Aim:Write a C program to check whether a given string belongs to the language defined by a Context Free Grammar (CFG)

S → 0S0 | 1S1 | 0 | 1 | ε

Algorithm:

Certainly! Here is an algorithm for the provided code:

1. User Input: Prompt the user to enter a string.

2. Check Validity:

- Initialize variables `flag` and `flag1` to 1.

- Iterate through each character in the string. If any character is not '0' or '1', set `flag` to 0 (invalid).

3. Output Result: - If `flag` is not 1, print "String is Not Valid" and end the program.

- If `flag` is 1, proceed to the next step.

4. Check Palindrome: - Initialize variables `a` to 0 and `b` to `l - 1`, where `l` is the length of the string.

- While `a` is not equal to half of the string length:

- If `s[a]` is not equal to `s[b]`, set `flag1` to 0 (not a palindrome).

- Increment `a` and decrement `b`.

5. Final Output:

- If `flag1` is 1, print "The string is a palindrome" and "String is accepted."

- If `flag1` is 0, print "The string is not a palindrome" and "String is Not accepted."

6. \*\*End of Program.\*\*Top of Form

Program:

#include<stdio.h>

#include<string.h>

int main()

{

char s[100];

int i,flag,flag1,a,b;

int l;

printf("enter a string to check:");

scanf("%s",s);

l=strlen(s);

flag=1;

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

{

if(s[i]!='0' && s[i]!='1')

{

flag=0;

}

}

if(flag!=1)

printf("string is Not Valid\n");

if(flag==1)

{

flag1=1;

a=0;b=l-1;

while(a!=(l/2))

{

if(s[a]!=s[b])

{

flag1=0;

}

a=a+1;

b=b-1;

}

if (flag1==1)

{

printf("The string is a palindrome\n");

printf("string is accepted\n");

}

else

{

printf("The string is not a palindrome\n");

printf("string is Not accepted\n");

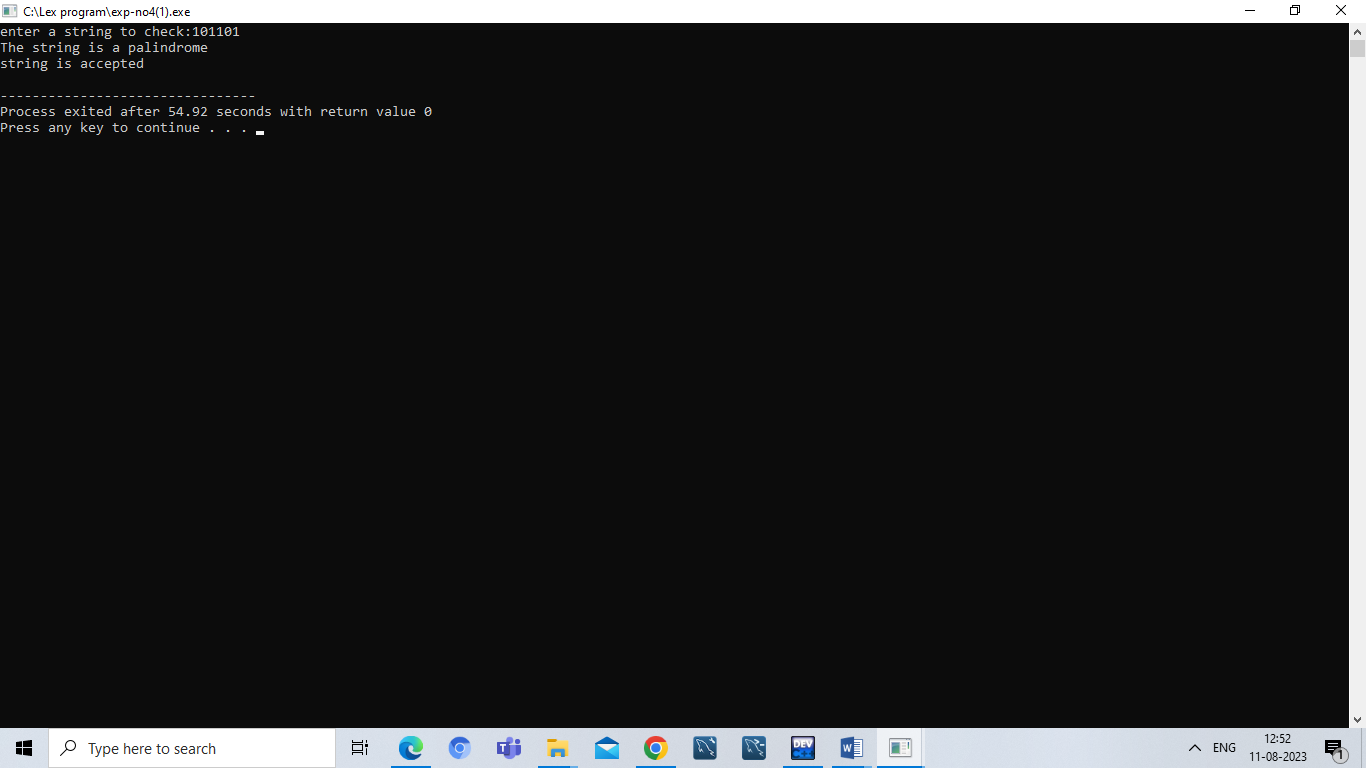
}

}

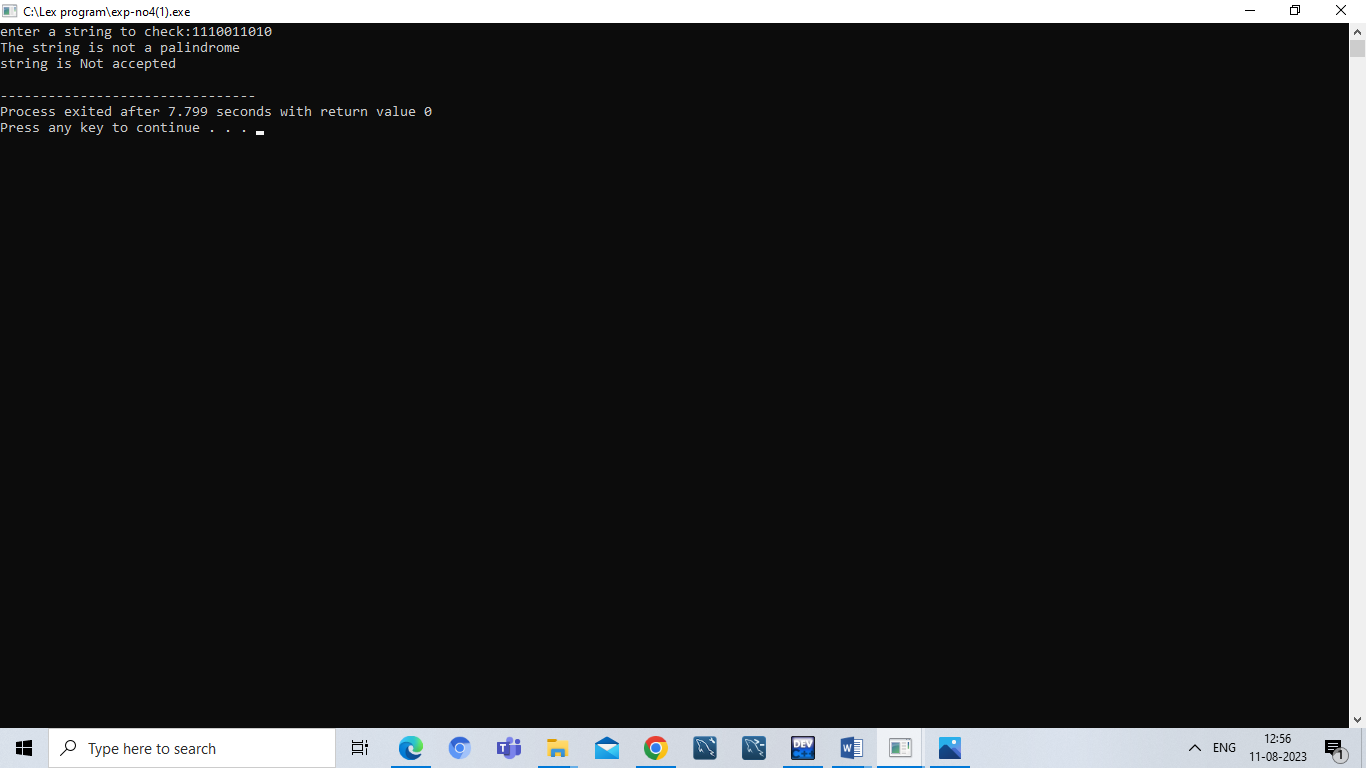
return 0;

}

Output 1:



Output 2:



Result:

Hence ,we successfully compiled the c program for CFG.