NAME: Aditya Somani

PRN: 71901204L ROLL:

T1851061

SL-5(Group'A')

Assignment no. 5

Write a program to implement a Recursive Descent Parser.

Recursive Descent Parser:

It is a kind of Top-Down Parser. A top-down parser builds the parse tree from the top to down,

starting with the start non-terminal. A Predictive Parser is a special case of Recursive Descent

Parser, where no Back Tracking is required.

By carefully writing a grammar means eliminating left recursion and left factoring from it, the

resulting grammar will be a grammar that can be parsed by a recursive descent parser.

CODE:

```
#include <stdio.h>

#include <conio.h>

#include <string.h>

char input[100];

int i, l;

void main()
```

```
{
// clrscr();
printf("\nRecursive descent parsing for the following grammar\n");
printf("\nE->TE'\n E'->+TE' |@\n T->FT'\n T'->*FT'|@\n F->(E)|ID\n");
printf("\nEnter the string to be checked:");
gets(input);
if (E())
{
if (input[i + 1] == '\0')
printf("\nString is accepted");

else
printf("\nString is not accepted");
}
else
printf("\nString not accepted");
getch();
}
int E()
{
if (T())
{
if (EP())
return (1);
```

```
else
return (0);
}
else
return (0);
}

int EP()
{
if (input[i] == '+')
{
i++;
if (T())
{
if (EP())
return (1);
else
return (0);
}
else
return (0);
}
else
return (1);
```

```
}

int T()
{
if (F())
{
if (TP())
return (1);
else
return (0);
}
else
return (0);
}

int TP()
{
if (input[i] == '*')
{
i++;
if (F())
{
if (TP())
return (1);
```

```
else
return (0);
}
else
return (0);
}
else
return (1);
}
int F()
{
if (input[i] == '(')
{
i++;
if (E())
{
if (input[i] == ')')
{
i++;
return (1);
}
else
return (0);
}
```

```
else
return (0);
}
else if (input[i] >= 'a' && input[i] <= 'z' || input[i] >= 'A' && input[i] <= 'Z')
{
i++;

return (1);
}
else
return (0);
}
```

OUTPUT:-

Recursive descent parsing for the following grammar

E->TE'

E'->+TE' |@

T->FT'

T'->*FT'|@

F->(E)|ID

```
/usr/include/stdio.h:638:14: note: declared here
main.c:13:5: warning: implicit declaration of function 'E' [-Wimplicit-function-declaration]
main.c:26:5: warning: implicit declaration of function 'T' [-Wimplicit-function-declaration]
main.c:28:5: warning: implicit declaration of function 'EP' [-Wimplicit-function-declaration]
main.c:57:5: warning: implicit declaration of function 'F' [-Wimplicit-function-declaration]
main.c:59:5: warning: implicit declaration of function 'TP' [-Wimplicit-function-declaration]
main.c:(.text+0x2d): warning: the `gets' function is dangerous and should not be used.

Recursive descent parsing for the following grammar

E->TE'
 E'->+TE' |@
 T->FT'
 T'->*FT'|@
 F->(E)|ID

Enter the string to be checked:A+B*C

String is accepted

...Program finished with exit code 255
Press ENTER to exit console.
```

```
main.c:26:5: warning: implicit declaration of function 'T' [-Wimplicit-function-declaration]
main.c:28:5: warning: implicit declaration of function 'EP' [-Wimplicit-function-declaration]
main.c:57:5: warning: implicit declaration of function 'F' [-Wimplicit-function-declaration]
main.c:59:5: warning: implicit declaration of function 'TP' [-Wimplicit-function-declaration]
main.c:(.text+0x2d): warning: the `gets' function is dangerous and should not be used.

Recursive descent parsing for the following grammar

E->TE'
 E'->+TE' |@
 T->FT'
 T'->*FT'|@
 F->(E)|ID

Enter the string to be checked:A*B*C

String is accepted
```

```
 if (F())
     ^
main.c:59:5: warning: implicit declaration of function 'TP' [-Wimplicit-function-declaration]
 if (TP())
     ^~
/tmp/ccuLaayk.o: In function `main':
main.c:(.text+0x2d): warning: the `gets' function is dangerous and should not be used.

Recursive descent parsing for the following grammar

E->TE'
 E'->+TE' |@
 T->FT'
 T'->*FT'|@
 F->(E)|ID

Enter the string to be checked:A+B/C

String is not accepted
```