DSP Assignment 2 Weekly

**Question - 1**

Write a Program that ask user to enter the string (Note : string can contain only ‘(’ , ’)’ , ’[’ , ’]’, ‘{’ , ‘}’ these mentioned 6 characters only )

You have to determine that input string is valid or not.

A input string is valid if and only if :

1. Open bracket must be enclosed by the same bracket.

2. Open Brackets Must be closed in correct order

**Source-Code:**

#include <stdio.h>

#include <stdlib.h>

#include<math.h>

#include<string.h>

#define MAX 30

// Stack Declaration

int top=-1;

int stack[MAX];

// Stack Overflow Condition and push

void push(char item){

//printf("%c pushed item", item);

if(item == MAX-1)

{

printf("\n Stack Overflow \n");

return;

}

else

{

top = top +1;

stack[top] = item;

}

}

// Stack Underflow Condition and pop

char pop(){

if(top == -1)

{

printf("\n Stack Underflow \n");

exit(1);

}

return(stack[top--]);

}

// Checking whether input given is valid or not.

int isBracket(char bracket)

{

if(bracket == '(' || bracket == ')' || bracket == '{' || bracket == '}' || bracket == '[' || bracket == ']')

return 1;

else

return 0;

}

int isMatchingPair(char bracket1, char bracket2)

{

if(bracket1 == '(' && bracket2 == ')')

return 1;

else if(bracket1 == '{' && bracket2 == '}')

return 1;

else if(bracket1 == '[' && bracket2 == ']')

return 1;

else

return 0;

}

// Checking whether it is a balanced one or not

int isExpression(char expression[])

{

int i;

char bracket2,bracket1;

for(int i = 0; i<strlen(expression); i++)

{

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

push(expression[i]);

if(expression[i]==')' || expression[i]=='}' || expression[i]==']')

{

if(top==-1) /\*stack empty\*/

{

printf("Not balanced\n"); //Right parentheses are more than left parentheses

return 0;

}

else

{

bracket2 = pop();

bracket1 = expression[i];

if(!isMatchingPair(bracket2,bracket1))

{

printf(" \n Mismatched parentheses are : ");

printf("%c and %c\n",bracket2,bracket1);

return 0;

}//end of if

}//end of else

if(top == -1 && i == strlen(expression)-1) /\*stack empty\*/

{

printf("\n Balanced Parentheses \n");

return 1;

}

}

}

return 0;

}

int main()

{

//code

char expression[MAX];

int valid,i;

//char bracket2,bracket1;

printf("Enter an algebraic expression : ");

gets(expression);

printf("\n Entered expression is: ");

puts(expression);

valid = isExpression(expression);

if(valid == 1)

printf(" True \n");

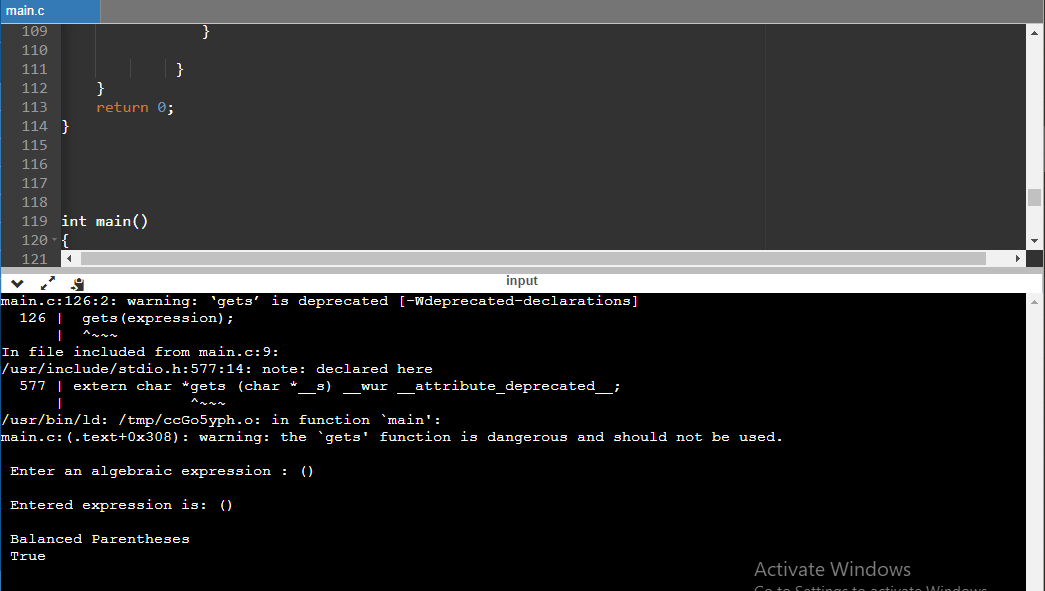
else

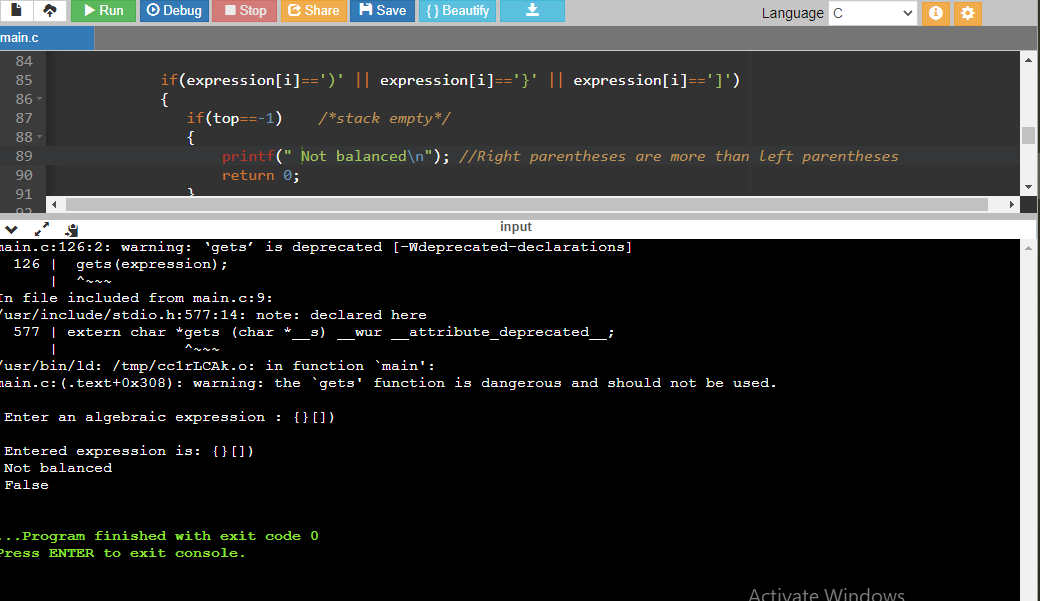
printf(" False \n");

return 0;

}

**Output**:





**Question - 2**

Create a singly link list which contains either 0 or 1 as element only

1. Provide an Insert function for inserting nodes with value either 0 or 1 only.(insert it at beginning or end)

2. Provide a function to calculate equivalent decimal value by taking singly link list elements as binary number.

**Source\_Code:**

#include <stdio.h>

#include<string.h>

#include<stdlib.h>

#define MAX 30

struct Node{

int val;

struct Node \*next;

};

struct Node \* head =NULL;

int decimalVal(struct Node \*head)

{

int dec = 0;

while (head != NULL)

{

// Left shift is equivalent to multiply

dec = (dec << 1) + head->val;

head = head->next;

}

return dec;

}

void insert(int ele){

struct Node \*temp, \*curr;

temp = (struct Node\*) malloc(sizeof(struct Node));

temp->val = ele;

temp->next =NULL;

if(head==NULL){

head = temp;

}

else{

curr = head;

while (curr->next != NULL)

{

curr = curr->next;

}

curr->next = temp;

}

}

void display()

{

struct Node \*temp = head;

while(temp!=NULL)

{

temp = temp->next;

}

}

int main()

{

char bin[MAX];

//int bin1[MAX];

int i=0,j=0,res,digit;

printf("\n Enter the binary number: ");

gets(bin);

while(i<strlen(bin))

{

digit =bin[i]-'0' ;

insert(digit);

i++;

}

display();

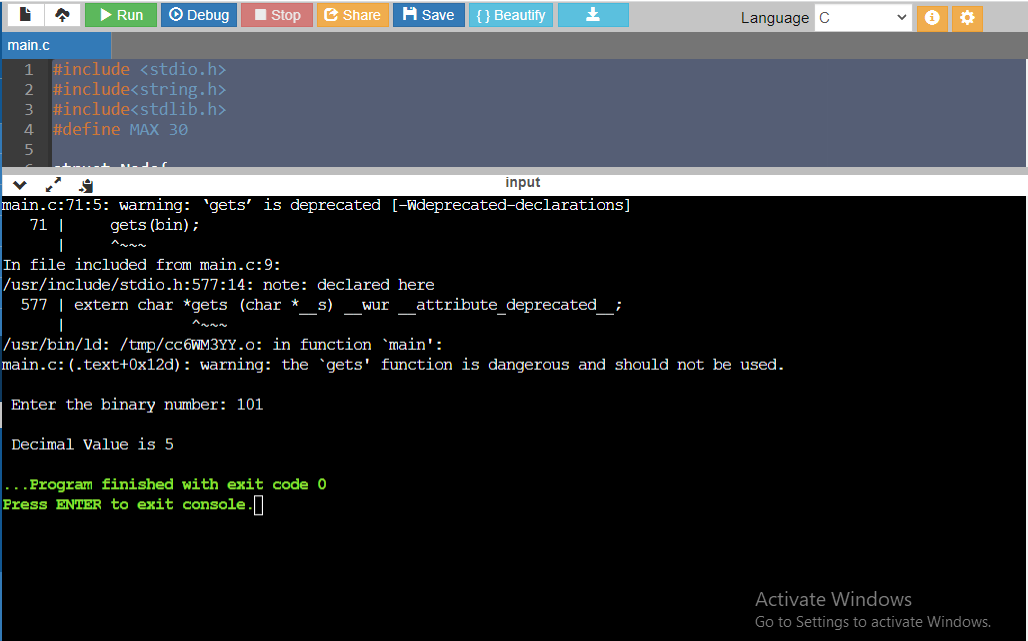
res = decimalVal(head);

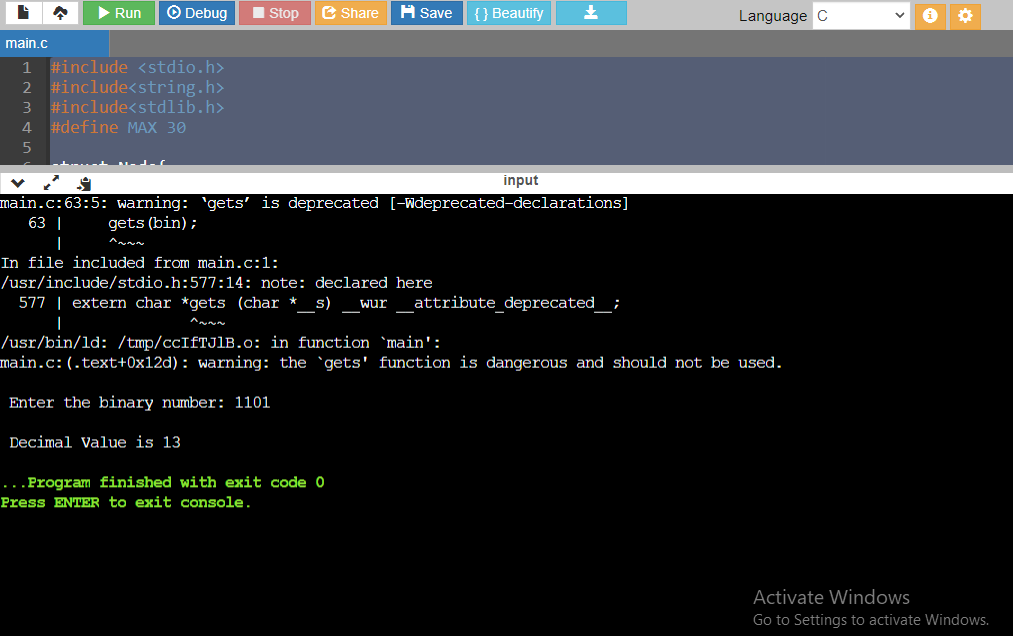
printf("\n Decimal Value is %d ",res);

return 0;

}

**Output**:





**Question 3:**

Write a source code to create a singly circular linked list. A node should consist of

• Character array to store three strings in “info” part

• Address of next node as “link” part

The following are the operations:

a) Create five nodes dynamically in a linked list with string values as given bellow (c) in

the “info” part

b) Third string value in the current node should match the first string value in the next

node

c) Display five nodes as follows

1. blue red green

2. green black yellow

3. yellow white orange

4. orange grey violet

5. violet brown blue

d) Display only the second-string values of a linked list

e) Replace first string values of the first and fifth node with the value “color” string and

display all five nodes and also display the output as “String not matching”

**Source\_Code:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<math.h>

struct Node {

char a[3][10]; //info part

struct Node \*next;

};

struct Node \*head = NULL;

void lastinsert()

{

struct Node \*newNode,\*temp;

newNode = (struct Node \*)malloc(sizeof(struct Node));

if(newNode == NULL)

{

printf("\nOVERFLOW\n");

}

else

{

if(head == NULL)

{

head = newNode;

newNode -> next = head;

for(int i=0;i<3;i++)

{

//printf("\nEnter a string %d: ",i+1);

scanf("%s",newNode -> a[i]);

}

} //end of if

else

{

temp = head;

while(temp -> next != head)

{

temp = temp -> next;

}

temp -> next = newNode;

newNode -> next = head;

for(int i=0;i<3;i++)

{

scanf(" %s",newNode->a[i]);

}

}//end of else

printf("\n Node inserted \n");

}

}

int isMatch(){

struct Node \*ptr;

ptr =head;

while(ptr->next != head)

{

if(strcmp(ptr->a[2],ptr->next->a[0])!=0)

{

return 0;

}

ptr = ptr->next;

}

return 1;

}

void display()

{

struct Node \*ptr;

ptr =head;

printf("\n The colors are \n");

while(ptr->next != head)

{

for(int i=0;i<3;i++)

{

printf(" %s\t", ptr->a[i]);

}

ptr = ptr->next;

printf("\n");

}

//printf("\n");

for(int i=0;i<3;i++)

{

printf(" %s\t ", ptr->a[i]);

}

printf("\n");

}

void display\_SecString()

{

struct Node \*ptr;

ptr =head;

while(ptr->next != head)

{

printf(" %s \n", ptr->a[1]);

ptr = ptr->next;

}

printf(" %s \n", ptr->a[1]);

}

void display\_Replace()

{

struct Node \*ptr;

ptr =head;

strcpy(ptr->a[0],"color");

while(ptr->next != head)

{

ptr = ptr->next;

}

strcpy(ptr->a[0],"color");

display();

}

int main()

{

int cnt;

printf("Enter number of Nodes ");

scanf("%d",&cnt);

printf("Node are %d \n",cnt);

printf("Enter elements to Insert \n");

for(int j=0; j<cnt ; j++)

{

lastinsert();

printf("Enter Again!\n");

}

display();

printf("\n The second string colors in the list are: \n");

display\_SecString();

if(isMatch())

{

printf("\n String Matching \n");

}

else

{

printf("\n String not Matching ");

}

display\_Replace();

if(isMatch())

{

printf("\n String Matching ");

}

else

{

printf("\n String not Matching ");

}

return 0;

}

**Output**:

