2] Recursive Descent parser:- (Top Down Parser)

In recursive descent parser for every variable or Non-terminal we are going to write a function.

ex. $E \rightarrow iE'$

$E' \rightarrow +iE' / \epsilon$

→ Function for variable E

```
E()
{
01  if(l=='i') /* l= lookahed is a global
                   variable */
02  {
03    match('i');
04    E'();
05  }
06 }
```

Function for variable E'

```
E'()
{
01  if(l=='+')
02  {
03    match('+');
04    match('i')
05    E'();
06  }
07  else
08    return; //is for E'→ε i.e nothing
09 }
```

line No

```
        match(char t)
          {
            if(l==t)
              {
                l=getchar(); //increment lookah
              }                          pointer
            else
              {
              printf("ee ERROR");
              }
          }

main()
  {
01    E();
02    if(l=='$')
03      {
04        printf("ee parsing suecessful");
05      }
```

i/p → i+i$

lookahead is pointing to first i

```
i+i$
↑
```

First function get called is main() . then E()
stack used by os.

main()	E()	E'()	E'()
return line NO 02	return line No or address =06	return address line No 09	$\lambda = \$$ then return back