Q1. WRITE A LEX PROGRAM TO CHECK WHETHER THE GIVEN NUMBER IS PALINDROME OR NOT?

SOURCE CODE:

%{

#include <stdio.h>

#include <string.h>

%}

%%

[a-zA-Z]+ {

int i = 0;

int j = strlen(yytext) - 1;

int palindrome = 1;

while (i < j) {

if (yytext[i] != yytext[j]) {

palindrome = 0;

break;

}

i++;

j--;

}

if (palindrome) {

printf("%s is a palindrome.\n", yytext);

} else {

printf("%s is not a palindrome.\n", yytext);

}

}

%%

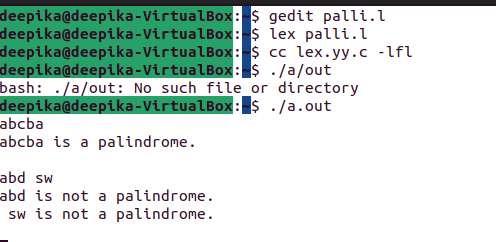
int main() {

yylex();

return 0;

}

OUTPUT:



Q2. WRITE A LEX PROGRAM TO IDENTIFY THAT THE GIVEN NUMBER IS POSITIVE NUMBER OR NEGATIVE NUMBER.

SOURCE CODE:

/\* Lex program to Identify and Count

Positive and Negative Numbers \*/

%{

int positive\_no = 0, negative\_no = 0;

%}

/\* Rules for identifying and counting

positive and negative numbers\*/

%%

^[-][0-9]+ {negative\_no++;

printf("negative number = %s\n",

yytext);} // negative number

[0-9]+ {positive\_no++;

printf("positive number = %s\n",

yytext);} // positive number

%%

/\*\*\* use code section \*\*\*/

int yywrap(){}

int main()

{

yylex();

printf ("number of positive numbers = %d,"

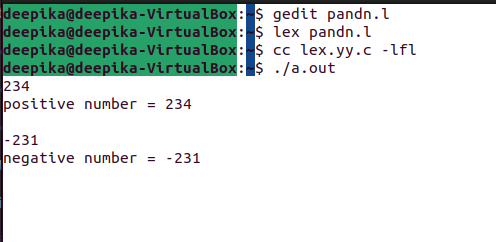
"number of negative numbers = %d\n",

positive\_no, negative\_no);

return 0;

}

OUTPUT:



.

Q3. WRITE A LEX PROGRAM TO CHECK WHETHER THE GIVEN YEAR IS LEAP YEAR OR NOT.

SOURCE CODE:

%{

void check(char \*);

%}

/\*Rule Section\*/

%%

[0-9] ;

[0-9][0-9] ;

[0-9][0-9][0-9] ;

[0-9][0-9][0-9][0-9] { printf("%s", yytext);check(yytext); }

[0-9][0-9][0-9][0-9][0-9]+ ;

%%

int main()

{

extern FILE \*yyin;

yyin=fopen("num", "r");

// The function that starts the analysis

yylex();

return 0;

}

void check(char \*a)

{

int x=0, i;

for(i=0;i<4;i++)

x=x\*10+(a[i]-'0');if(x%400==0)

printf("\tleap year\n");

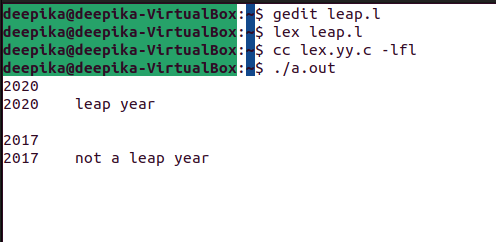
else if(x%4==0&&x%100!=0)

printf("\tleap year\n");

else

printf("\tnot a leap year\n");

}

OUTPUT:

Q4. WRITE A LEX PROGRAM TO CHECK WHETHER THE GIVEN NUMBER IS ARMSTRONG NUMBER OR NOT.

SOURCE CODE:

%{

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

%}

%%

[0-9]+ {int num=atoi(yytext);

int n=0,sum=0,temp=num;

while(temp>0)

{

n++;

temp=temp/10;

}

temp=num;

while(num>0)

{

int pow=1;

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

pow\*=num%10;

sum+=pow;

num=num/10;

}

if(temp==sum)

printf("Armstrong Number\n");

else

printf("Not an Armstrong Number\n");

} ;

\n ;

%%

int main()

{

printf("Enter a number:\n"); yylex();

return 0;

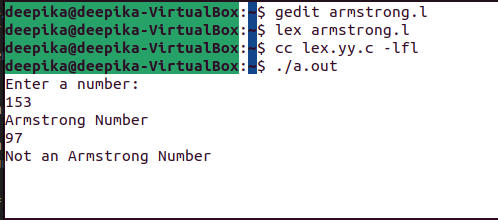
}

int yywrap()

{

return 1;}

OUTPUT:



.

Q5.WRITE A LEX PROGRAM THAT COUNTS THE NUMBER OF KEYWORDS.

SOURCE CODE:

%{

#include<stdio.h>

#include<string.h>

int i = 0;

%}

/\* Rules Section\*/

%%

([a-zA-Z0-9])\* {i++;} /\* Rule for counting

number of words\*/

"\n" {printf("%d\n", i); i = 0;}

%%

int yywrap(void){}

int main()

{

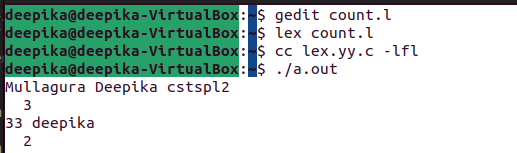
// The function that starts the analysis

yylex();

return 0;

}

OUTPUT:



Q6. WRITE A LEX PROGRAM THAT ACCEPTS A DFA ENDS WITH 11.

SOURCE CODE:

%{

#include <stdio.h>

%}

%option noyywrap

%%

0(0|1)\*11 { printf("Accepted\n"); }

0(0|1)\* { printf("Rejected\n"); }

. ;

%%

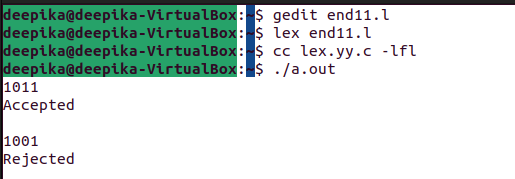
int main() {

yylex();

return 0;

}

OUTPUT:



Q7. WRITE A LEX PROGRAM THAT ACCEPTS A DFA WHICH CONTAINS EVEN NUMBER OF A.

SOURCE CODE:

%{

%}

%s A DEAD

%%

<INITIAL>a BEGIN A;

<INITIAL>b BEGIN INITIAL;

<INITIAL>[^ab\n] BEGIN DEAD;

<INITIAL>\n BEGIN INITIAL; {printf("Accepted\n");}

<A>a BEGIN INITIAL;

<A>b BEGIN A;

<A>[^ab\n] BEGIN DEAD;

<A>\n BEGIN INITIAL; {printf("Not Accepted\n");}

<DEAD>[^\n] BEGIN DEAD;

<DEAD>\n BEGIN INITIAL; {printf("Invalid\n");}

%%

int yywrap(){

return 1;

}

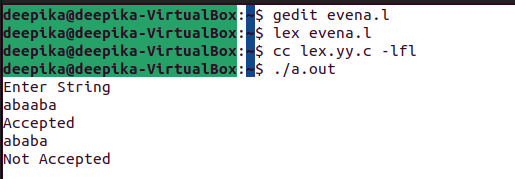
int main()

{

printf("Enter String\n");

yylex();

return 0;}

OUTPUT:

Q8. WRITE A LEX PROOGRAM THAT ACCEPTS A DFA WITH ABC.

SOURCE CODE:

%{

#include <stdio.h>

%}

%option noyywrap

%%

abc { printf("Accepted\n"); }

a(b|c) { printf("Rejected\n"); }

b(a|c) { printf("Rejected\n"); }

c(a|b) { printf("Rejected\n"); }

. ;

%%

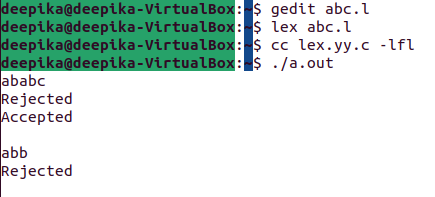
int main() {

yylex();

return 0;

}

OUTPUT:



Q9. WRITE A LEX PROGRAM TO COUNT THE LENGTH OF A STRING.

SOURCE CODE:

%{

#include <stdio.h>

int count = 0;

%}

%option noyywrap

%%

. { count++; }

\n { printf("Length of string: %d\n", count); }

%%

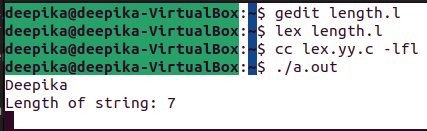
int main() {

yylex();

return 0;

}

OUTPUT:



Q10. WRITE A LEX PROGRAM TO COUNT NUMBER OF WORDS.

SOURCE CODE:

%{

#include<stdio.h>

#include<string.h>

int i = 0;

%}

/\* Rules Section\*/

%%

([a-zA-Z0-9])\* {i++;} /\* Rule for counting

number of words\*/

"\n" {printf("%d\n", i); i = 0;}

%%

int yywrap(void){}

int main()

{

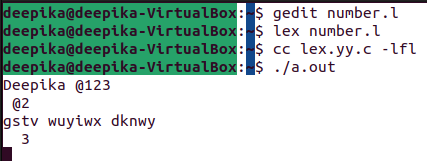
// The function that starts the analysis

yylex();

return 0;

}

OUTPUT:



Q11.WRITE A LEX PROGRAM FOR BINARY TO DECIMAL CONVERSION.

SOURCE CODE:

%{

#include <stdio.h>

int decimal = 0;

%}

%%

[01]+ {

for (int i = 0; i < strlen(yytext); i++) {

decimal = decimal \* 2 + (yytext[i] - '0');

}

printf("%d\n", decimal);

decimal = 0;

}

\n {

// Do nothing on newlines

}

. {

printf("Invalid input\n");

}

%%

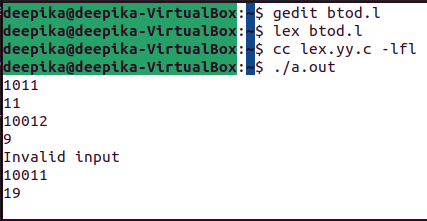
int main() {

yylex();

return 0;

}

OUTPUT:



Q12. WRITE A LEX PROGRAM THAT ACCEPTS ALL EVEN NO. OF A AND B.

SOURCE CODE:

%{

#include <stdio.h>

%}

%%

(a{2}b{2})\* {

printf("Accepted.\n");

}

a{2}b{0,1}|a{0,1}b{2} {

printf("Rejected.\n");

}

. ;

%%

int main() {

yylex();

return 0;

}

OUTPUT:

