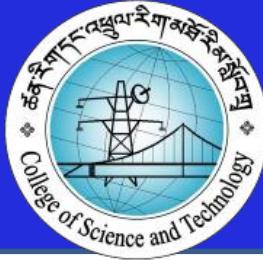




Royal University of Bhutan



Unit V – Part 03

(Character Arrays and String)

Lecture Slide

AS2023





Objectives

By the end of this session, students will be able to:

- Discuss how string variable are declared and initialized
- Explain how strings are read from terminal
- Describe how strings are written to screen
- Illustrate how strings are manipulated
- Explain how to pass string to a function



Royal University of Bhutan



String

- String is a sequencee of characters that is treated as a single data item
- Common operations on Strings:
 - Reading and writing strings
 - Combining string together
 - Copying one string to another
 - Comparing strings for equality
 - Extracting portion of string



Declaring and Initializing String variables



- C does not support string as datatype
- Allows character arrays to represent string
- General form

```
char string_name [size] ;
```

- Size determines the number of characters in the character array **string_name**
- Example:

```
char city [20] ;  
char name [30] ;
```
- Compiler will automatically supplies a null character ('\0') at the end of string
- Therefore, the *size* of the string should be equal to the *maximum number of characters* in the string *plus one*



Declaring and Initializing String variables



- Character array can be initialized in the following ways:

```
char city[9] = "New York";
```

```
char city[9] = { 'N', 'E', 'W', ' ', 'Y', 'o', 'r', 'k', '\0' };
```

- We must supply explicitly the null terminator when we initialize a character by listing its elements
- C also allows to initialize without specifying the size of an array

```
char string[] = { 'G', 'o', 'd', '\0' };
```

- We can declare the size larger than the string size in the initializer but not vice versa

```
char string[10] = { 'G', 'o', 'd', '\0' };
```



Reading Strings from Terminal



- *scanf* function can be used with *%s* format specification

```
char array_name[size];  
scanf ("%s", array_name);
```

- *scanf* function can read only a word and terminates with first *white space*

Example: given *New York* but stores only *New*



Reading Strings from Terminal



- *getchar* function can be used to read single character from the terminal

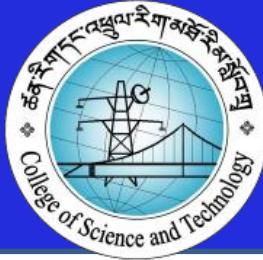
```
char ch;  
ch = getchar();
```

- Reading is terminated when new line character is entered
- *gets* function is a convenient method for reading a string containing white space

```
char line[80];  
gets (line);  
printf ("%s", line);
```



Reading Strings from Terminal



- *fgets* function is a convenient method for reading a string containing white space
- It terminates reading whenever it encounters a newline character like *gets* function

```
char line[80];  
fgets (line, sizeof(line), stdin);
```



Writing strings to screen

- *printf* is used extensively with %s to print string on the screen

```
printf ("%s", string);
```

- C supports *putchar()* to output the values of character variables

```
char ch;  
putchar(ch);
```

- *puts* is more convenient way of printing string values

```
char line[80];  
gets (line);  
puts(line);
```



Arithmetic Operations on Character



- C allows us to manipulate characters the same way we do with numbers
- Any character constant is converted to its equivalent integer value
- The integer value depends on the local character set of the systems
- Arithmetic operations can be performed
- It can be also used on relational expression



Putting String together

- Strings cannot be joined by arithmetic operations
- Concatenation is the process of combining two strings
- The target string variable should be large enough hold total characters

```
string 3 = string1 + string2  
string 2 = string1 + "Hello"
```



Comparison of two strings

- C does not allow comparison of two strings directly
`if (name1 == name2)`
`if (name2 == "ABC")`
- String have to be compared character by character
- It is done until mismatch or null character is found



String handling Functions

Function	Action
strcat ()	Concatenates two string
strcmp ()	Compares
strcpy ()	Copies one string over other
strlen ()	Finds the length of a string



Passing Strings to Functions



- Similar to those for passing arrays to functions
- Basic rules
 - The string to be passed must be declared as a formal argument of the function when it is defined
 - The function prototype must show that the argument is a string
 - A call to the function must have as string array name without subscripts as its actual argument



Thank you