Exercise 3: Eliminating Left Recursion

Ram Kaushik R

February 3, 2020

Assignment	3
Reg No	312217104125
Name	Ram Kaushik R

1 Program

```
#include<stdio.h>
#include<string.h>
int main() {
  char non_terminal, productions[10][100], splits[10][10];
  int num;
  printf("Enter number of productions: ");
  scanf("%d", &num);
  printf("Enter the grammar:\n");
  for(int i = 0; i < num; i++)
    scanf("%s", productions[i]);
  for(int i = 0; i < num; i++) {
    printf("\n%s", productions[i]);
    non_terminal = productions[i][0];
    char production[100], *token;
    int j, flag = 0;
    for(j = 0; productions[i][j + 3] != '\0'; j++)
      production[j] = productions[i][j + 3];
    production[j] = '\0';
    j = 0;
    token = strtok(production, "|");
    while(token != NULL) {
      strcpy(splits[j], token);
      if(token[0] == non_terminal && flag == 0)
flag = 1;
      else if(token[0] != non_terminal && flag == 1)
flag = 2;
      j++;
      token = strtok(NULL, "|");
    }
```

```
    if(flag == 0)
      printf(" is not left recursive.\n");
    else if(flag == 1)
      printf(" is left recursive, cannot reduce.\n");
    else {
      printf(" is left recursive. After elimination:\n");
      flag = 0;
      for(int k = 0; k < j; k++) {
if(splits[k][0] != non_terminal) {
  if(flag != 0)
    printf("|%s%c\'", splits[k], non_terminal);
  else {
    flag = 1;
    printf("%c->%s%c\'", non_terminal,
            splits[k], non_terminal);
  }
}
      }
      printf("\n");
      flag = 0;
      for(int k = 0; k < j; k++) {
if(splits[k][0] == non_terminal) {
  if(flag != 0)
    printf("|%s%c\'", splits[k] + 1, non_terminal);
  else {
    flag = 1;
    printf("%c\'->%s%c\'", non_terminal,
            splits[k] + 1, non_terminal);
  }
}
      }
      printf("|e\n");
    }
  }
}
```

2 Input

```
E->E+T|T
T->T*F|F
F->id|(E)
```

3 Output

```
E->E+T|T is left recursive. After elimination:
```

E->TE’
E’->+TE’|e

T->T*F|F is left recursive. After elimination:
T->FT’
T’->*FT’|e

F->id|(E) is not left recursive.