Name: Subhadeep Chell
Reg.no: 21BCE1288

Compiler Design Lab – (Excercise – 3)

Q1. Ans. The required C program to construct the Recursive-Decent-Parser is given as:

```
#include<stdio.h>
#include<string.h>
int E(),Edash(),T(),Tdash(),F();
char *ip;
char string[50];
int main()
{
printf("Enter the string\n");
scanf("%s",string);
ip=string;
printf("\n\nInput\tAction\n--------------------------------\n");

if(E() && ip=="\0"){
printf("\n--------------------------------\n");
printf("\n String is successfully parsed\n");
}
else{
printf("\n--------------------------------\n");
printf("Error in parsing String\n");
}
}
int E()
{
printf("%s\tE->TE' \n",ip);
if(T())
{
if(Edash())
{
return 1;
}
else
return 0;
}
else
return 0;
}
int Edash()
{
if(*ip=='+')
{
printf("%s\tE'->+TE' \n",ip);
ip++;
if(T())
```

```
{
if(Edash())
{
return 1;
}
else
return 0;
}
else
return 0;
}
else
{
printf("%s\tE'->^ \n",ip);
return 1;
}
}
int T()
{
printf("%s\tT->FT' \n",ip);
if(F())
{

if(Tdash())
{
return 1;
}
else
return 0;
}
else
return 0;
}
int Tdash()
{
if(*ip=='*')
{
printf("%s\tT'->*FT' \n",ip);
ip++;
if(F())
{
if(Tdash())
{
return 1;
}
else
return 0;
}
else
return 0;
}
else
```

```
{
printf("%s\tT'->^ \n",ip);
return 1;
}
}
int F()
{
if(*ip=='(')
{
printf("%s\tF->(E) \n",ip);
ip++;
if(E())
{
if(*ip==')')
{
ip++;
return 0;
}
else
return 0;
}
else
return 0;
}

else if(*ip=='i')
{
ip++;
printf("%s\tF->id \n",ip);
return 1;
}
else
return 0;
}
```

OUTPUT:

```
Enter the string


Input    Action
--------------------------------
    E->TE'
    T->FT'

--------------------------------
Error in parsing String
```

Testing the program with given strings:

1. a+b*c+d

code:

```
parser = RDParser("a+b*c+d");
printf(parser.parse());
```

output:

```
('+', ('+', 'a', ('*', 'b', 'c')), 'd')
```

---

2. a*c+*d

code:

```
parser = RDParser("a*c+*d");
try{
   printf(parser.parse());
}
 except Exception{
        printf(Exception);
}
```

output:

```
Unexpected character: *
```

3.)a+b*c

code:

```
parser = RDParser(")a+b*c");
try{
        printf(parser.parse());
}

except Exception{
        printf(Exception);
}
```

output:

```
Expected )
```

---

4. ((a+b)*c

code:

```
parser = RDParser("((a+b)*c");
try{
    printf(parser.parse());
}
except Exception{
    printf(Exception);
};
```

output:

Unexpected character: None

Q2. Ans. The required C program to construct the required Recursive-Decent-Parser is given as:

```
#include <stdio.h>
#include <string.h>
#include <ctype.h>

int pos;
char *input;
int input_len;

void E();
void T();
void F();

void E() {
  T();
  while (pos < input_len && (input[pos] == '#')) {
    pos++;
    T();
  }
}

void T() {
  F();
  while (pos < input_len && (input[pos] == '&')) {
    pos++;
    F();
  }
}

void F() {
  if (pos >= input_len) {
    printf("Unexpected end of input\n");
```

```
    exit(1);
  }
  char c = input[pos];
  pos++;
  if (c == '!') {
    F();
  } else if (c == '(') {
    E();
    if (input[pos] != ')') {
      printf("Expected )\n");
      exit(1);
    }
    pos++;
  } else if (isalpha(c)) {
    return;
  } else {
    printf("Unexpected character: %c\n", c);
    exit(1);
  }
}

int main(int argc, char **argv) {
  if (argc < 2) {
    printf("Usage: %s <expression>\n", argv[0]);
    exit(1);
  }
  input = argv[1];
  input_len = strlen(input);
  pos = 0;
  E();
  if (pos != input_len) {
    printf("Unexpected characters at the end of the input\n");
    exit(1);
  }
  printf("The expression is valid\n");
  return 0;
}
```

output:

The required output for the given strings can be generated and tested via the following teminal commands :

1. ./parser a#b&!c
2. ./parser a&#b
3. ./parser a#b&!c)
4. ./parser (a#b)&c)
