LAB 9
STRINGS

Faculty of Electronic Engineering Technology Universiti Malaysia Perlis

1. OBJECTIVES:

1.1 To introduce the string data structure.
1.2 To be able to declare and initialize strings variables.
1.3 To be able to use string and string processing functions in programs.
1.4 To understand the two dimensional (2-D) arrays of type of characters which build up a list ofstrings.

2. INTRODUCTION:

Array is a collection or a data structure of a fixed number of components where in all of the components are of the **same type.**

The data type **string** is a programmer-defined and is not part of the **C** language. The **C** standard library supplies it.

Strings can be treated as array of type **char** used to store names of people, places, or anything that involves a combination of letters. Numbers can be stored as character, a string can be an array of numbers, too. To use the data type string, the program must include the header file string.

```
#include <string.h>
```

To declare a string, use the below command:

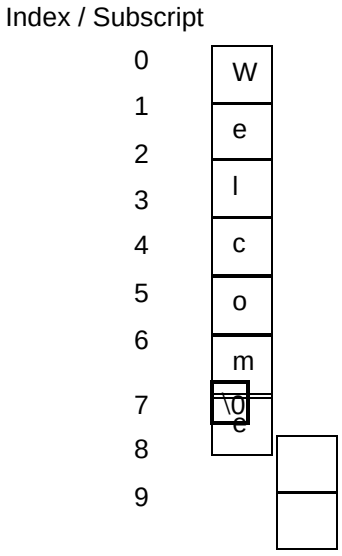
```
char string1[10];
```

The variable **string1** will hold strings from 0 to 9 characters long.

If we initialize the above variable with “Welcome” by using the below command

```
char string1[10] = {"Welcome"};
```

The illustration of the above array is given as:



Notice that **string1[7]** contains the character ‘**\0**’, the null character marks the end of a string.

2.1 Two Dimensional (2-D) Array of Character

```
char namelists[10][50];
```

The above data structure of array can store a list of names or strings; it can store 10 names or strings with the size of up to 50 characters.

3. TASKS:

3.1 The program shows how to declare, initialize and display strings. Type, compile and run the program. [Note : The function "**sizeof**" is to get the variable size in term of byte and the function "**strlen**" is to find the length of the string]

```
#include <stdio.h> #include
<string.h>

int main( )
{
        //3 ways of declaring and initializing strings char
        string1[10] = {"Welcome"};
        char string2[] = {'W','e','l','c','o','m','e','\0'}; char
        string3[] = "Good Bye";

        printf ("\nDisplay content of string1 : %s",string1);
        printf ("\nDisplay content of string2 : %s",string2);
        printf ("\nDisplay content of string3 : %s",string3);
        printf ("\n\nSize of string1 is %d",sizeof (string1));        //size of string1
        printf ("\nSize of string2 is %d",sizeof (string2));         //size of string2
        printf ("\nSize of string3 is %d",sizeof (string3));         //size of string3

        printf ("\n\nLength of string1 is %d",strlen(string1));      //length of string1
        printf ("\nLength of string2 is %d",strlen(string2));       //length of string2
        printf ("\nLength of string3 is %d\n",strlen(string3));     //length of string3
return 0;
}
```

a. Write the output of the program.

```
Display content of string1 : Welcome
Display content of string2 : Welcome
Display content of string3 : Good Bye
Size of string1 is 10
Size of string2 is 8
Size of string3 is 9
Length of string1 is 7
Length of string2 is 7
Length of string3 is 8
```

b. Comment on the differences in strings size and length in your output.

Size are reading the array size, where Length are reading the number of data.

c. The string library provides many functions for string processing. Find out what each of the string function listed below does:

i. strcpy - Copies the string pointed to, by *src* to *dest*.

ii. strcmp - Compare two strings

iii. strcat - Appends the string pointed to, by *src* to the end of the string pointed to by *dest*.

iv. atoi - This function returns the converted integral number as an int value. If no valid conversion could be performed, it returns zero.

v. atof - This function returns the converted floating point number as a double value. If no valid conversion could be performed, it returns zero (0.0).

vi. getchar - This function returns the character read as an unsigned char cast to an int or EOF on end of file or error.

vii. putchar - writes a single character to the standard output stream, stdout.

viii. gets - used to read a string or a text line.

ix. puts - used to write a line or string to the output(stdout) stream

3.2 The program shows how to assign values into strings variables. Type, compile and run the program.

```
#include <stdio.h>
#include <string.h>

#define STRING_LENGTH 20

int main( )
{
    char name[STRING_LENGTH];
    float marks[3]; float total = 0; int
    i;

    printf("\nEnter student name : ");
    scanf ("%s",name);              /*to store the input value to variable name*/
                                    /*note that there is no "&" sign at beginning of name, since
                                    name is an array and holds the address value.*/
    for (i=0;i<3;i++)
    { printf("Enter student test %d : ",i+1);
        scanf("%f",&marks[i]);
        total = total + marks[i];
    }

    /*Display information*/
    printf ("\nStudent name : %s", name); for (i=0;i<3;i++)
    printf ("\nTest %d mark : %5.2f",i+1, marks[i]);

    printf("\n\nTotal marks for %s : %5.2f\n",name,total);

    return 0;
}
```

a. Write the output of the program

```
Enter student name : Name
Enter student test 1 : 10
Enter student test 2 : 20
Enter student test 3 : 30

Student name : Name
Test 1 mark : 10.00
Test 2 mark : 20.00
Test 3 mark : 30.00

Total marks for Name : 60.00
```

b. Write a program that takes a list of students' names and marks and calculates the average marks. You are required to declare two arrays called **names** and **marks**. Assume number of students are 5.

Declare array **names**: (use 2-D array) char names[num_std][name_len]; //name length can be 20 characters long

Declare array **marks**: float marks[num_std]; Sample output:

```
Enter student name : Jason
Enter student marks : 60
Enter student name : Ahmad
Enter student marks : 77
Enter student name : Chong
Enter student marks : 88
Enter student name : Kumar
Enter student marks : 70
Enter student name : Daniel
Enter student marks : 55

Jason 60.00
Ahmad  77.00
Chong   88.00
Kumar   70.00
Daniel   55.00

Average marks for 5 students : 70.00
```

```c
#include <stdio.h>
#include <string.h>

int main ()
{
   int i;
   char name [5][20];
   float marks [5], avg, tot;

   for (i=0; i<5;i++)
   {
      printf("Enter student name : ");
      scanf ("%s",&name [i]);
      printf("Enter student marks :");
      scanf ("%f",&marks[i]);
   }

   for (i=0; i<5;i++)
   {
      printf("%s   %.2f\n",name[i],marks [i]);
   }

   for (i=0; i<5;i++)
   {
     tot += marks[i];
   }
   avg = tot/5;

   printf ("Average marks for 5 student: %.2f", tot);
}
```

3.3 Write a program that resembles a phone book, which stores and displays the names, the addresses (cities) and the telephone numbers for 10 people.

```
#include <stdio.h>
#include <string.h>

int main ()
{
   int i;
   char name [10][20], city [10][20];
   int  pNum [10][20], avg, tot;

   for (i=0; i<10;i++)
   {
      printf("Enter name : ");
      scanf ("%s",&name [i]);
      printf("Enter phone number :");
      scanf ("%d",&pNum[i]);
      printf("Enter city :");
      scanf ("%s",&city[i]);
   }

   for (i=0; i<10;i++)
   {
      printf("%s   %d   %s\n",name[i],pNum [i],city[i]);
   }

   return 0;

}
```