Adamson University
900 San Marcelino St., Ermita Manila
TEL. NO.: (02) 524 - 2011
www.adamson.edu.ph

Lecture #3
STRING

I. OBJECTIVES

1. To learn how a string can be created and manipulated and how it is represented in memory in C++.
2. To be able appreciate string functions as a part of many application programs made in C++ language.

II. DISCUSSION

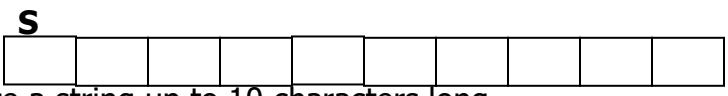
A **string** is a sequence of characters,
which is to be considered as a single data item.

The characters of a string are displayed enclosed within double quotes " ". Here are some examples of strings: "Jay-ar", "$2500", "rainbow", "123". These strings consist main of letters, digits, symbols and control characters.

In C++ there is no specific elemental variable type to store strings of characters and in order to fulfill this feature we can use arrays of type char, which are successions of char elements. The data type char is used to store single character, for that reason arrays of them are generally used to make strings of single characters.

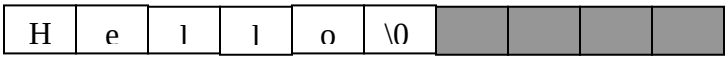
For example, the following array:

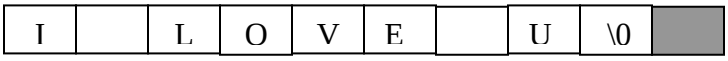
char4 S[10];



Can store a string up to 10 characters long.

This maximum size of 10 characters is not required to be always full used. Thus, we could represent **S** storing the strings "**Hello**" and "**I LOVE U**"





Notice that a null character '\0' is included as sentinel value to mark the end of the string. The box in gray indicates indeterminate values. If you read the characters in starting at indexed variable S[10], proceed to S[1], then S[2], and so forth. Since the symbol '\0' occupies one element of the array, the maximum length that the array can hold is one less that the size of the array.

Adamson University
900 San Marcelino St., Ermita Manila
TEL. NO.: (02) 524 - 2011
www.adamson.edu.ph

String variable declaration

A string variable is declared tpo be an array of characters in the format:
char_array_name[string_size];

Initialing a String Variable

If we want to initialize string of characters with predetermined values we can do it similar to any other array:

char my_string[ ]="Hello";
char my_string[ ]={'H','e,'l,'l,'o,'\0'};

In both cases the array or string of characters my_string is declared with a size of six characters: the five elements that compose Hello plus a null character '\0' which specifies the end of the string.

Assigning Values to Strings

Since, we can only assign only an element of an array and not the entire array, what would be valid to assign a string of characters to an array of char using a method like this:

array_name[array_index]=character_constant_value;
my_string[0]= 'H'; -stores character H to array my_string at index 0.
text[3]= ' '; -stores blank space to array text at6 index 3.
World[9]='\n' -stores a carriage return to array word at index 9.

In C++, you can store string values in variables and manipulate string values similar to how you manipulate data of other types. In C++ there is also the class string from the standard library, whose members are, declare in the <string>header file. Here are some functions in string.h:

Function	Description	Example
strcpy(target_string_var,source_string)	Copies the string value source_string into the string variable target_string_var	**strcpy(mystring,"hello");** The content of mystring will be "hello"
strcat(target_string_var,source_string)	Concatenates the string value source_string onto the end of the string in the string variable target_string_var.	**char string_var[10]= 'I love";** **strcat(string_var,"YOU");** The value of string_var will be "I Love You".
strlen(source_string)	Returns an integer value equal to the length of the source_string	**Char message[10]="hello';** **Int x;** **X=strlen(message);** The value of x will be 5.

Strcmp(string_1,string_2)	Returns 0 if string_1 and string_2 are the same. Returns a value less than 0 If string_1 is less than string_2. Returns a value greater than 0 if string_1 is greater than string_2.	**Strcmp("hello", "HELLO");** It will return a value greater than 0. **Strcmp("good"."good");** It will return a zero value which means FALSE.

Using getline( ) to Read Strings

Another preferred method in reading strings, which is the solution to whitespace dilemma is to use the member function of the iostream class called getline( ). The getline function is used in conjunction with *cin* rather than the >> operator. This function will allow cin to read the entire string, including whitespace.

cin.getline(string_var,array_size, delimiting character);

where the string_var is the name of the character array, array_size is the size of the array into which the string will be read and the delimiting character terminates the string entry and is not store as the part of the string.

Converting Numeral Strings

C++ has several standard functions for converting string to int, long ints, and doubles. The stdlib.h library provides three useful functions for this purpose:

- atoi-converts a string to an int type.
  syntax: atoi(string_var):
- atof-converts a string to a float type.
  syntax:atof(string_var);
- Atoll-convert a string to a long int type.
  syntax:atoll(string_var);