Ayush Goyal
190905522 CSE D Roll 62

Compiler Design Lab 6(Week 5): Recursive Descent Parser

Write a recursive descent parser for the following simple grammars:

1.   **S -> a | > | (T)**
       T -> T, S|S

Code:

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

int curr = 0;
char str[100];
void S();
void T();
void tPrime();

void printValid(){
        printf("\n *** Sucessful! ***\n");
        exit(0);
}

void printInvalid(){
        printf("\n *** Unsucessful! ***\n");
        exit(0);
}

void S(){
        if(str[curr] == 'a' || str[curr] == '>'){
                curr++;
                return;
        }
        else if(str[curr] == '('){
                curr++;
                T();
                if(str[curr] == ')'){
                        curr++;
                        return;
                }
                else
                        printInvalid();
        }
        else
                printInvalid();
}

void T(){
```

```c
        S();
        tPrime();
}

void tPrime(){
        if(str[curr] == ','){
                curr++;
                S();
                tPrime();
        }
}

void main(){
        printf("Enter the String: \n");
        scanf("%s",str);
        S();
        if(str[curr] == '$')
                printValid();
        else
                printInvalid();
}
```

Output:

2.   **S -> UVW**
      U -> (S) | aSb | d
      V -> aV | ε
      W -> cW | ε

Code:

```c
#include<stdio.h>
#include<stdlib.h>
```

```c
#include<string.h>

int curr = 0;
char str[100];
void S();
void U();
void V();
void W();

void printValid(){
        printf("\n *** Sucessful! ***\n");
        exit(0);
}

void printInvalid(){
        printf("\n *** Unsucessful! ***\n");
        exit(0);
}

void S(){
        U();
        V();
        W();
}

void U(){
        if(str[curr] == '('){
                curr++;
                S();
                if(str[curr] == ')'){
                        curr++;
                        return;
                }
                else
                        printInvalid();
        }
        else if(str[curr] == 'a'){
                curr++;
                S();
                if(str[curr] == 'b'){
                        curr++;
                        return;
                }
                else
                        printInvalid();
        }
        else if(str[curr] == 'd'){
                curr++;
                return;
        }
        else
                printInvalid();
```

```
}

void V(){
        if(str[curr] == 'a'){
                curr++;
                V();
        }
}

void W(){
        if(str[curr] == 'c'){
                curr++;
                W();
        }
}

void main(){
        printf("Enter the String: \n");
        scanf("%s",str);
        S();
        if(str[curr] == '$')
                printValid();
        else
                printInvalid();
}
```
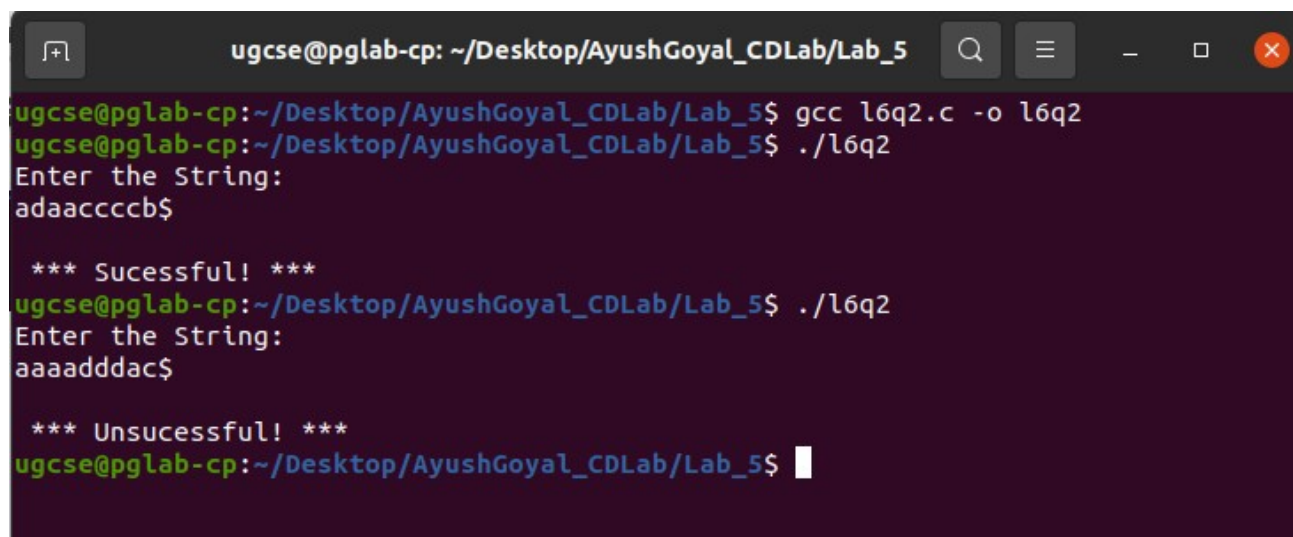
Output:

3.   S -> aAcBe
        A -> Ab|b
        B -> d

Code:

```c
#include<stdio.h>
#include<stdlib.h>
#include<string.h>

int curr = 0;
char str[100];
void S();
void A();
void B();
void aPrime();

void printValid(){
        printf("\n *** Sucessful! ***\n");
        exit(0);
}

void printInvalid(){
        printf("\n *** Unsucessful! ***\n");
        exit(0);
}

void S(){
        if(str[curr] == 'a'){
                curr++;
                A();
                if(str[curr] == 'c'){
                        curr++;
                        B();
                        if(str[curr] == 'e'){
                                curr++;
                                return;
                        }
                        else
                                printInvalid();
                }
                else
                        printInvalid();
        }
        else
                printInvalid();
}

void A(){
        if(str[curr] == 'b'){
                curr++;
                aPrime();
        }
        else
                printInvalid();
}

void aPrime(){
```

```
        if(str[curr] == 'b'){
                curr++;
                aPrime();
        }
}

void B(){
        if(str[curr] == 'd'){
                curr++;
                return;
        }
        else
                printInvalid();
}

void main(){
        printf("Enter the String: \n");
        scanf("%s",str);
        S();
        if(str[curr] == '$')
                printValid();
        else
                printInvalid();
}
```

Output:

4.   **S -> (L) | a**
        L -> L,S | S

Code:

```
#include<stdio.h>
#include<stdlib.h>
```

```c
#include<string.h>

int curr = 0;
char str[100];
void S();
void L();
void lPrime();

void printValid(){
        printf("\n *** Sucessful! ***\n");
        exit(0);
}

void printInvalid(){
        printf("\n *** Unsucessful! ***\n");
        exit(0);
}

void S(){
        if(str[curr] == 'a'){
                curr++;
                return;
        }
        else if(str[curr] == '('){
                curr++;
                L();
                if(str[curr] == ')'){
                        curr++;
                        return;
                }
                else
                        printInvalid();
        }
        else
                printInvalid();
}

void L(){
        S();
        lPrime();
}

void lPrime(){
        if(str[curr] == ','){
                curr++;
                S();
                lPrime();
        }
}

void main(){
        printf("Enter the String: \n");
```
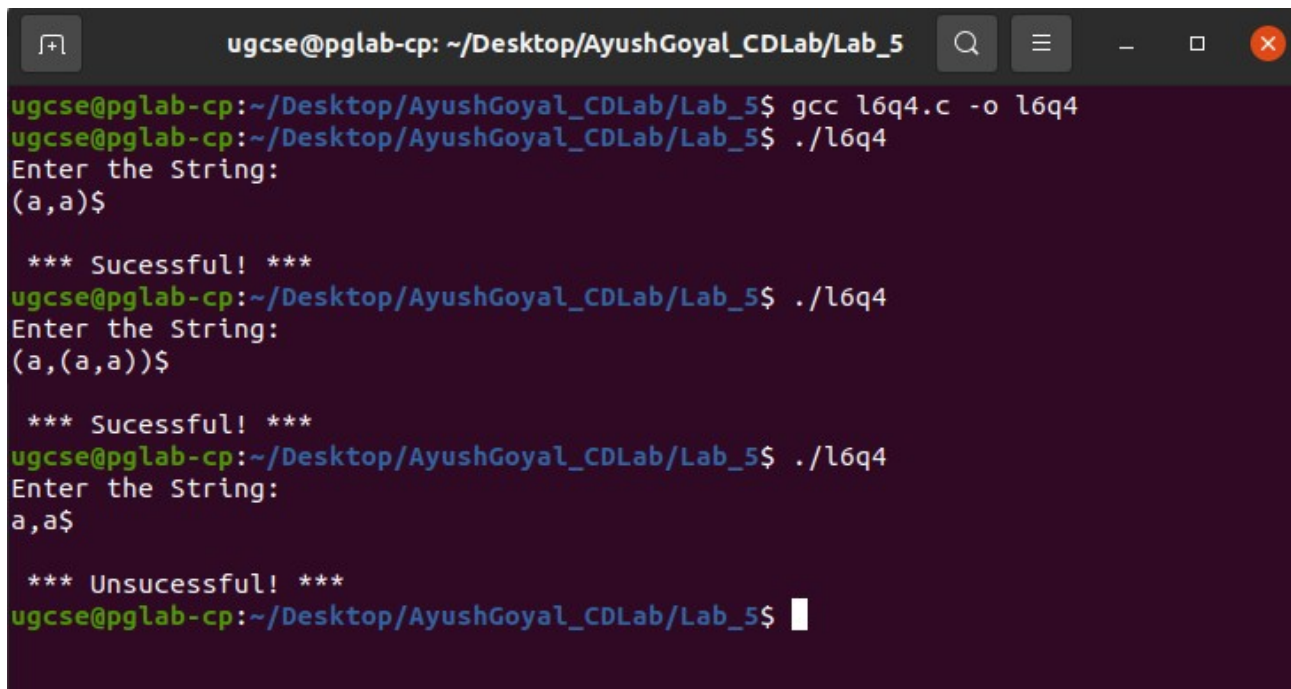
```
        scanf("%s",str);
        S();
        if(str[curr] == '$')
                printValid();
        else
                printInvalid();
}
```

Output: