LAB10: Make recursive descendent parser .

IT057-Kathiriya Darshakkumar

Grammer:

P -> R P'

P'-> + R P' | - R P' | P

R -> Q R'

R'-> * Q R' | / Q R' | P

Q-> (P) | id

Valid String: d , d+d,d+d*d,d*d*d….

InValid String:ddd+d,*d,/d/….

Code:

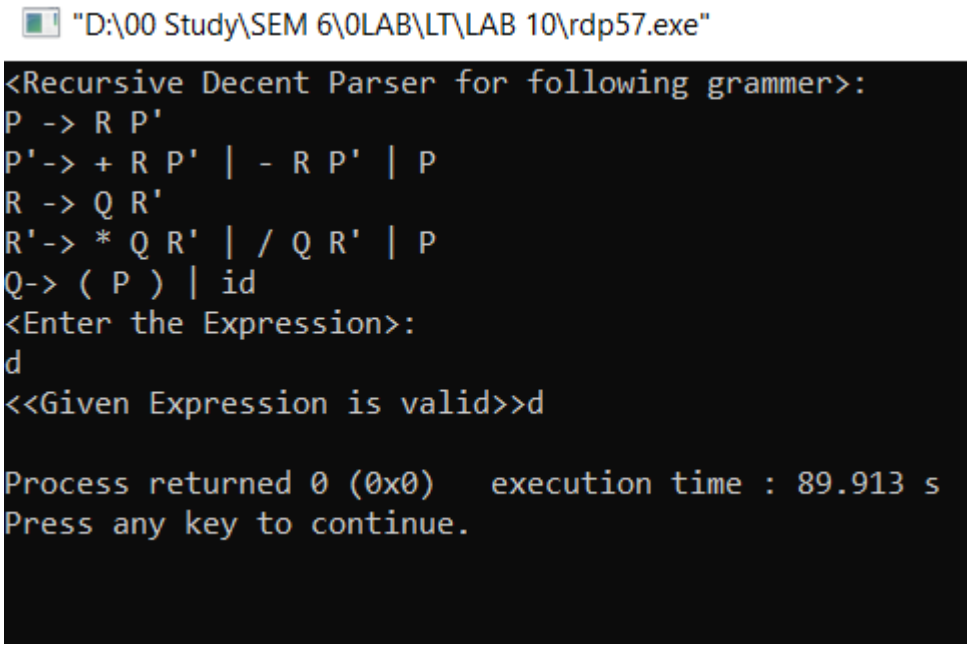
```
#include<bits/stdc++.h>
#include<ctype.h>
using namespace std;
int cnt,flag;
char expr[100];
void P();
void Q();
void Pdash();
void Rdash();
void R();
int main()
{
   cnt=0,flag=0;
   cout<<"<Recursive Decent Parser for following grammer>:"<<endl;
   cout<<"P -> R P'\nE'-> + R P' | - R P' | P\nT -> Q R'\nT'-> * Q R' | / Q R' | P\nF-> ( P ) | id"<<endl;
   cout<<"<Enter the Expression>: "<<endl;
```

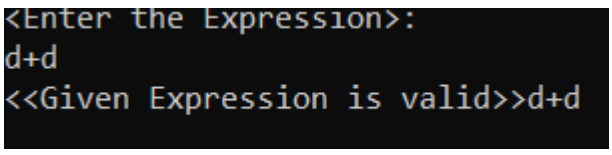
```
    cin>>expr;
    P();
    if((strlen(expr)==cnt)&&(flag==0))
    {
       cout<<"<<Given Expression is valid>>"<<expr<<endl;
    }
    else
    {
       cout<<"<<Given Expression is invalid>>"<<expr<<endl;
    }
}
void P()
{
    R();
    Rdash();
}
void R()
{
    Q();
    Pdash();
}
void Q()
{
    if(expr[cnt]=='d')
    {
       cnt++;

    }
    else if(expr[cnt]=='(')

    {
```

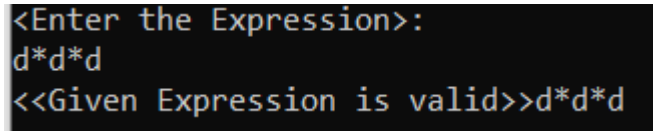
```
        cnt++;
        P();
        if(expr[cnt]==')')

        {
           cnt++;

        }
        else

        {
           flag=1;
        }

    }
    else

    {
        flag=1;

    }
}
void Rdash()
{
    if(expr[cnt]=='+'||expr[cnt]=='-')
    {
        cnt++;
        R();
        Rdash();
    }
```

```
}
void Pdash()
{
   if(expr[cnt]=='*'||expr[cnt]=='/')
   {
      cnt++;
      Q();
      Pdash();
   }
}
```

Outputs:







```
<Enter the Expression>:
d+d*d
<<Given Expression is valid>>d+d*d
```

```
<Enter the Expression>:
**d**
<<Given Expression is invalid>>**d**
```

```
<Enter the Expression>:
ddd*d*d
<<Given Expression is invalid>>ddd*d*d
```

```
<Enter the Expression>:
d/d/
<<Given Expression is invalid>>d/d/
```