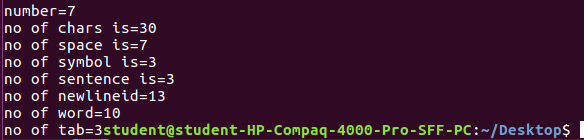
Exp 1

LETTER,WORD…

|  |  |
| --- | --- |
| **Lex program:**  %{  int c=0;  int b=0;  int d=0;  int f=0;  int n=0;  int e=0;  int s=0;  %}  %%  [0-9]+ c++;  [a-z|A-Z] b++;  [' '] d++;  [=\*%+&]\* f++;  [.] n++;  [\n] e++;  [\t] s++;  %%  int main()  {  yyin=fopen("num.txt","r");  yylex();  printf("\nnumber=%d",c);  printf("\nno of chars is=%d",b);  printf("\nno of space is=%d",d);  printf("\nno of symbol is=%d",f);  printf("\nno of sentence is=%d",n);  printf("\nno of newlineid=%d",e);  printf("\nno of word=%d",(d+n));  printf("\nno of tab=%d",s);  return 0;  } | **Text file**  0  1  2  3  4  5  6  a b c.  \*  +  &  Hello have a Nice day re.  hi komals. |

**Output:**



Exp 2

IDENTIFIER

**Lex file:**

%{

int add(char[]);

char ar[10][10];

int i=0;

%}

%%

#include<.[a-z]+.h> {printf("%s include statement\n",yytext);}

int|main|return {printf("%s keyword\n",yytext);}

"("|"{" {printf("%s opening symbols\n",yytext);}

")"|"}" {printf("%s closing symbols\n",yytext);}

[a-zA-Z][a-zA-z0-9]\* {printf("%s identifier \n",yytext);add(yytext);}

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

[+|=] {printf("%s operators\n",yytext);}

%%

int main()

{

yyin=fopen("exp3.c","r");

yylex();

return 0;

}

int add(char s[])

{

strcpy(ar[i],s);

printf("identifier added in symbol table");

i++;

}

**C file:**

#include<stdio.h>

int main()

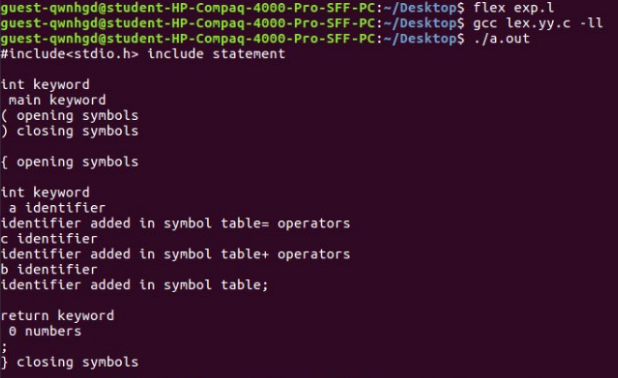
{

int a=c+b;

return 0;

}

**Output:**

****

Exp 3

ERITHMETIC EXPRESSION

e.l FILE

%{

#include"y.tab.h"

extern int yylval;

%}

%%

[0-9]+ {yylval=atoi(yytext); return NUM;}

\n {return 0;}

.{ return yytext[0];}

%%

\*\*\*\*\*\*\*\*eX3.y\*\*\*\*\*\*\*\*\*\*\*

%{

#include<stdio.h>

#include<math.h>

#include<stdlib.h>

%}

%token NUM

%left '+' '-'

%%

stmt :exp {printf("Answer: %d \n",$1);}

;

exp: exp '+' exp {$$=$1+$3;}

| exp '-' exp {$$=$1-$3;}

| NUM{$$=$1;}

;

%%

int main()

{

printf("Enter the Arithmetic expression\n");

yyparse();

printf("Valid expression!");

return 0;

}

yyerror()

{

printf("Invalid erpression!");

exit(0);

}

int yywrap()

{

return 1;

}

OUTPUT:

student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./a.out

Enter the Arithmetic expression

4+3

Answer: 7

Valid expression!

student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./a.out

Enter the Arithmetic expression

15-

Invalid erpression!

student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./a.out

Enter the Arithmetic expression

9

Answer: 9

Valid expression!student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$

**Exp 4**

**CALCULATOR**

\*\*\*\*\*\*\*4.l\*\*\*\*\*\*\*\*\*

%{

#include "y.tab.h"

%}

%%

[0-9]+ {yylval.num=atof(yytext); return number;}

[-+\*/] {return yytext[0];}

COS|cos {return cos1;}

SIN|sin {return sin1;}

TAN|tan {return tan1;}

%%

int yywarp()

{

return 1;

}

\*\*\*\*\*\*\*\*4.y\*\*\*\*\*\*\*

%{

#include<stdio.h>

#include<math.h>

%}

%union {float num;}

%start line

%token cos1

%token sin1

%token tan1

%token <num> number

%type <num> exp

%%

line:exp

|line exp

;

exp:number {$$=$1;}

| exp'+' number {$$=$1+$3;printf("\n%f+%f=%f\n",$1,$3,$$);}

| exp'-' number {$$=$1-$3;printf("\n%f-%f=%f\n",$1,$3,$$);}

| exp'\*' number {$$=$1\*$3;printf("\n%f\*%f=%f\n",$1,$3,$$);}

| exp'/' number {$$=$1/$3;printf("\n%f/%f=%f\n",$1,$3,$$);}

| cos1 number {printf("%f",cos(($2/180)\*3.14));}

| sin1 number {printf("%f",sin(($2/180)\*3.14));}

| tan1 number {printf("%f",tan(($2/180)\*3.14));}

;

%%

int main()

{

yyparse();

return 0;

}

int yyerror()

{

exit(0);

}

\*\*\*\*\*\*Output:\*\*\*\*\*\*\*\*\*\*\*

student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ flex 4.l

student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ yacc -d 4.y

student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ gcc lex.yy.c y.tab.c -ll -lm

student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ ./a.out

sin90

1.000000

tan 45

0.999204

cos60

0.500460

sin30

0.499770

6/2

6.000000/2.000000=3.000000

6+2

6.000000+2.000000=8.000000

**Exp 5**

**FIRST AND FOLLOW**

import java.io.\*;

import java.util.\*;

class follow

{

public static void main(String args[])

{

char t,d;

Vector g=new Vector();

g.addElement("S-ab");

g.addElement("S-Ac");

g.addElement("A-eB");

g.addElement("B-f");

System.out.println(g);

LL1 ob=new LL1();

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

{

String s=g.elementAt(i).toString();

char c=s.charAt(2);

t=ob.first(c,g);

System.out.println("First of "+ s.charAt(0)+ "="+t);

}

d=ob.follow('S',g);

System.out.println("Follow of S="+d);

d=ob.follow('A',g);

System.out.println("Follow of A="+d);

d=ob.follow('B',g);

System.out.println("Follow of B="+d);

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

{

String s=g.elementAt(i).toString();

char c=s.charAt(2);

t=ob.parse(c,g);

System.out.println("M["+ s.charAt(0)+","+t+"]->"+s);

}

}

}

class LL1

{

char first(char a,Vector g)

{

char t1=' ';

if(a=='#')

{ return a;}

else if(a>='a'&&a<='z')

{ return a;}

else

{

for(int j=0;j<4;j++)

{

String temp=g.elementAt(j).toString();

if(temp.charAt(0)==a)

{

t1=temp.charAt(2);

}

}

return first(t1,g);

}

}

char follow(char b,Vector g)

{

if(b=='S')

{ return '$'; }

else

{ char t2=' ';

for(int j=0;j<4;j++)

{

String temp=g.elementAt(j).toString();

for(int k=2;k<temp.length();k++)

{

if(temp.charAt(k)==b)

{

if(k==temp.length()-1)

{ t2=follow(temp.charAt(0),g); }

else{

t2=first(temp.charAt(k+1),g);}

break;

}

}

}

return t2;

}

}

char parse(char c,Vector g)

{

char t1=' ';

if(c>='a' && c<='z')

{

t1=first(c,g);

}

else if(c>='A' && c<='Z')

{

t1=first(c,g);

}

else if(c=='#')

{

t1=follow(c,g);

}

return t1;

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\* OUTPUT \*\*\*\*\*\*\*\*\*\*\*\*

student@student-SMBIOS-newer-than-version:~/Desktop$ javac follow.java

student@student-SMBIOS-newer-than-version:~/Desktop$ java follow[S-ab, S-Ac, A-eB, B-f]

First of S=a

First of S=e

First of A=e

First of B=f

Follow of S=$

Follow of A=c

Follow of B=c

M[S,a]->S-ab

M[S,e]->S-Ac

M[A,e]->A-eB

M[B,f]->B-f

**Exp 6**

**3 ADDRESS CODE**

teach5a.l

%{

#include"y.tab.h"

%}

%%

[a-zA-Z]+ {strcpy(yylval.str,yytext); return Var;}

[0-9]+ {strcpy(yylval.str,yytext); return Num;}

\n {return 0;}

. {return yytext[0];}

%%

int yywrap()

{

return 1;

}

teach5a.y

%{

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

char \* createT();

int tempcount=0;

int top=-1;

%}

%union

{

char str[30];

}

%left '+'

%left '-'

%left '\*'

%left '/'

%token <str> Var

%token <str> Num

%type <str> s

%type <str> exp

%%

s: Var '=' exp {printf("\n%s=%s\n",$1,$3);}

exp: '(' exp ')' {strcpy($$,$2);}

| exp '+' exp {strcpy($$,createT());printf("\n%s=%s+%s",$$,$1,$3);}

|exp '-' exp {strcpy($$,createT());printf("\n%s=%s-%s",$$,$1,$3);}

| exp '\*' exp {strcpy($$,createT());printf("\n%s=%s\*%s",$$,$1,$3);}

| exp '/' exp {strcpy($$,createT());printf("\n%s=%s/%s",$$,$1,$3);}

| Num {strcpy($$,$1);}

| Var {strcpy($$,$1);}

%%

char \* createT()

{

char snum[30],\*ptr;

sprintf(snum,"t%d",tempcount);

ptr=snum;

tempcount++;

return ptr;

}

int main()

{

yyparse();

return 0;

}

int yyerror(char \*err)

{

printf("\nInvlaid");

exit(0);

}

\*\*\*\*\*\*\*\*\*\*\* OUTPUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

student@kshitij:~/Desktop$ flex teach5a.l

student@kshitij:~/Desktop$ yacc -d teach5a.y

student@kshitij:~/Desktop$ gcc lex.yy.c y.tab.c -ll

student@kshitij:~/Desktop$ ./a.out

b=x+y/c+d\*p

t0=y/c

t1=x+t0

t2=d\*p

t3=t1+t2

b=t3

**Exp 7**

**DEAD CODE ELIMINATION**

import java.util.\*;

class dc

{

public static void main(String []args)

{

Vector <String> v=new Vector();

v.addElement("x=10");

v.addElement("y=z+b");

v.addElement("z=z-a");

int c=0;

System.out.println("The value of code is\n" +v);

String s1=new String();

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

{

s1=v.elementAt(i).toString();

String t=new String(s1.substring(0,1));

for(int j=i+1;j<2;j++)

{

String s2=v.elementAt(j).toString();

if(!s2.contains(t)){

c=i;

break;

}

}

}

v.removeElementAt(c);

System.out.println("The value of after dead code elimination is\n" +v);

}

}

**Output:**

student@student-HP-Compaq-4000-Pro-SFF-PC:~/Desktop$ java dc

The value of code is

[x=10, y=z+b, z=z-a]

The value of after dead code elimination is

[y=z+b, z=z-a]

**Exp 8**

**CODE GENERATION**

import java.io.\*;

class CodeGen

{

public static void main(String args[])

{

Console c= System.console();

int i=0,a,b;

int R[] =new int[20];

String s1,s2,s3,s4;

String s[]=new String[5];

int n=Integer.parseInt(c.readLine("enter no of operations:"));

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

{

s[j]=c.readLine("\nenter Statement:");

}

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

{

a=s[j].indexOf('=');

b=3;

s1=s[j].substring(0,a);

s2=s[j].substring(a+1,b);

s3=s[j].substring(b+1);

s4=s[j].substring(b,b+1);

System.out.printf("\nLDF R%d,%s",++i,s2);

System.out.printf(" \nLDF R%d,%s",++i,s3);

switch(s4)

{

case "\*":

System.out.printf("\nMULF R%d,R%d",--i,++i); break;

case "/":

System.out.printf("\nDIVF R%d,R%d",--i,++i); break;

case "+":

System.out.printf("\nADDF R%d,R%d",--i,++i); break;

case "-":

System.out.printf("\nSUBF R%d,R%d",--i,++i); break;

}

System.out.printf("\nSTF %s,R%d\n",s1,--i);

++i;

}

}

}

/\*OUTPUT:

enter no of operations:2

enter Statement:x=y+z

enter Statement:Z=A\*B

LDF R1,y

LDF R2,z

ADDF R1,R2

STF x,R1

LDF R3,A

LDF R4,B

MULF R3,R4

STF Z,R3 \*/

**exp 9**

**ASSESMBLER**

**1.Mnemonic OpcodeTable**

from tabulate import tabulate

lc=[0,0,0,4,8,8,12,16,20,24]

lab=["program","","","","data1","","four","five","temp",""]

ins=["start","using","A","A","EQU","ST","DC","DC","DS","END"]

op1=[0,0,1,2,"F10",1,"F4","F5","1F"]

op2=["",15,"FOUR","FIVE","","F4"]

for i in range(0,len(ins)):

if ins[i]=="A":

ins[i]=4

elif ins[i]=="ST":

ins[i]=4

#mnemonicopcodetablecontents---------

print(ins)

print('MnemonicOpcodeTable')

print

printtabulate([['A',4],['ST',4]],headers=['INSTRUCTION','Value'])

**Output:**

Python2.7.10(default,Jul142015,19:46:27)

['start','using',4,4,'EQU',4,'DC','DC','DS','END']

MnemonicOpcodeTable

INSTRUCTION Value

------------- -------

A 4

ST 4

**2.Base Table**

from tabulate import tabulate

lc =[0,0,0,4,8,8,12,16,20,24]

lab =["program","","","","data1","","four","five","temp",""]

ins =["start","using","A","A","EQU","ST","DC","DC","DS","END"]

op1 =[0,0,1,2,"F10",1,"F4","F5","1F"]

op2 =["",15,"FOUR","FIVE","","F4"]

x = []

for i in range(0,len(ins)):

if ins[i]=="using": x.append(op2[i])

x.append("Y") x.append(0)

#base table contents---------

print(x)

for i in range(0,len(ins)):

if ins[i]=="A":

ins[i]=4

elif ins[i]=="ST":

ins[i]=4

print(ins) print

print('Base Table')

print tabulate([[15, 'y', 0]], headers=['a','b','c'])

**Output:**

Python 2.7.10 (default, Jul 14 2015, 19:46:27)

[15, 'Y', 0]

['start', 'using', 4, 4, 'EQU', 4, 'DC', 'DC', 'DS', 'END']

Base Table

a b c

--- ---- ----

15 y 0

**3. Pseudo-op Table (POT)**

lc=[0,0,0,4,8,8,12,16,20,24]

lable=['PROGRAM',' ',' ',' ','DATA1',' ','FOUR','FIVE','TEMP',' ']

Inst=['START','USING','A','A','EQU','ST','DC','DC','DS','END']

Op1=[0,'\*',1,2,'F10',1,'F4','F5','1F',' ']

Op2=[' ',15,'FOUR','FIVE',' ','TEMP',' ',' ',' ',' '

pot={'START':'R1', 'USING':'R2', 'DC':'R3', 'DS':'R4', 'EQU':'R5', 'END':'R6'}

for i in Inst:

if i in pot.keys():

print("For "+i+" "+"routine "+pot[i]+" is called")

**OUTPUT:**

For START routine R1 is called

For USING routine R2 is called

For EQU routine R5 is called

For DC routine R3 is called

For DC routine R3 is called

For DS routine R4 is called

For END routine R6 is called

**4. Symbol Table:**

label=['program','-','-','-','data1','-','four','five','temp','-']

instruction=['start','using','add','add','equ','st','dc','dc','ds','end']

op1=['0','\*','1','2','10','1','4','5','1','-']

op2=['-','15','four','five','-','temp','-','-','-','-']

lc1=0

print('name',end='\t')

print('value',end='\t')

print('length',end='\t')

print('R/A',end='\t')

print()

for i in range(0,len(label)):

if(label[i]!='-'):

print(label[i],end="\t")

if(label[i]=='four' or label[i]=='five' or label[i]=='temp'):

print(lc1,end="\t")

else:

print(op1[i],end="\t")

if(label[i]=='program'):

print(1,end="\t")

else:

print(4,end="\t")

if(label[i]=='data1'):

print('A',end="\t")

else:

print('R',end="\t")

print(" ")

if(instruction[i]=='add' or instruction[i]=='st' or instruction[i]=='dc' or instruction[i]=='ds'):

lc1=lc1+4

**OUTPUT**

name value length R/A

program 0 1 R

data1 10 4 A

four 12 4 R

five 16 4 R

temp 20 4 R

**5. LITERAL TABLE**

import java.util.\*;

import java.lang.\*;

class Literal

{

public static void main(String args[])

{

int i,lc\_ct=0,flag=0,index=0;

//ArrayList<String> xy = new ArrayList<String>();

Vector vop1=new Vector();

Vector vop2=new Vector();

int arrayLc[];

char xy;

String[] arrayLabel= new String[] {"PROGRAM","","","","DATA1","","FOUR","FIVE","TEMP",""};

String[] arrayinst=new String[] {"START","USING","A","A","EQU","ST","DC","DC","DS","END"};

vop1.add("0"); vop1.add("\*"); vop1.add("1"); vop1.add("2"); vop1.add("F10");

vop1.add("1"); vop1.add("F4"); vop1.add("F5"); vop1.add("1F"); vop1.add("NULL");

vop2.add("\0"); vop2.add("15"); vop2.add("FOUR"); vop2.add("FIVE"); vop2.add("\0");

vop2.add("=F4");vop2.add("\0"); vop2.add("\0"); vop2.add("\0"); vop2.add("\0");

for(i=3;i<10;i++)

{

if(arrayinst[i]=="A" ||arrayinst[i]=="DC" ||arrayinst[i]=="DS" || arrayinst[i]=="END"||arrayinst[i]=="ST")

lc\_ct=lc\_ct+4;

String word =(String)vop2.get(i);

xy=word.charAt(0);

if(xy== '=' )

{

flag=1;

index=i;

}

}

System.out.println("\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LITERAL TABLE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");

if(flag==1)

System.out.println("NAME: "+vop2.get(index).toString()+" | VALUE: "+lc\_ct+" | LENGTH:4 | R/A: R");

System.out.println("LOCATION COUNT:"+lc\_ct);

}

}

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*OUTPUT\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

root1@root1-HP-280-G1-MT:~/Desktop$ javac Literal.java

Note: Literal.java uses unchecked or unsafe operations.

Note: Recompile with -Xlint:unchecked for details.

root1@root1-HP-280-G1-MT:~/Desktop$ java Literal

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*LITERAL TABLE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

NAME: =F4 | VALUE: 24 | LENGTH:4 | R/A: R

LOCATION COUNT:24

EXP 11

MACRO

**INPUT**

pass.txt

MACRO

&LAB ADDS &ARG1 &ARG2 &ARG3

A 1, &ARG1

A 2, &ARG2

A 3, &ARG3

MEND

D1 DC F4

D2 DC F5

D3 DC F6

FIRST ADDS D1,D2,D3

**\*\*\*\* MNT \*\*\***

import java.io.\*;

import java.util.\*;

class Macro

{

public static void main(String args[])

{

Scanner src=new Scanner(System.in);

String mn[]=new String[50];

int mdti[]=new int[50];

int c=0,a=0;

System.out.println("Enter the name of file with format");

String f=src.next();

try

{

FileReader fr=new FileReader(f);

BufferedReader br = new BufferedReader(fr);

String x;

while((x=br.readLine())!=null)

{

if(x.equals("MACRO"))

{

x=br.readLine();

int i = x.indexOf(" ");

mn[c]=x.substring(0,i);

mdti[c]=c+1;

while(!x.equals("MEND"))

{

x=br.readLine();

}

c++;

}

}

}

catch(Exception e){ System.out.println(e);}

System.out.println("\*\*\*\*\*\*\*\* MNT \*\*\*\*\*\*\*\*\*");

System.out.println("Index\tName \tMDT index");

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

{

System.out.println((j+1)+"\t"+mn[j]+"\t\t"+mdti[j]);

}

System.out.println();

}

}

**OUTPUT**

student@student-SMBIOS-newer-than-version:~/Desktop$ javac Macro.java

student@student-SMBIOS-newer-than-version:~/Desktop$ java Macro

Enter the name of file with format : pass.txt

**\*\*\*\*\*\*\*\* MNT \*\*\*\*\*\*\*\*\***

Index Name MDT index

1 ADDS 1

**\*\*\*\* MDT \*\*\*\***

**Macro.py**

word\_list=[]

l1=[]

l2=[]

l3=[]

l4=[]

count=1

argc=1

f = open('sample.txt')

for word in f.read().split():

word\_list.append(word)

final\_list=word\_list

index=final\_list.index('MACRO')

print("|------------------MACRO DEFINITION TABLE-------------------|")

print("|MDTC | INSTRUCTION |")

print("|-----------------------------------------------------------|")

if('MACRO' in final\_list):

index=final\_list.index('MACRO')+1

if('&LAB' in final\_list or 'ADDS' in final\_list or'ARG1' in final\_list or'ARG2' in final\_list):

l1.append(final\_list[index])

index=index+1

l1.append(final\_list[index])

index=index+1

if any("ARG" in s for s in final\_list):

l1.append(final\_list[index])

index=index+1

l1.append(final\_list[index])

index=index+1

l1.append(final\_list[index])

index=index+1

print(''+str(count)+'\t'+str(l1))

count=count+1

if('A' in final\_list):

l2.append('A')

l2.append(argc)

l2.append('#'+str(argc))

index=index+1

print(''+str(count)+'\t'+str(l2))

count=count+1

if('A' in final\_list):

l3.append('A')

l3.append(argc+1)

l3.append('#'+str(argc+1))

index=index+1

print(''+str(count)+'\t'+str(l3))

count=count+1

if('A' in final\_list):

l4.append('A')

l4.append(argc+2)

l4.append('#'+str(argc+2))

index=index+1

print(''+str(count)+'\t'+str(l4))

count=count+1

if('MEND' in final\_list):

print(''+str(count)+'\t'+'MEND')"""

**OUTPUT:**

RESTART: C:\Users\Rachana\AppData\Local\Programs\Python\Python37-32\Scripts\macro.py

|------------------MACRO DEFINITION TABLE-------------------|

|MDTC | INSTRUCTION |

|-------------------------------------------------------------------------|

1 ['&LAB', 'ADDS', '&ARG1', '&ARG2', '&ARG3']

2 ['A', 1, '#1']

3 ['A', 2, '#2']

4 ['A', 3, '#3']

5 MEND

**ALA**

import java.util.\*;

import java.lang.\*;

class ala

{

public static void main(String args[])

{

int i,m;

String[] arrayIndex=new String[4];

String[] arrayName=new String[4];

String[] arrayIndex4=new String[4];

String[] arrayName4=new String[4];

String[] arrayInst1={" ","MACRO"," "," "," "};

String[] arrayInst2={"&LAB","ADDS","&Arg1","&Arg2","&Arg3"};

int k=0;

System.out.println(arrayInst1[1]);

if(arrayInst1[1]=="MACRO")

{

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

arrayIndex[i]="#"+i;

}

for(i=0;i<5;i++){

if(i!=1){

arrayName[k]= arrayInst2[i];

k++;}

}

}

System.out.println("\n\t ALA TABLE");

System.out.println("INDEX | NAME");

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

{

System.out.println((arrayIndex[i])+" | "+(arrayName[i]));

}

String[] arrayInst3={"FIRST","ADDS","D1","D2","D3"};

int x=0;

for(m=0;m<4;m++){

arrayIndex4[m]="#"+m;

}

for(m=0;m<5;m++){

if(m!=1){

arrayName4[x]= arrayInst3[m];

x++;}

}

System.out.println("\n\tALA TABLE 2");

System.out.println("INDEX | NAME");

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

{

System.out.println((arrayIndex4[m])+" | "+(arrayName4[m]));

}

}

}

**\*\*\*\*\*\*\*\*\*\* OUTPUT \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***

C:\Program Files\Java\jdk1.7.0\bin>javac ala.java

C:\Program Files\Java\jdk1.7.0\bin>java ala

ALA TABLE

INDEX | NAME

#0 | &LAB

#1 | &Arg1

#2 | &Arg2

#3 | &Arg3

ALA TABLE 2

INDEX | NAME

#0 | FIRST

#1 | D1

#2 | D2

#3 | D3