DATE:23.03.15

**ASSIGNMENT NO.17**

->PROBLEM STATEMENT:

Write a program in C to enter the name of students(maximum 10) and arrange them in lexicographical order using selection sort.

**->ALGORITHM:**

**Step 1:-** Declare character arrays,name[10][30],temp[30],small[30]

**Step 2:-** Input the total number of students,total

**Step 3:-** For(i=0 to total)

1. Input the name of the student,gets(name[i])
2. Increment i by 1

[End of For loop]

**Step 4:-** For(i=0 to total)

1. *strcpy*(small,name[i])
2. pos=i
3. For(j=i+1 to total)
4. If(*strcmp*(name[j],small)<0)Then
5. *strcpy*(small,name[j)
6. pos=j

[End of If ]

1. Increment j by 1

[End of For loop of Step 4.iii.]

1. *strcpy*(temp,name[i])
2. *strcpy*(name[i],name[pos])
3. *strcpy*(name[pos],temp)
4. Increment i by1

[End of For loop of Step 4]

**Step 5:-** For(i=0 to total)

1. *puts*(name[i])
2. Increment i by 1

[End of For loop]

**Step 6:-** End

->CODE:

/\*\*\*\*\*\*\*\*\*\*\*\*\*PROGRAM TO ARRANGE THE NAMES OF STUDENTS IN LEXICOGRAPHICAL ORDER USING SELECTION SORT\*\*\*\*\*\*\*\*\*\*\*\*/

#include<stdio.h>

#include<conio.h>

#include<String.h>

void main()//Declaration of main function

{

/\*Declaration of variables\*/

char name[10][30],temp[30],small[30];

int i,j,pos,total;

clrscr();

do

{

printf("\n Enter the number of students (Maximum 10 students) :");

scanf("%d",&total);//Input the number of students

}

while(total>10);

for(i=0;i<total;i++)

{

printf("\n Enter the name of student[%d] :",i+1);

fflush(stdin);

gets(name[i]);//Input the name of the students

}

for(i=0;i<total;i++)//Sort the names in lexicographical order using Selection sort

{

strcpy(small,name[i]);

pos=i;

for(j=i+1;j<total;j++)

{

if(strcmp(name[j],small)<0)

{

strcpy(small,name[j]);

pos=j;

}

}

strcpy(temp,name[i]);

strcpy(name[i],name[pos]);

strcpy(name[pos],temp);

}

printf("\n The name of the students arranged in lexographical order is as follows \n");

for(i=0;i<total;i++)

{

puts(name[i]);//Display the names

}

getch();

}

**->OUTPUT :**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Enter the number of students (Maximum 10 students) :12

Enter the number of students (Maximum 10 students) :5

Enter the name of student[1] :AHINDRA MONDAL

Enter the name of student[2] :RANAJIT MONDAL

Enter the name of student[3] :AKASH ROY

Enter the name of student[4] :GURPREET SINGH

Enter the name of student[5] :SOURAV GANGULY

The name of the students arranged in lexographical order is as follows

AHINDRA MONDAL

AKASH ROY

GURPREET SINGH

RANAJIT MONDAL

SOURAV GANGULY

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**->DISCUSSION:**

1. *String is a character array terminated with a null character (\0)*. The general form of declaration of a string variable is: ***char string\_name [size];***the **size** determines the no. of characters in the string name. When a compiler assigns a character string to a character array it automatically supplies a null character at the end of the string. So, the size should be equal to the maximum no of elements in the character array plus one.
2. The input function *scanf* can be used with %s format specification to read in a string of characters. The problem with this *scanf* function is that it terminates its input on the first white apace (blanks, tabs, new lines etc.) it finds. Also note that, unlike previous *scanf* calls, in the case of character arrays, the *ampersand (&) operator* is not required before the variable name, as the name of the string itself is a pointer to the first element.
3. We can use the function ***getchar()*** , repeatedly to read successive single characters from the input and place them into the character array by using a loop. This function has no parameters. The reading is terminated when a new line character (\n) is encountered and null character is placed at the end of the string. We also can use another standard library function ***gets()***,defined in the header file ***stdio.h***, to read an entire line of string from input and place it to the character array without using any loop. This function is very simple, with one parameter it is called under as: ***gets(str);*** where ***str*** is the string variable declared properly.
4. We can use the function ***putchar()*** , repeatedly to output a string of characters stored in a character array by using a loop. This function requires one parameter, the proper string variable which will be outputted. We also can use another standard library function ***puts()****,*defined in the header file ***stdio.h***, to print the value of the string stored in character array without using any loop. This function is very simple, with one parameter it is called under as: ***puts(str);*** where ***str*** is the string variable declared properly.
5. **About standard library string functions:**
   1. ***strlen():*** This function counts and returns the no of characters in a string. It takes the form:

***n = strlen(string);***

Where n is an integer variable which receives the value of the length of the given string.

* 1. ***strcpy():*** This function assigns the contents of string2 to string1. It takes the form:

***strcpy(string1, string2);***

The size of string1 should be larger than the size of string2.

* 1. ***strcmp():*** This function compares two strings identified by the arguments and has a value ‘0’ if they are equal. It takes the form:

***n = strcmp(string1, string2);***

String1 and string2 may be string variable or string constant.

* 1. ***strcat():*** This function joins two strings. It takes the form:

***strcat(string1, string2);***

When the function is executed, string2 is appended to string1, but string2 remains unchanged.

1. The main logic behind this program is that we have used two for loops to compare between the first String and with it all the remaining Strings and we have interchanged the Strings accordingly alphabetically using the function *strcmp()* incorporating the method of selection sort.