If u r using the windows ka editor:-
lex filename.l
gcc lex.yy.c
.\a.exe

SPCC Practical Questions 2023

1	Questions	Marks

A	Write a program to create your own 'C' library using macros that can find the area of geometrical shapes (any 4)
	
	- Area.h [sq, rect, tri, circle(macros)]
	area.h
	#define c_area(r) 3.14*r*r
	#define s_area(s) s*s
	#define t_area(b,h) 0.5*b*h
	#define r_area(l,b) l*b
	
	area.cpp
	#include<iostream>
	#include<conio.h>
	#include"area.h"
	using namespace std;
	int main()
	{
	int b,h,s,r,l,x;
	float area;
	cout<<"1.Area of square\n2.Area of rectangle\n3.Area of triangle\n4.Area of Circle\n";
	cout<<"Enter the operation to be performed: ";
	cin>>x;
	switch(x)
	{
	case 1:
	cout<<"enter the side of square ";
	cin>>s;
	area=s_area(s);
	cout<<"Area of square is: "<<area;
	break;
	case 2:
	cout<<"enter the length and breadth of rectangle ";
	cin>>l>>b;
	area=r_area(l,b);
	cout<<"Area of rectangle is: "<<area;
	break;
	case 3:
	cout<<"enter the base and height of triangle ";
	cin>>b>>h;
	area=t_area(b,h);
	cout<<"Area of triangle is: "<<area;
	break;
		10 Marks

```
case 4:
cout<<"enter the radius of circle ";
cin>>r;
area=c_area(r);
cout<<"Area of circle is: "<<area;
break;
}
return 0;
```

B Write a LEX program to count and identify Vowels and consonants with output — 5 Marks

Code:

```
%{
#include <stdio.h>
#include <string.h>
int v=0;
int c=0;
%}

%%
[a-zA-Z]   {if (strchr("aeiouAEIOU", *yytext))
          {        printf("Vowel: %s\n", yytext);v++;  }
           else
          {        printf("Consonant: %s\n", yytext);c++;      }
}
.         { }
%%

int main()
{
          printf("Enter the String: ");
          yylex();
          printf("Vowels=%d & Consonants=%d",v,c);
}
int yywrap()
{
          return 1;
}
```

```
Enter the String: Hello
Consonant: H
Vowel: e
Consonant: l
Consonant: l
Vowel: o

Vowels=2 & Consonants=3
```

2	**Questions**	**Marks**
A	Write a program to convert the given computation into three address code. x = (a+b) * (c-d)	10 Marks

B	Write a LEX program to count and identify uppercase and lowercase letter with output
	Code:
	%{
	#include<stdio.h>
	int u=0;
	int l=0;
	%}
	
	%%
	[A-Z] {printf("Uppercase: ");ECHO;u++;printf("\t\n");}
	[a-z] {printf("Lowercase: ");ECHO;l++;printf("\t\n");}
	. ;
	%%
	
	int main()
	{
	printf("Enter a string\n");
	yylex();
	printf("Uppercase=%d and Lowercase=%d",u,l);
	}
	int yywrap()
	{
	return 1;
		5 Marks

}

```
Enter a string
Hello
Uppercase: H
Lowercase: e
Lowercase: l
Lowercase: l
Lowercase: o

Uppercase=1 and Lowercase=4
```

3	**Questions**	**Marks**

A	Write a program to create your own 'C' library using macros for conversions. (metre ⇔ feet, litre ⇔ cubic feet, °C ⇔ °F)
	convert.h
	#define fah(c) (c*1.8)+32
	#define cel(f) (f-32)/1.8
	#define metre(f) f*0.3048
	#define feet(m) m/0.3048
	#define cubft(l) l/28.3168507
	#define litre(cf) cf*28.3168507
	
	convert.cpp
	#include<iostream>
	#include<conio.h>
	#include"convert.h"
	using namespace std;
	int main()
	{
	float c,f,cf,l,m,result;
	int x;
	cout<<"1.Convert Fahrenheit to Celsius \n2.Convert Celsius to Fahrenheit
	\n3.Convert Feet to M\n4.Convert M to Feet\n5.Convert Litre to cubic
	feet\n6.Convert
	cubic feet to litre\n";
	cout<<"Enter the operation to be performed:"<<endl;
	cin>>x;
	switch(x)
	{
	case 1:
	cout<<"Enter the value for F: ";
	cin>>f;
	result=cel(f);
	cout<<"Result is: "<<result;
		10 Marks

```
break;
case 2:
cout<<"Enter the value for C: ";
cin>>c;
result=fah(c);
cout<<"Result is: "<<result;
break;
case 3:
cout<<"Enter the value for F: ";
cin>>f;
result=metre(f);
cout<<"Result is: "<<result;
break;
case 4:
cout<<"Enter the value for M: ";
cin>>m;
result=feet(m);
cout<<"Result is: "<<result;
break;
case 5:
cout<<"Enter the value for Litre: ";
cin>>l;
result=cubft(l);
cout<<"Result is: "<<result;
break;
case 6:
cout<<"Enter the value for Cubic feet: ";
cin>>cf;
result=litre(cf);
cout<<"Result is: "<<result;
break;
}
return 0;
}
```

B

Write a LEX program to count the number of characters, words, sentences, lines, tabs, numbers and blank spaces present in input

5 Marks

```
%option noyywrap
%{
  #include<stdio.h>
  int character = 0;
  int word = 0;
  int sentence = 0;
  int line = 0;
  int tab = 0;
```

```
  int number = 0;
  int space = 0;
  int total_character = 0;
%}

%%
[a-zA-Z]+[0-9]* {word++; character=character+yyleng;}
[\n] {line++;}
[.] {sentence++; character=character+yyleng;}
[\t] {tab++; character=character+yyleng;}
[0-9] {number++; character=character+yyleng;}
" " {space++; character=character+yyleng;}
\n\n {printf("Characters Count = %d\nWords Count = %d\nSentences Count =
%d\nLines Count = %d\nTabs Count = %d\nNumbers Count = %d\nSpaces
Count = %d",character,word,sentence,line+1,tab,number,space);}
%%

int main()
{
        printf("Enter Text : \n");
        yylex();
        return 0;
}
```

4	**Questions**	**Marks**
A	Write a program to convert the given computation into three address code. x = a+ b*c -d and Display Quadruples and Triples	10 Marks

B	Write a LEX program to count and identify tokens with output
	%{
	#include<stdio.h>
	int v=0,op=0,id=0;
	%}
	
	%%
		5 Marks

```
[0-9][0-9]* {id++;printf("\nIdentifier:");ECHO;}
[\+\-\*\/\=] {op++;printf("\nOperartor:");ECHO;}
"(" {v++;}
")" {v--;}
.|\n {return 0;}
%%


int main()
{
   printf("Enter the expression:\n");
   yylex();
   if((op+1) ==id && v==0)
   {
      printf("\n\nIdentifiers are:%d\nOperators are:%d\n",id,op);
      printf("\nExpression is Valid\n");
   }
   else
      printf("\nExpression is Invalid\n");
   return 1;
}
int yywrap()
{
   return 1;
}
```

```
cmb@fedora:~/Desktop
[cmb@fedora Desktop]$ lex op.l
[cmb@fedora Desktop]$ ls
ab.l   helloworld.l  token.l       vc.l      yexp.l   y.tab.c
ab.y   lex.yy.c      upperlower.l  vc_new.l  yexp.l.  y.tab.h
a.out  op.l          v_c.l         voco.l    yexp.y
[cmb@fedora Desktop]$ gcc lex.yy.c
[cmb@fedora Desktop]$ ./a.out
Enter the expression:
4+4-2

Identifier:4
Operartor:+
Identifier:4
Operartor:-
Identifier:2

Identifiers are:3
Operators are:2

Expression is Valid
```

```
[cmb@fedora Desktop]$ ./a.out
Enter the expression:
3-+

Identifier:3
Operartor:-
Operartor:+
Expression is Invalid
```

5	**Questions**	**Marks**
A	Write a program to create your own 'C' library using macros for conversions. (binary ⇔ decimal, binary ⇔ hexadecimal)	10 Marks

B	Write a LEX program to recognize valid arithmetic expressions
	%{
	#include<stdio.h>
	int v=0,op=0,id=0;
	%}
	
	%%
	[0-9][0-9]* {id++;printf("\nIdentifier: ");ECHO;}
	[\+\-*\/\=] {op++;printf("\nOperator: ");ECHO;}
	"(" {v++;}
	")" {v--;}
	.|\n {return 0;}
	%%
	
	int main()
	{
	printf("Enter the expre: \n");
	yylex();
	if((op+1) ==id && v==0)
	{
	printf("Identifier are: %d \n operator are: %d ",id,op);
	printf("|n Exp is valid\n");
	}
	else
	printf("|n Exp is Invalid\n");
	return 1;
	}
		5 Marks

6	**Questions**	**Marks**
A	Write a program to create your own 'C' library using macros to generate series. (Factorial, prime numbers, leap years)	10 Marks
B	Write a YACC program for Calculator performing four basic operations (+ , -, * and /)	5 Marks

7	**Questions**	**Marks**
A	Write a program to create your own 'C' library using macros to generate series. (Fibonacci Series, prime numbers, leap years)	10 Marks
B	Write a YACC program that accepts all the strings ending with b preceded by any number of a's (a^nb)	5 Marks

anb.l

```
%{
#include "y.tab.h"
%}

%%
a { yylval = *yytext; return A; }
b { yylval = *yytext; return B; }
\n { return NL; }
. { return yytext[0]; }
%%

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

Anb.y

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

%token A B NL

%%
input: line NL | input line NL ;
line: As Bs ;

As: | As A ;

Bs: B ;
%%

int main() {
        yyparse();
        return 0;
}

void yyerror(char* s) {
        printf("The text is not of type (a^n b) \n");
}
```

8	**Questions**	**Marks**
A	Write a program to convert the given computation into three address code and Display Quadruples and Triples *x = a*b/c+d;*	10 Marks
B	Write a YACC program that accepts all the strings ending with b preceded by any number of a's ($a^n b^n$)	5 Marks

9	**Questions**	**Marks**
A	Write a program to create your own ‘C’ library using macros to find the properties of a given number n – factorial of n, sum of natural numbers till n	10 Marks
B	Write a YACC program that accepts all the strings ending with b preceded by any number of a’s ($a^n b^{n+1}$)	5 Marks

10	**Questions**	**Marks**

A	Consider the following program, Display the Pass-1 of the Program
	START 501
	A DS 1
	B DS 1
	C DS 1
	READ A
	READ B
	MOVER AREG, A
	ADD AREG, B
	MOVEM AREG, C
	PRINT C
	END
		10 Marks

B	Write a YACC program that accepts all the strings ending with b preceded by any number of a’s ($a^{2n} b^n$)	5 Marks

11	**Questions**	**Marks**
A	For the given program, Display the Pass-2 by taking intermediate code as an input Assembly program LC **Intermediate code (PASS-1)** START 501 **(AD,01) (c,501)** A DS 1 501 **(S,0) (DL,0) (c,1)** B DS 1 502 **(S,1) (DL,0) (c,1)** C DS 1 503 **(S,2) (DL,0) (c,1)** READ A 504 **(IS,09) (S,0)** READ B 505 **(IS,09) (S,1)** MOVER AREG, A 506 **(IS,04) (RG,01) (S,0)** ADD AREG, B 507 **(IS,01) (RG,01) (S,1)** MOVEM AREG, C 508 **(IS,05) (RG,01) (S,2)** PRINT C 509 **(IS,10) (S,2)** END 510 **(AD,02)**	10 Marks
B	Write a LEX program to count number of lines, numbers and blank spaces.	5 Marks

12	**Questions**	**Marks**

A	Consider the following Three address code as Input and display Triples and Quadruples f=c+d e=a-f g=b*e	10 Marks

```
#include <iostream>
#include <vector>
#include <string>

using namespace std;

void qQuadruple(vector<string> expression) {
    cout << "op\targ1\targ2\tresult" << endl;
    for (int i = 0; i < expression.size(); i++) {
        string expR = expression[i];
        char op = expR[3];
        char arg1 = expR[2];
        char arg2 = expR[4];
        char result = expR[0];
        cout << op << "\t" << arg1 << "\t" << arg2 << "\t" << result << endl;
    }
}

void tTriples(vector<string> expression) {
    cout << "#\top\targ1\targ2" << endl;
    int c = 0;
    for (int i = 0; i < expression.size(); i++) {
        string expR = expression[i];
        char op = expR[3];
        char arg1 = expR[2];
        char arg2 = expR[4];
        cout << i+c << "\t" << op << "\t" << arg1 << "\t" << arg2 << endl;
        if (expR[0] != NULL) {
            ++c;
            cout << i+c << "\t" << expR[1] << "\t" << expR[0] << "\t" << i+c-1 <<
endl;
        }
    }
}
```

```
int main() {
    vector<string> exp;
    int n;
    string input;
    cout << "Enter the number of expressions: ";
    cin >> n;
    cin.ignore(); // To consume the newline character after the integer input
    cout << "Enter the expressions: " << endl;
    for (int i = 0; i < n; i++) {
        getline(cin, input);
        exp.push_back(input);
    }
    cout << "Quadruple:" << endl << endl;
    qQuadruple(exp);
    cout << endl << "Triple:" << endl << endl;
    tTriples(exp);
    return 0;
}
```

```
Enter the number of expressions: 3
Enter the expressions:
f=c+d
e=a-f
g=b*e
Quadruple:

op      arg1    arg2    result
+       c       d       f
-       a       f       e
*       b       e       g

Triple:

#       op      arg1    arg2
0       +       c       d
1       =       f       0
2       -       a       f
3       =       e       2
4       *       b       e
5       =       g       4
```

B	Write a YACC program that accepts all the strings ending with b preceded by any number of a's ($a^n b^n c^n$)	5 Marks

13	**Questions**	**Marks**

A	Write a program to optimize the given three address code.
	T1= 5*3+10 // Constant folding
	T3=T1 //variable propagation
	T2=T1+T3
	T5=4*T2 // common sub-expression elimination
	T6=4*T2+100
		10 Marks

B	Write a LEX program to count the number of tokens with uppercase characters.	5 Marks

14	**Questions**	**Marks**

A	Write a program to generate the three address code of
	pi = 3.145;
	x = a * pi * 180 + b * pi * 2;
		10 Marks

B	Write a LEX program to check valid Mobile Number (10 digit)
	%{
	#include<stdio.h>
	%}
	
	%%
	
	^[0-9]{10}$ {
	printf("Valid mobile number\n");
	return 0;
	}
	
	.* {
	printf("Invalid mobile number\n");
	return 1;
	}
	
	%%
		5 Marks

```
int main() {
   yylex();
   return 0;
}
```

15	Questions	Marks
A	Write a C/ C++/ Javaprogram to to design lexical analyzer for a language whose grammar is known. LINE □If PHRASE then ACTION. LINE / ∈ PHRASE□NOUN VERB NOUN NOUN□(a-z) * VERB□hate / like ACTION□they NOUN **Input:** "If dogs hate cats then they chase. $" **Output**:(k) (n,1) (v) (n,2) (k) (a) (n,3) (op) Identify and count the number of tokens	10 Marks

```
#include <iostream>
#include <string>
#include <vector>
#include <bits/stdc++.h>
using namespace std;
vector<string> keyword {"if","then"};
vector<string> action {"they"};
vector<string> verb {"hate","like"};
vector<string> noun;
int k=0,a=0,v=0,n=0,o=0,ex=0;

int location(vector<string> vect, string arg){
   vector <string> :: iterator itr;
   itr = find (vect.begin(), vect.end(), arg);
   if (itr != vect.end ()){
      int index = distance(vect.begin (), itr);
      return index;
   }
   else{
      return -1;
```

```
    }
}

void result(string token){
    if (location(keyword,token)!=-1){
        cout<<"Keyword : "<<token<<endl;
        k+=1;    }
    else if(location(action,token)!=-1){
        cout<<"Action  : "<<token<<endl;
        a+=1;  }
    else if(location(verb,token)!=-1){
        cout<<"Verb    : "<<token<<endl;
        v+=1;  }
    else{
        if(location(noun,token)!=-1){
            cout<<"Noun "<<location(noun,token)+1<<"  : "<<token<<endl;
            ex+=1;   }
        else{
            cout<<"Noun "<<n+1<<"  : "<<token<<endl;
            noun.push_back(token);
            n+=1;        }  }   }

int main(){
    string inp,token="";
    cout<<"\nEnter the state for lexical analyzing: ";
    getline(cin,inp);
    cout<<"\n";
    for ( int i=0;  i<inp.length(); i++){
        if (inp[i]!=' '){
            if (inp[i]=='.'){
                if (token!=""){
                    result(token);
                    token="";   }
                cout<<"Operator(.)\n"<<endl;
                o+=1;  }
```

```
            else{
                token+=inp[i];
            }   }
        else{
            if (token!=""){
                result(token);
                token="";
            }
        }
    }
    cout<<"<eof>\n"<<endl;
    cout<<"Symbol table -\n";
    for(int i=0; i < noun.size(); i++){
        cout<<i+1<<"  "<<noun[i]<<endl;
    }
    cout<<"\nTotal number of keywords   : "<<k<<endl;
    cout<<"Total number of actions    : "<<a<<endl;
    cout<<"Total number of verbs      : "<<v<<endl;
    cout<<"Total number of nouns      : "<<n+ex<<endl;
    cout<<"Total number of operators  : "<<o<<endl;
    cout<<"-----------------------------"<<endl;
    cout<<"Total number of tokens     : "<<k+a+v+n+o+ex<<"\n"<<endl;
    return 0;
}
```

OUTPUT:

```
Enter the state for lexical analyzing: if dogs hate cats then they chase. if cats like milk then they drink. $

Keyword : if
Noun 1  : dogs
Verb    : hate
Noun 2  : cats
Keyword : then
Action  : they
Noun 3  : chase
Operator(.)

Keyword : if
Noun 2  : cats
Verb    : like
Noun 4  : milk
Keyword : then
Action  : they
Noun 5  : drink
Operator(.)

<eof>

Symbol table -
1  dogs
2  cats
3  chase
4  milk
5  drink

Total number of keywords   : 4
Total number of actions    : 2
Total number of verbs      : 2
Total number of nouns      : 6
Total number of operators  : 2
----------------------------
Total number of tokens     : 16
```

B

Lex program to take check whether the given number is even or odd

5 Marks

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

DIGIT [0-9]

%%
{DIGIT}+    {
            int num = atoi(yytext); //atoi is ASCII to integer
            if (num % 2 == 0) {
                printf("%d is even.\n", num);
            } else {
                printf("%d is odd.\n", num);
            }
        }
%%

int main() {
    yylex();
    return 0;
}
```