**INDERPRASTHA ENGINEERING COLLEGE GHAZIABAD**



**Department of Information Technology**

**Compiler Design Lab (RCS-652)**

**(2019-20)**

|  |  |  |
| --- | --- | --- |
| **Name** | **:** | **Natasha Sharma** |
| **Roll Number** | **:** | **1703013043** |
| **Course** | **:** | **B.Tech. (I.T.)** |
| **Year** | **:** | **3** |
| **Semester** | **:** | **6** |
| **Section** | **:** | **A** |

**INDEX**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S.No.** | **OBJECTIVE** | **DATE** | **PAGE No** | **SIGN** | **Remark** |
|  |  |  |  |  |  |
| 1. | Write a C program to implement the | 24-Jan-2020 | 1-3 |  |  |
|  | Lexical Analyzer for If Statement |  |  |  |  |
|  |  |  |  |  |  |
| 2. | Write a C program to implement the | 07-Feb-2020 | 4-6 |  |  |
|  | Lexical Analyzer for Arithmetic |  |  |  |  |
|  | Expression |  |  |  |  |
|  |  |  |  |  |  |
| 3. | Write a C program to identify whether | 07-Feb-2020 | 7-8 |  |  |
|  | a given line is a comment or not |  |  |  |  |
|  |  |  |  |  |  |
| 4. | Write a C program to recognize strings | 14-Feb-2020 | 9-12 |  |  |
|  | under a\*, a\*b+, abb |  |  |  |  |
|  |  |  |  |  |  |
| 5. | Write a C program to test whether a | 14-Feb-2020 | 13-14 |  |  |
|  | given identifier is valid or not valid |  |  |  |  |
|  |  |  |  |  |  |
| 6. | Write a C program for construction of | 28-Feb-2020 | 15-18 |  |  |
|  | NFA from Regular Expression |  |  |  |  |
|  |  |  |  |  |  |
| 7. | Write a C program for construction of | 28-Feb-2020 | 19-22 |  |  |
|  | DFA from NFA |  |  |  |  |
|  |  |  |  |  |  |
| 8. | Write a C program to implement Shift | 21-Mar-2020 | 23-26 |  |  |
|  | Reduce Parsing algorithm |  |  |  |  |
|  |  |  |  |  |  |
| 9. | Write a C program to implement | 21-Mar-2020 | 27-28 |  |  |
|  | Operator Precedence Parser |  |  |  |  |
|  |  |  |  |  |  |
| 10. | Write a C program to implement | 21-Mar-2020 | 29-30 |  |  |
|  | Recursive Descent Parser |  |  |  |  |
|  |  |  |  |  |  |
| 11. | Write a C program for implementing | 21-Mar-2020 | 31-32 |  |  |
|  | the functionalities of predictive parser |  |  |  |  |
|  |  |  |  |  |  |
| 12. | Write a C program for constructing of | 21-Mar-2020 | 33-35 |  |  |
|  | LL(1) parsing |  |  |  |  |
|  |  |  |  |  |  |
| 13. | Write a program to design LALR | 21-Mar-2020 | 36-41 |  |  |
|  | Bottom up Parser |  |  |  |  |
|  |  |  |  |  |  |
| 14. | Write a C program to implement Code | 04-April-2020 | 42-45 |  |  |
|  | Optimization Techniques |  |  |  |  |
|  |  |  |  |  |  |
| 15. | Write a C program to implement Code | 11-April-2020 | 46-47 |  |  |
|  | Generator |  |  |  |  |
|  |  |  |  |  |  |

**WAP for implementation of LEXICAL ANALYZER for IF statement.**

#include<stdio.h>

#include<ctype.h>

#include<conio.h>

#include<string.h>charvars[100][100];i

ntvcnt;

charinput[1000],c;chartoken[50],tlen;

intstate=0,pos=0,i=0,id;

char\*getAddress(charstr[])

{

for(i=0;i<vcnt;i++)if(strcmp(str,vars[i])==0

)returnvars[i];strcpy(vars[vcnt],str);return

vars[vcnt++];

}

intisrelop(charc)

{

if(c=='>'||c=='<'||c=='|'||c=='=')return1;

elsereturn0;

}

intmain(void)

{

clrscr();

printf("EntertheInputString:");gets(input);

do

{

c=input[pos];putchar(c);switch(state)

{

case0:if(c=='i')state=1;break;

case1:if(c=='f')

{

printf("\t<1,1>\n");state=2;

}

break;case2:

if(isspace(c))printf("\b");if(isalpha(c)

)

{

token[0]=c;tlen=1;state=3;

}

if(isdigit(c))state=4;

if(isrelop(c))state=5;

if(c==';')printf("\t<4,4>\n");

if(c=='(')printf("\t<5,0>\n");

if(c==')')printf("\t<5,1>\n");

if(c=='{')printf("\t<6,1>\n");

if(c=='}')printf("\t<6,2>\n");break;

case3:

if(!isalnum(c))

{

token[tlen]='\o';printf("\b\t<2,%p>\n",getAddress(token));

state=2;

pos--;

1

NATASHA SHARMA

1703013043

**WAP for implementation of LEXICAL ANALYZER for IF statement.**

}

elsetoken[tlen++]=c;break;

case4:if(!isdigit(c))

{

printf("\b\t<3,%p>\n",&input[pos]);state=2;

pos--;

}

break;case5:

id=input[pos-1];if(c=='=')

printf("\t<%d,%d>\n",id\*10,id\*10);else

{

printf("\b\t<%d,%d>\n",id,id);pos--;

}

state=2;break;

}

pos++;

}

while(c!=0);

getch();

return0;

}

2

NATASHA SHARMA

1703013043

**Output of LEXICAL ANALYZER for IF statement.**



3

NATASHA SHARMA

1703013043

**WAP for implementation of LEXICAL ANALYZER for Arithmetic Expression.**

#include<stdio.h>

#include<ctype.h>

#include<conio.h>

#include<string.h>

char vars[100][100];

int vcnt;

char input[1000],c;

char token[50],tlen;

int state=0,pos=0,i=0,id;

char \*getAddress(char str[]){

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

if(strcmp(str,vars[i])==0)

return vars[i];

strcpy(vars[vcnt],str);

return vars[vcnt++];

}

int isrelop(char c){

if(c=='+'||c=='-'||c=='\*'||c=='/'||c=='%'||c=='^') return 1;

else return 0;

}

int main(void)

{

clrscr();

printf("Enter the Input String:");

gets(input);

do

{

c=input[pos];

putchar(c);

switch(state)

{

case 0: if(isspace(c))

printf("\b");

if(isalpha(c))

{

token[0]=c;

tlen=1;

state=1;

}

if(isdigit(c))

state=2;

if(isrelop(c))

NATASHA SHARMA 4

1703013043

**WAP for implementation of LEXICAL ANALYZER for Arithmetic Expression.**

state=3;

if(c==';')

printf("\t<3,3>\n");

if(c=='=')

printf("\t<4,4>\n");

break;

case 1: if(!isalnum(c))

{

token[tlen]='\o';

printf("\b\t<1,%p>\n",getAddress(token));

state=0;

pos--;

}

else token[tlen++]=c;

break;

case 2: if(!isdigit(c))

{

printf("\b\t<2,%p>\n",&input[pos]);

state=0;

pos--;

}

break;

case 3:

id=input[pos-1];

if(c=='=')

printf("\t<%d,%d>\n",id\*10,id\*10);

else{

printf("\b\t<%d,%d>\n",id,id);

pos--;

}

state=0;

break;

}

pos++;

}

while(c!=0);

getch();

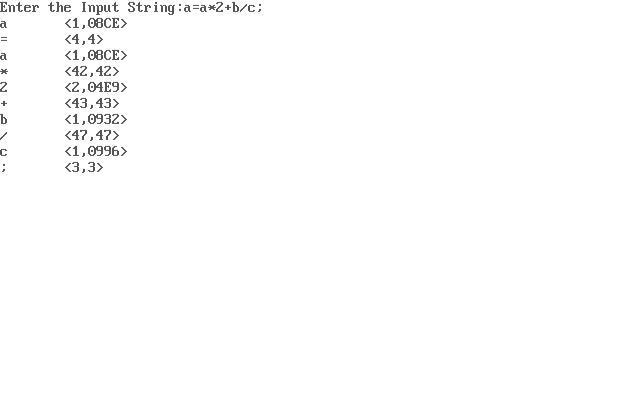
return 0;

}

NATASHA SHARMA 5

1703013043

**OUTPUT for implementation of LEXICAL ANALYZER for Arithmetic Expression.**



NATASHA SHARMA 6

1703013043