  // cArray1 now contains the string "hellod"

char cArray1[20]; char cArray2[20] = "hello world";
strncpy( cArray1, cArray2, 5 );  // cArray1 could contain the string "hello……"
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions - strcat

- Append a copy of one string to the end of the first

char *strcat (char * destination, char *source);

```
char cArray1[20] = "hello";
char cArray2[20] = "world";
char cArray3[100];
strcpy( cArray3, cArray1 );   // cArray3 now contains the string "hello"
strcat( cArray3, " " );       // cArray3 now contains the string "hello "
strcat( carray3, carray2 );   // cArray3 now contains the string "hello world"

char first[100] = "Fred", last[100] = "Flintstone";
char fullname[100];
Strcpy( fullname, last );     // fullname now contains "Flintstone"
Strcat( fullname, ", " );     // fullname now contains "Flintstone, "
Strcat( fullname, first );    // fullname now contains "Flintstone, Fred"
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions - strncat

- Append n numbers of characters to the end of a string

  char *strncat(char *dest, char *src, int n);

```
char first[100] = "Fred";
char last[100] = "Flintstone";
char abbrev[100] = "";
strcpy( abbrev, last);         // abbrev – "Flintstone"
strcat( abbrev, ", ");         // abbrev – "Flintstone, "
strncat( abbrev, first, 1 );   // abbrev = "Flintstone, F"
strcat( abbrev, "." );         // abbrev = "Flintstone, F."
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions - strcmp

- Ordering strings
  - Can not do if (str1 < str2) – this is comparing memory addresses
  - Must use strcmp function

    int strcmp(char *str1, char *str2);
    - The integer returned tells you the order
    - if str1 is before str2, the integer returned will be less than 0
    - if str1 is after str2, the integer returned is greater than 0
    - if str1 is equal to str2, the integer returned will be 0

strcmp example (less than)

str1	h (104)	e (101)	l (108)	l (108)	o (110)	\0 (0)
str2	h (104)	e (101)	l (108)	p (112)	\0 (0)	
Subtraction	0	0	0	-4		

```
char str1[100] = "hello";
char str2[100] = "help";
int order;
order = strcmp(str1, str2);                          // a negative value is return
If( order < 0 )
        cout << str1 << " then " << str2 << endl;
else if( order > 0 )
        cout << str2 << " then " << str1 << endl;
else
        cout << str1 << " is the same as " << str2 << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

strcmp example (Greater than)

str1	h (104)	e (101)	l (108)	p (112)	\0 (0)	
str2	h (104)	e (101)	l (108)	l (108)	o (110)	\0 (0)
Subtraction	0	0	0	+4		

```
char str1[100] = "help";
char str2[100] = "hello";
int order;
order = strcmp(str1, str2);                    // a positive value is return
If( order < 0 )
        cout << str1 << " then " << str2 << endl;
else if( order > 0 )
        cout << str2 << " then " << str1 << endl;
else
        cout << str1 << " is the same as " << str2 << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

strcmp example (Equal)

str1	h (104)	e (101)	l (108)	l (108)	o (110)	\0 (0)
str2	h (104)	e (101)	l (108)	l (108)	o (110)	\0 (0)
Subtraction	0	0	0	0	0	0

```
char str1[100] = "hello";
char str2[100] = "hello";
int order;
order = strcmp(str1, str2);                          // a zero value is return
If( order < 0 )
        cout << str1 << " then " << str2 << endl;
else if( order > 0 )
        cout << str2 << " then " << str1 << endl;
else
        cout << str1 << " is the same as " << str2 << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

strcmp example Careful

str1	B (66)	e (101)	e (101)	n (110)	\0 (0)	
str2	a (97)	p (112)	e (101)	\0 (0)		
Subtraction	-31					

```
char str1[100] = "Been";
char str2[100] = "ape";
int order;
order = strcmp(str1, str2);                    // a negative value is return
If( order < 0 )                                // 'B' comes before 'a'
        cout << str1 << " then " << str2 << endl;
else if( order > 0 )
        cout << str2 << " then " << str1 << endl;
else
        cout << str1 << " is the same as " << str2 << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions - strncmp

- Compare on the first n number of characters
  - int strncmp(char *str1, char *str2, int n);
  - Return values are the same as strcmp function -, 0, +

```
char str1[100] = "hello";
char str2[100] = "helpme";
int order;
order = strncmp( str1, str2, 3);              // compare only the first 3 chars
if( order == 0 )
        cout << "The first 3 characters are the same" << endl;
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Length

- Returns the number of characters in a C string

  int strlen(char *str);

- The number returned is also the position of the null terminator.

```
char str1[100] = "Hello world";
char str2[100] = "Fred Flintstone";

cout << strlen( str1 ) << endl;     // outputs 11
cout << strlen( str2 ) << endl;     // outputs 15
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions – _stricmp & _strincmp

- Case insensitive compare – Microsoft VS only
  - int _stricmp(char *str1, char *str2);
  - int _strincmp(char *str1, char *str2, int n);

```
char str1[100] = "Been";
char str2[100] = "apple";
int order;
order = _stricmp( str1, str2 );            // a positive would be returned
Order = _strincmp( str1, str2, 3);         // a positive would be returned
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions- _strupr

- Convert entire string to upper case

  char *_strupr(char *str);

```
char str1[100] = "hello";
_strupr( str1 );
Cout << str1 << endl;          // HELLO is outputted
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions- _strlwr

- Convert entire string to lower case

  char*_strlwr(char *str);

```
char str1[100] = "HELLO";
_strlwr( str1 );
Cout << str1 << endl;          // hello is outputted
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

String Functions- _strrev

- Reverses the entire string within the array

  char *_strrev(char *str);

```
char str1[100] = "hello";
_strrev( str1 );
Cout << str1 << endl;          // olleh is outputted
```

SOUTH DAKOTA
SCHOOL OF MINES
& TECHNOLOGY

The Input Buffer

- When you type at the keyboard, you store the characters typed into a buffer, view it as an array of characters.
- Input operations take one or more characters from the front of the buffer and convert them appropriately
- Your program must account for the characters in the buffer

Input Buffer – example (Incorrect)

```
int number;
char strLine[100];

cin >> number;                          // enter 55 <return>
cin.getline( strLine, 80);              // Doesn't wait for input
cout << number << " " << strLine<< endl; // 55 is outputted
```

Why?

Buffer for number:	5	5	\n	
Buffer after number:	\n			

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Input Buffer – example (correct version 1)

```
int number;
char strLine[100];

cin >> number;                                  // enter 55 <return>
cin.ignore( );                                  // or cin.ignore( 256, '\n' );
cin.getline( strLine, 80);                      // waits for input
cout << number << " " << strLine<< endl;        // 55 is outputted plus next line
```

Why?

Buffer for number:	5	5	\n	
Buffer after number:				

← nothing in buffer, waits for you to type something.

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Input Buffer – example (correct version 2)

```
int number;
char strLine[100];

cin >> number >> ws;                    // enter 55 <return>
cin.getline( strLine, 80);              // waits for input
cout << number << " " << strLine<< endl; // 55 is outputted plus next line
```

Why?

Buffer for number:	5	5	\n	
Buffer after number:				

← nothing in buffer, waits for you to type something.

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

Ignore

- Ignore n number of characters or until a delimiter is encountered
  - Ex cin.ignore(int n, char ch);
  - Default is 1 char, delimeter is eof
  - Ignore 4 character or until eof whichever is first
  - Ignore 6 characters or until semicolon

```
cin.ignore();
cin.ignore( 4 );
cin.ignore( 6 , ':' );
```

Files work the same way:

```
fin.ignore();
fin.ignore( 4 );
fin.ignore( 6, ':' );
```

SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

WS

- Discards all whitespace from the stream, quits when it encounters a non whitespace character

```
cin >> number >> ws;
ws(cin);
```

- Files work the same way

```
fin >> number >> ws;
ws(fin);
```