LP LAB ASSIGNMENT-4

SUBMITTED BY: TEJSWO TIWARY
ROLL: 207278
SECTION:C

1. Write a lex specification to generate a scanner that will take a decimal between 1 to 999 in words and prints its numeric value as output.

CODE:

```
%{
        #include <stdio.h>
        #include <string.h>

        int num=0;

%}
%%

[Oo]ne          {num+=1;}
[Tt]wo          {num+=2;}
[Tt]hree        {num+=3;}
[Ff]our{num+=4;}
[Ff]ive {num+=5;}
[Ss]ix          {num+=6;}
[Ss]even        {num+=7;}
[Ee]ight        {num+=8;}
[Nn]ine         {num+=9;}
[Tt]en          {num+=10;}
[Ee]leven       {num+=11;}
[Tt]welve       {num+=12;}
[Tt]hirteen     {num+=13;}
[Ff]ourteen     {num+=14;}
[Ff]ifteen      {num+=15;}
[Ss]ixteen      {num+=16;}
[Ss]eventeen    {num+=17;}
[Ee]ighteen     {num+=18;}
[Nn]ineteen     {num+=19;}
[Tt]wenty       {num+=20;}
[Tt]hirty       {num+=30;}
[Ff]orty        {num+=40;}
[Ff]ifty{num+=50;}
[Ss]ixty        {num+=60;}
[Ss]eventy      {num+=70;}
[Ee]ighty       {num+=80;}
[Nn]inety       {num+=90;}
[Hh]undred      {num*=100;}
[\n]            {
                        printf("Number: %d\n\n", num);
                        num=0;
                }
```

```
.                   {;}
%%
int yywrap(){}
int main()
{
        printf("Enter Numbers:\n");
        yylex();
        return 0;
}
```

OUTPUT:

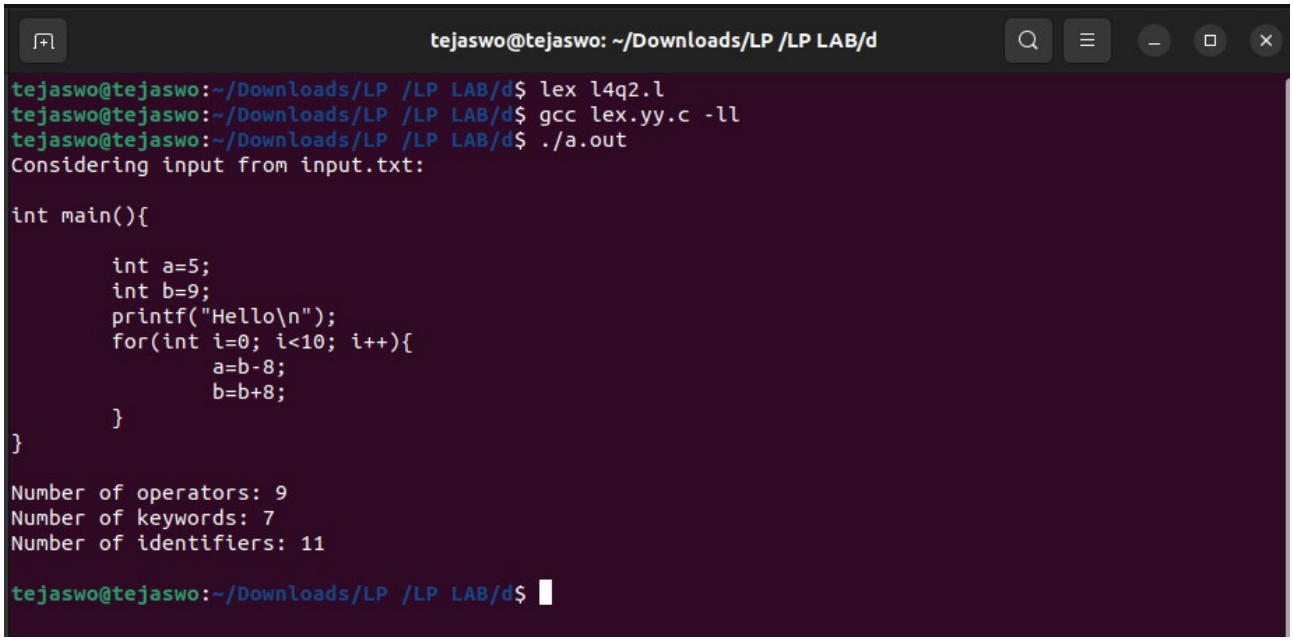
2. Write a lex program to specifying the scanner that recognises some of the keywords like begin, if, some of the operators, and an identifier, which is defined as any string that starts with letter and followed by letters or digits, and counts the number of identifiers, keywords, and operators encountered in the input given to the scanner.The action taken by the scanner is to increment a counter named key when it recognizes a keyword, increment a counter op, when it encounters an operator, and increment a counter id, when it encounters an identifier.

CODE:

```
%{
        #include <stdio.h>
        #include <string.h>

        int key=0, op=0, id=0;

%}

%%

"+"|"-"|"*"|"/"|"%"|"="|"=="|"<"|">"|">="|"<="|"&&"|"||"               {printf("%s",yytext);op++;}
```

```
"+="|"-="|"*="|"/="|"!="|"++"|"--"                    {printf("%s",yytext);op++;}

int|char|main|"#include"|if|else|for|while|auto|break|printf    {printf("%s",yytext);key++;}


[a-zA-z][a-zA-Z0-9]*          {printf("%s",yytext);id++;}

\"([^\\\"])*\"          {printf("%s",yytext);//Any single line string of form "text"
                        }

.                       {printf("%s",yytext);}

%%

int yywrap(){}
int main()
{
        printf("Considering input from input.txt:\n\n");
        extern FILE *yyin;
        yyin=fopen("input","r");
        yylex();
        printf("Number of operators: %d\n", op);
        printf("Number of keywords: %d\n", key);
        printf("Number of identifiers: %d\n\n", id);
        return 0;
}
```

OUTPUT:



3. YACC program which accept strings that starts and ends with 0

or 1.

CODE:

l3q3.l:

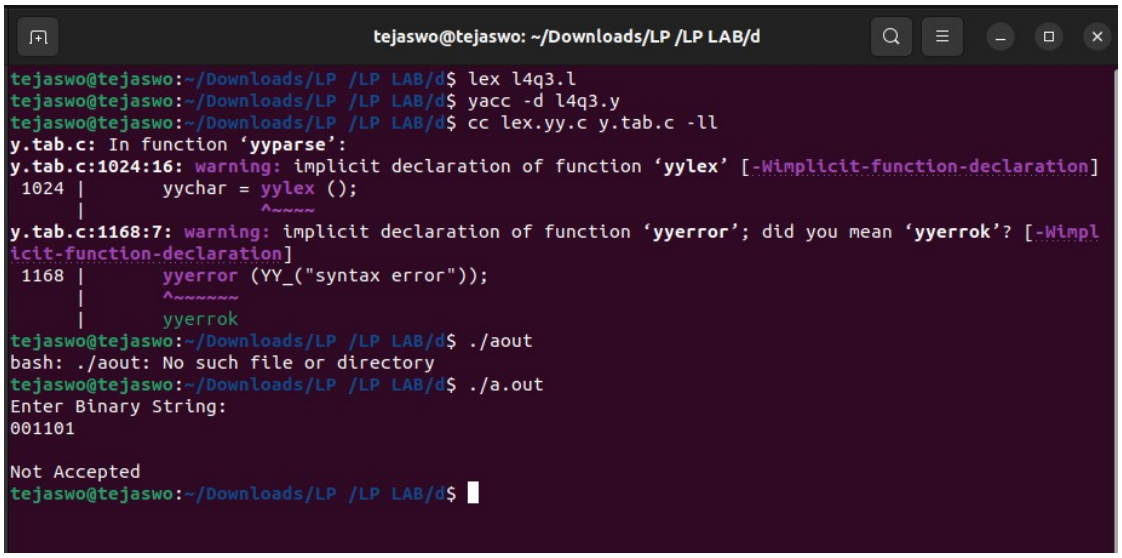
```
%{
	#include <stdlib.h>
	#include "y.tab.h"
	extern int yylval;
%}

%%

0 {yylval = 0; return ZERO;}

1 {yylval = 1; return ONE;}

.|\n {yylval = 2; return 0;}
```

l3q3.y:

```
%%
%{
	#include <stdio.h>
%}

%token ZERO ONE

%%
E :	T	{
			printf("Accepted\n");
			return 0;
			}
;

T :	S;

S :	A | ZERO Z | ONE O;

Z :	A Z | ZERO;

O :	A O | ONE;

A :	ZERO | ONE;


%%

int main() {
	printf("Enter Binary String:\n");
	yyparse();
}
```

```
int yyerror(char* s) {
        printf("\nNot Accepted\n");
}
```

OUTPUT:



4. YACC program to check whether given string is Palindrome or not.

CODE:

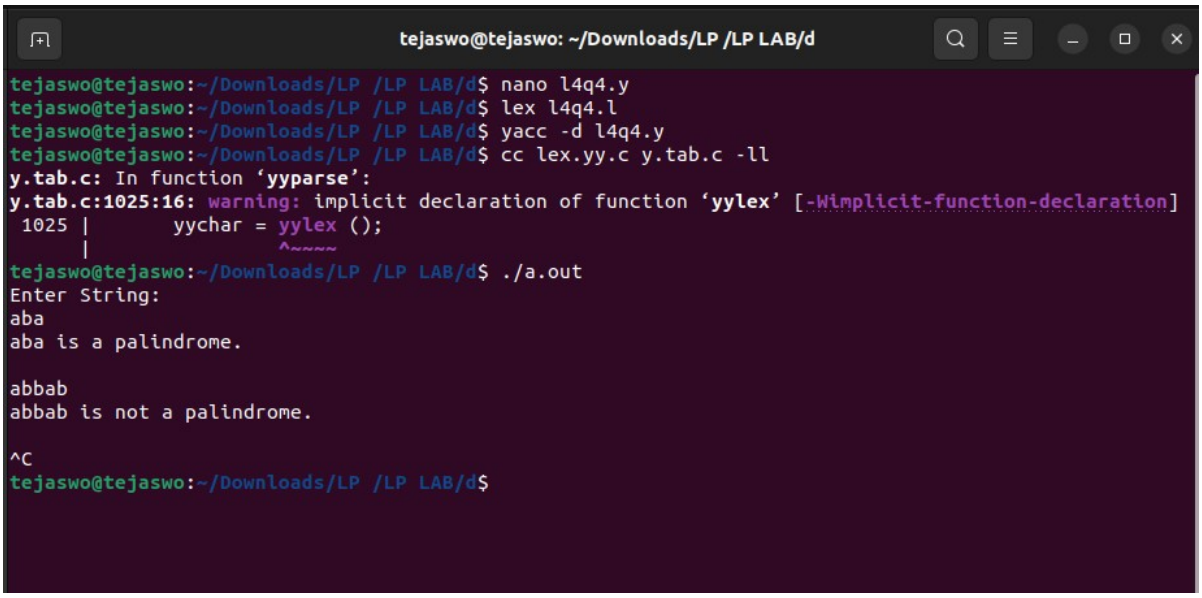
l4q4.l

```
%{
        #include <stdio.h>
        #include <stdlib.h>
        #include <string.h>
        #include "y.tab.h"
%}

%%
[a-zA-Z]+       {yylval.string = yytext; return STR;}
\n              {return newline;}
[ \t]           ;
.               ;
%%

int yywrap() {
        return 1;
}
```

l4q4.y

```
%{
  #include <stdio.h>
  #include <stdlib.h>
  #include <string.h>
  void yyerror(char *);

%}

%union {char *string;}
%start lines
%token newline
%token <string> STR
%type <string> E S

%%

lines:
   |   lines S newline     {;}
   ;
S :      E        {
                  int flag = 0;
                  int k = strlen($1) - 1;
                  for (int i = 0; i <= k/2; i++) {
                        if ($1[i] == $1[k-i]) {}
                        else {
                           flag = 1;
                        }
                  }
                  if (flag == 1) printf("%s is not a palindrome.\n\n",$1);
                  else printf("%s is a palindrome.\n\n",$1);
                  }
   ;
E :      STR        {$$ = $1;}
   ;
%%

int main() {
         printf("Enter String:\n");
         yyparse();
}

void yyerror(char *s) {
         printf("\nERROR!\n");
         exit(1);
}
```

OTUPUT:



5. YACC program for Conversion of Infix to Postfix expression.

CODE:

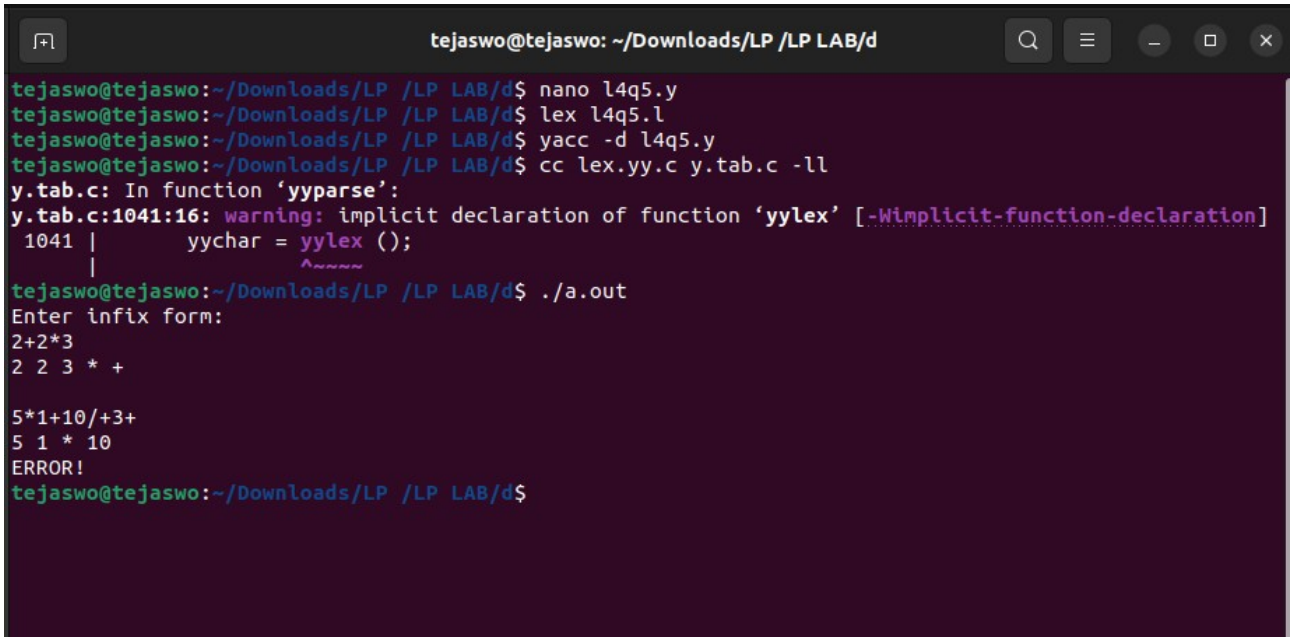
l4q5.l:

```
%{
	#include <stdio.h>
	#include <stdlib.h>
	#include "y.tab.h"
%}

%%

[0-9]+		{yylval.number = atoi(yytext); return num;}
[+*-/]		{return yytext[0];}
\n		{return newline;}

.		;

%%

int yywrap() {
	return 1;
}
```

l4q5.y:

```
%{
        #include <stdio.h>
        #include <stdlib.h>
        void yyerror(char *);

%}

%union {int number;}
%start lines
%token <number> num
%token newline
%token exits

%%

lines :  |lines E newline          {printf("\n\n");}
                ;

E :     E '+' T             {printf("+ ");}
        |   E '-' T         {printf("- ");}
        |   T               {;}
        ;

T :     T '*' F             {printf("* ");}
        |   T '/' F         {printf("/ ");}
        |   F       {;}
        ;

F :     num                 {printf("%d ", $1);}
        ;
%%

int main() {
        printf("Enter infix form: \n");
        yyparse();
}

void yyerror(char *s) {
        printf("\nERROR!\n");
        exit(1);
}
```

OUTPUT:


tejaswo@tejaswo: ~/Downloads/LP /LP LAB/d
tejaswo@tejaswo:~/Downloads/LP /LP LAB/d$ nano l4q5.y
tejaswo@tejaswo:~/Downloads/LP /LP LAB/d$ lex l4q5.l
tejaswo@tejaswo:~/Downloads/LP /LP LAB/d$ yacc -d l4q5.y
tejaswo@tejaswo:~/Downloads/LP /LP LAB/d$ cc lex.yy.c y.tab.c -ll
y.tab.c: In function ‘yyparse’:
y.tab.c:1041:16: warning: implicit declaration of function ‘yylex’ [-Wimplicit-function-declaration]
1041 | yychar = yylex ();
| ^~~~~
tejaswo@tejaswo:~/Downloads/LP /LP LAB/d$ ./a.out
Enter infix form:
2+2*3
2 2 3 * +
5*1+10/+3+
5 1 * 10
ERROR!
tejaswo@tejaswo:~/Downloads/LP /LP LAB/d$