**ASSIGNMENT NUMBER:**

**PROBLEM STATEMENT:**

Program in C to check wether a string is balanced or not from algebraic sense.

**THEORY:**

While doing calculations in Algebra, there are certain expressions which are enclosed with various types of brackets to denote which operations are to be done in which order. In this program the compiler will check wether the brackets are given in right order or not. The brackets are considered as a string in our program.

**For example**, the program should print “Balanced” for expression = “[()]{}{[()()]()}” and “string not Balanced” for exp = “[(])”.

**ALGORITHM:**

**Input:** The elements of an array, Say a, whose data type is charecter.

**Output:** The given string is balanced in algebraic sense or suitable unsuccessfull message.

**Data Structure Used:** Three arrays a[],b[] and c[], used for storing the string given as a input by the user.

**STEPS:**

Step 1: Repeat through step 2 to step 14 while(a[i]!=NULL)

Step 2: if(a[i]= ‘[’)

Then

Step 3: set c1=c1+1

[End of if condition]

Step 4: If(a[i]=‘]’)

Then

Step 5: set c1=c1-1

[End of If condition]

Step 6: If(a[i]=’{’)

Then

Step 7: set c2=c2+1

[End of If condition]

Step 8: If(a[i]=‘}’)

Then

Step 9: set c2=c2-1

[End of If condition]

Step 10: If(a[i]=’(’)

Then

Step 11: set c3=c3+1

[End of If condition]

Step 12: If(a[i]=’)’)

Then

Step 13: set c3=c3-1

[End of If condition]

Step 14: set i=i+1

[End of for loop]

Step 15: Repeat through step 15 to step 24 while(a[i]!=NULL)

Step 16: If**(** a[i]=’[’ OR a[i]=’{’ OR a[i]=’(’ **)**

Then

Step 17: set b[j]=a[i]

Step 18: set j=j+1

[End of If condition]

Step 19: If( a[i]=’]’ )

Then

Step 20: set c[k++]= ‘[’

[End of If Condition]

Step 21: If( a[i]=’}’ )

Then

Step 22: set c[k++]= ‘{’

[End of If Condition]

Step 23: If( a[i]=’)’ )

Then

Step 24: set c[k++]= ‘(’

[End of If Condition]

[End of for loop]

Step 25: set m=strcmp(b,strrev(c)) *// strcmp is a function that compares the //string provided to the function as //argument. Strrev is a function that //reverses the string given as a argument to this function.*

Step 26: If( c1=0 AND c2=0 AND c3=0 AND m=0)

Then

Step 27: Print” String is Valid from algebraic Sense”

Else

Step 28: Print” String is not balanced ”

[END]

**SOURCE CODE:**

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

int main()

{

char a[20],b[20],c[20];

int i,c1=0,c2=0,c3=0,k=0,j=0,m; //*Variable declaration*

printf("\nEnter the String:");

gets(a); //*taking the string to check its validity*

for(i=0;a[i]!='\0';i++)

{

if(a[i]=='[')

c1++;

if(a[i]==']')

c1--;

if(a[i]=='{')

c2++;

if(a[i]=='}')

c2--;

if(a[i]=='(')

c3++;

if(a[i]==')')

c3--;

}

for(i=0;a[i]!='\0';i++)

{

if(a[i]=='['||a[i]=='{'||a[i]=='(')

b[j++]=a[i];

if(a[i]==']')

c[k++]='[';

if(a[i]=='}')

c[k++]='{';

if(a[i]==')')

c[k++]='(';

}

c[k]='\0';

b[j]='\0';

m=strcmp(b,strrev(c)); // *Comparing the reversed string with the orignal one // and storing the return value in variable ‘m’.*

if(c1==0&&c2==0&&c3==0&&m==0)

printf("\nString is valid from alegebraic sense!!");

else

printf("\nString is not balanced!!");

return 0;

}

**INPUT AND OUTPUT:**

SET 1: Enter the String: [{65+(30-11)}]

String is valid from algebraic sense!!

SET 2: Enter the String: ({32-76}\*(5+{6-3)}/41[10\*(15-10]}

String is not balanced!!

SET 3: Enter the String: [{(45-30)\*(12+12)]

String is not balanced!!

**DISCUSSION:**

1) This program uses array to store the strings and then check its validity. For larger strings more space is required hence more memory consumption by the program.

2) The complexity of this program is high as it is upon the user to give the string as input. So if the user gives a large string and in our program we haven’t declared a large array, then it will be problematic. Although if we declare a large array, but the user gives a short string as input, then there will be problem of memory wastage.