Int main()

{

Char a[51];

}

We know char[51] means max possible input we can give is 50 and we have taken 1 extra for the string terminator.

What if the user puts 51 or more than 51 characters?

Then UNDEFINED BEHAVIOUR.

#include<stdio.h>

int getStringLength(char \*p){

int x;

x=0;

while(\*p != '\0'){

x++;

p++;

}

return x;

}

int main(){

char name[51];

int l;

printf("Enter a string: ");

scanf("%s", name);

l = getStringLength(name);

printf("Length of (%s) is %d\n", name, l);

return 0;

}

While ( \*p != ‘\0’ )

Q. when control goes in loop?

1. Theoretical answer: jab P jis ko point kar raha h us ki value \o NAHI hogi
2. Jab p me jis ka address he us ki value \o nhi hogi.

When we have to copy one string from one variable to another, how?

~~A = B;~~

Here we are just assigning the address of the b to a.

Lets make function name copy\_string.

STRING CAN NOT BE COPIED USING ASSIGNMENT (=) OPERATOR.

Lets make copy function

#include<stdio.h>

void copyString(char \*p, char \*q){

while(\*q != '\0'){

\*p = \*q;

q++;

p++;

}

\*p = '\0';

}

int main(){

char a[51], b[51];

printf("Enter a string: ");

scanf ("%s", a);

copyString(b,a);

printf("String in array (a) is (%s)\n",a);

printf("String in array (b) is (%s)\n",b);

return 0;

}

/\*

String in array (a) is (great)

String in array (b) is (great)

\*/

Logic to traverse is fix

Void function (char \*p)

{

while(\*p != ‘\0’){

// code

p++;

}

}

Convert to Upper case

#include<stdio.h>

void toUpperCase(char \*p){

while(\*p != '\0'){

//if(\*p >= 90 && \*p <= 122){ // correct

//if(\*p >= 'a' && \*p <= 'z'){ // correct // ultimately ascii value will be placed

if(\*p >= 90 && \*p <= 'z'){ // this is also correct

\*p = \*p-32;

}

p++;

}

}

int main(){

char a[51];

printf("Enter a string to convert into upper case: ");

scanf ("%s", a);

printf("Input before conversion: %s\n",a);

toUpperCase(a);

printf("After conversion string is : (%s)\n",a);

return 0;

}

/\*

Enter a string to convert into upper case: Ujjain

Input before conversion: Ujjain

After conversion string is : (UJJAIN)

\*/

**Assignments**

1. Convert to Lower case
2. Capitalizse the string
3. Determine if the given string is palindrom
4. Compare case sensitive
5. Compare in case sensitive

Assignment 1:

#include<stdio.h>

void toLowerCase(char \*p){

while(\*p != '\0'){

if(\*p >= 65 && \*p <= 90 ){

\*p = \*p+32;

}

p++;

}

}

int main(){

char str[51];

printf("Enter a string: ");

scanf("%s", str);

printf("String Before conversion: %s\n", str);

toLowerCase(str);

printf("String After conversion: %s\n", str);

return 0;

}

Assignment 2

#include<stdio.h>

void capitalize(char \*p){

if(\*p != '\0' && \*p >= 97 && \*p <= 122 ) \*p = \*p-32;

p++;

while(\*p != '\0'){

if(\*p >= 65 && \*p <= 90 ){

\*p = \*p+32;

}

p++;

}

}

int main(){

char str[51];

printf("Enter a string: ");

scanf("%s", str);

printf("String Before conversion: %s\n", str);

capitalize(str);

printf("String After conversion: %s\n", str);

return 0;

}

Assignment 3:

#include<stdio.h>

int isPalindrop(char \*p){

int length = 0;

int y, flag, center;

char \*q;

q = p;

while(\*q != '\0'){

q++;

length++;

}

if(length %2 == 0){

center = length/2;

}else{

center = length/2+1;

}

flag = 0;

q--; // becase q is on \0 we have to compare 1 less than that.

for( y = 0; y<center ; y++){

if(\*p != \*q){

flag = 1;

}

p++;

q--;

}

return flag;

}

int main(){

char str[51];

printf("Enter a string: ");

scanf("%s", str);

if(isPalindrop(str) == 1){

printf("String %s is not Palindrom\n", str);

}else{

printf("String %s is Palindrom\n", str);

}

return 0;

}

Assignment 4:

// Case sensative comparision

#include<stdio.h>

int compareStrings(char \*p, char \*q){

int flag = 1;

while(\*p != 0){

printf("(%c) (%c)", \*p, \*q);

if(\*p == \*q){

flag = 0;

}else{

flag = 1;

}

if(\*q == '\0' || flag == 1) break;

p++;

q++;

}

return flag;

}

int main(){

char a[51],b[51];

printf("Enter a string: ");

scanf("%s", a);

printf("Enter another string: ");

scanf("%s", b);

if(compareStrings(a,b) == 1){

printf("(%s) and (%s) are not same\n",a,b);

}else{

printf("(%s) and (%s) are same\n",a,b);

}

return 0;

}

Assignment 5:

// inCase sensative comparision

#include<stdio.h>

int compareStrings(char \*p, char \*q){

int flag = 1;

while(\*p != 0){

if(\*p >= 90 && \*p <= 'z'){

\*p = \*p-32;

}

if(\*q >= 90 && \*q <= 'z'){

\*q = \*q-32;

}

if(\*p == \*q){

flag = 0;

}else{

flag = 1;

}

if(\*q == '\0' || flag == 1) break;

p++;

q++;

}

return flag;

}

int main(){

char a[51],b[51];

printf("Enter a string: ");

scanf("%s", a);

printf("Enter another string: ");

scanf("%s", b);

if(compareStrings(a,b) == 1){

printf("(%s) and (%s) are not same\n",a,b);

}else{

printf("(%s) and (%s) are same\n",a,b);

}

return 0;

}