NAME:AYANIKA PAUL
Roll No. 22
D1

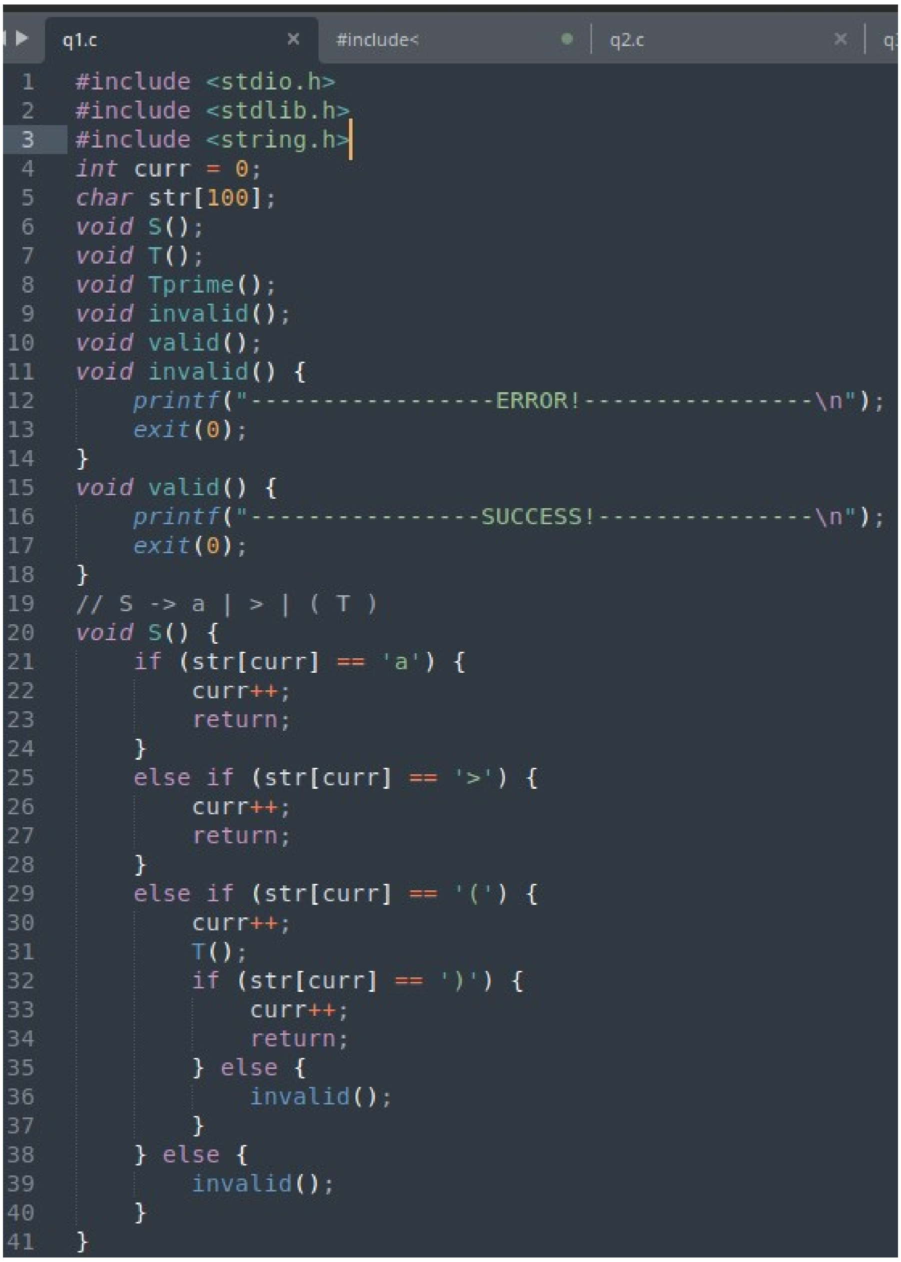
LAB 6

Write a recursive descent parser for the following simple grammars.

1. S → a | > | (T)
   T → T, S|S

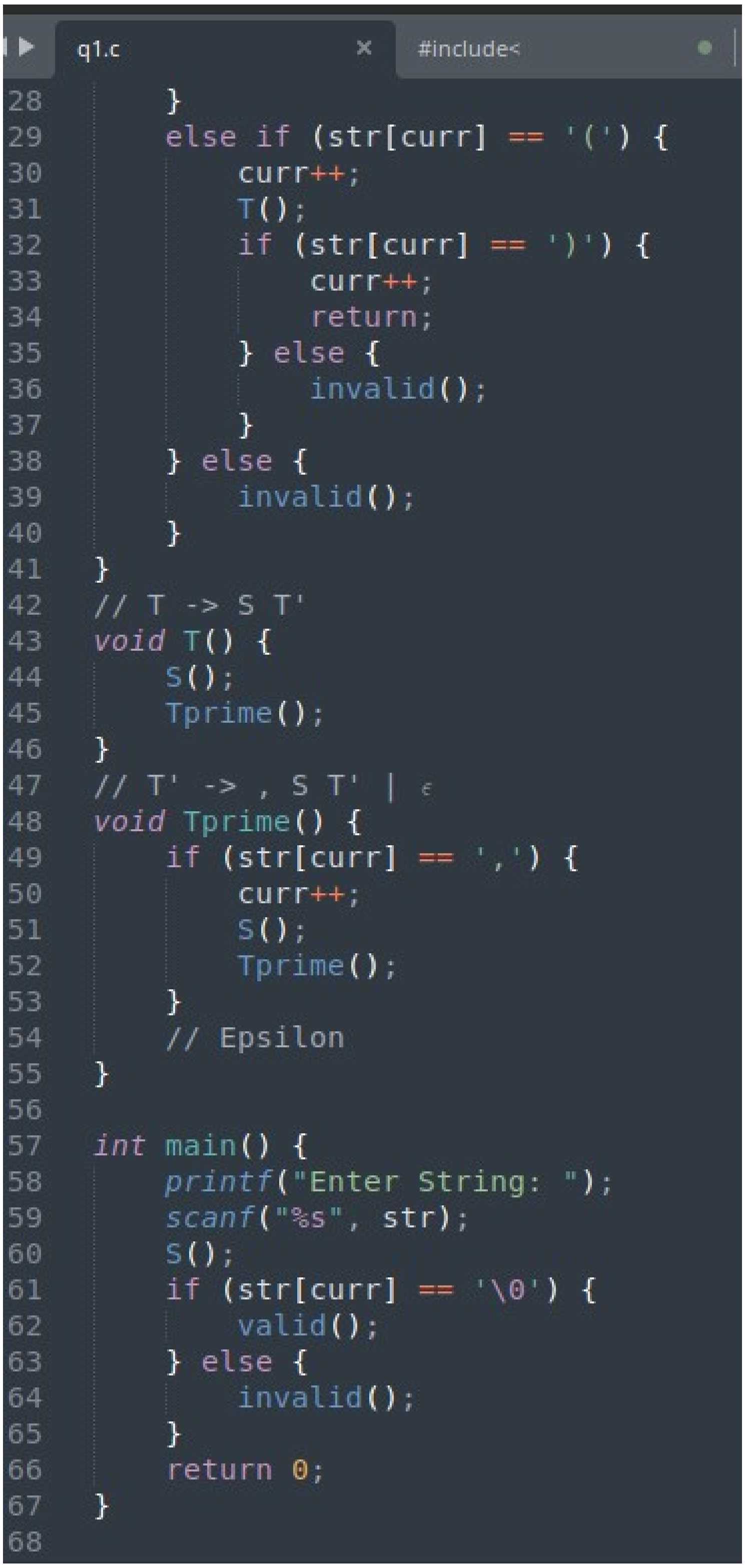
After removing Left Recursion ,
  S -> a | > | (T)
  T -> S T'
  T' -> , S T'| Epsilon

CODE:

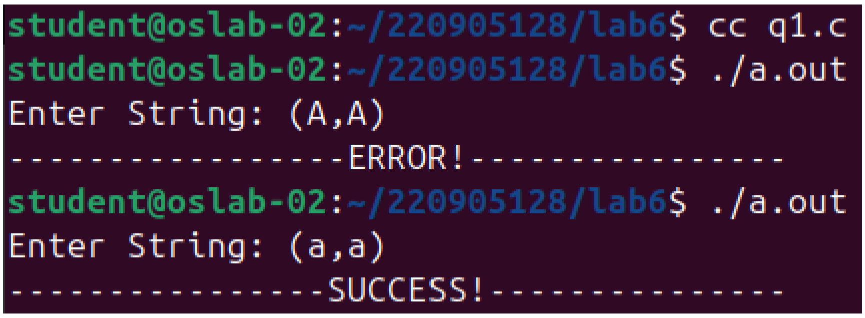


```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int curr = 0;
char str[100];
void S();
void T();
void Tprime();
void invalid();
void valid();
void invalid() {
    printf("-----------------ERROR!----------------\n");
    exit(0);
}
void valid() {
    printf("----------------SUCCESS!---------------\n");
    exit(0);
}
// S -> a | > | ( T )
void S() {
    if (str[curr] == 'a') {
        curr++;
        return;
    }
    else if (str[curr] == '>') {
        curr++;
        return;
    }
    else if (str[curr] == '(') {
        curr++;
        T();
        if (str[curr] == ')') {
            curr++;
            return;
        } else {
            invalid();
        }
    } else {
        invalid();
    }
}
```

OUTPUT:



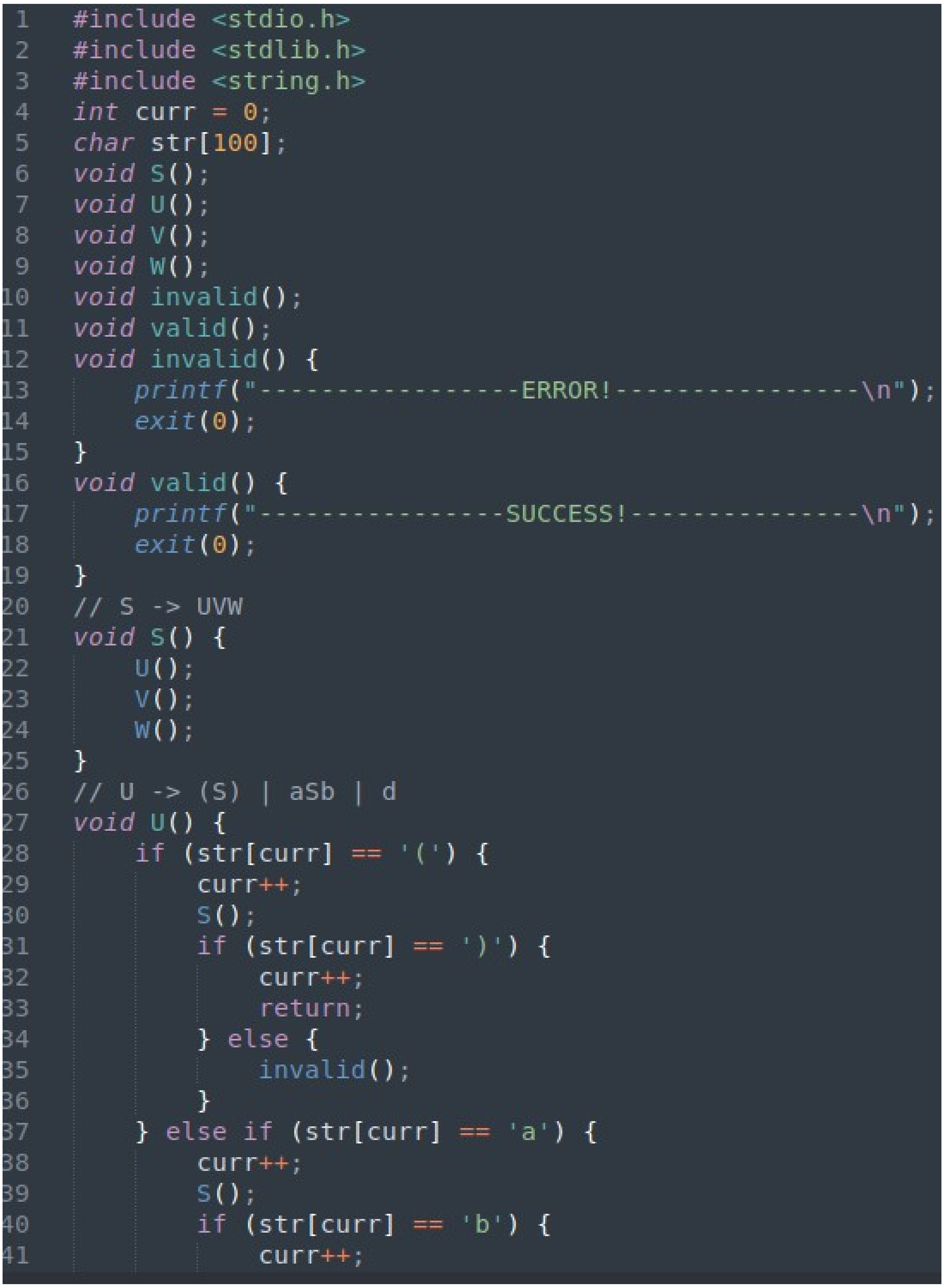
```c
        }
        else if (str[curr] == '(') {
            curr++;
            T();
            if (str[curr] == ')') {
                curr++;
                return;
            } else {
                invalid();
            }
        } else {
            invalid();
        }
    }
    // T -> S T'
    void T() {
        S();
        Tprime();
    }
    // T' -> , S T' | ϵ
    void Tprime() {
        if (str[curr] == ',') {
            curr++;
            S();
            Tprime();
        }
        // Epsilon
    }

    int main() {
        printf("Enter String: ");
        scanf("%s", str);
        S();
        if (str[curr] == '\0') {
            valid();
        } else {
            invalid();
        }
        return 0;
    }

```



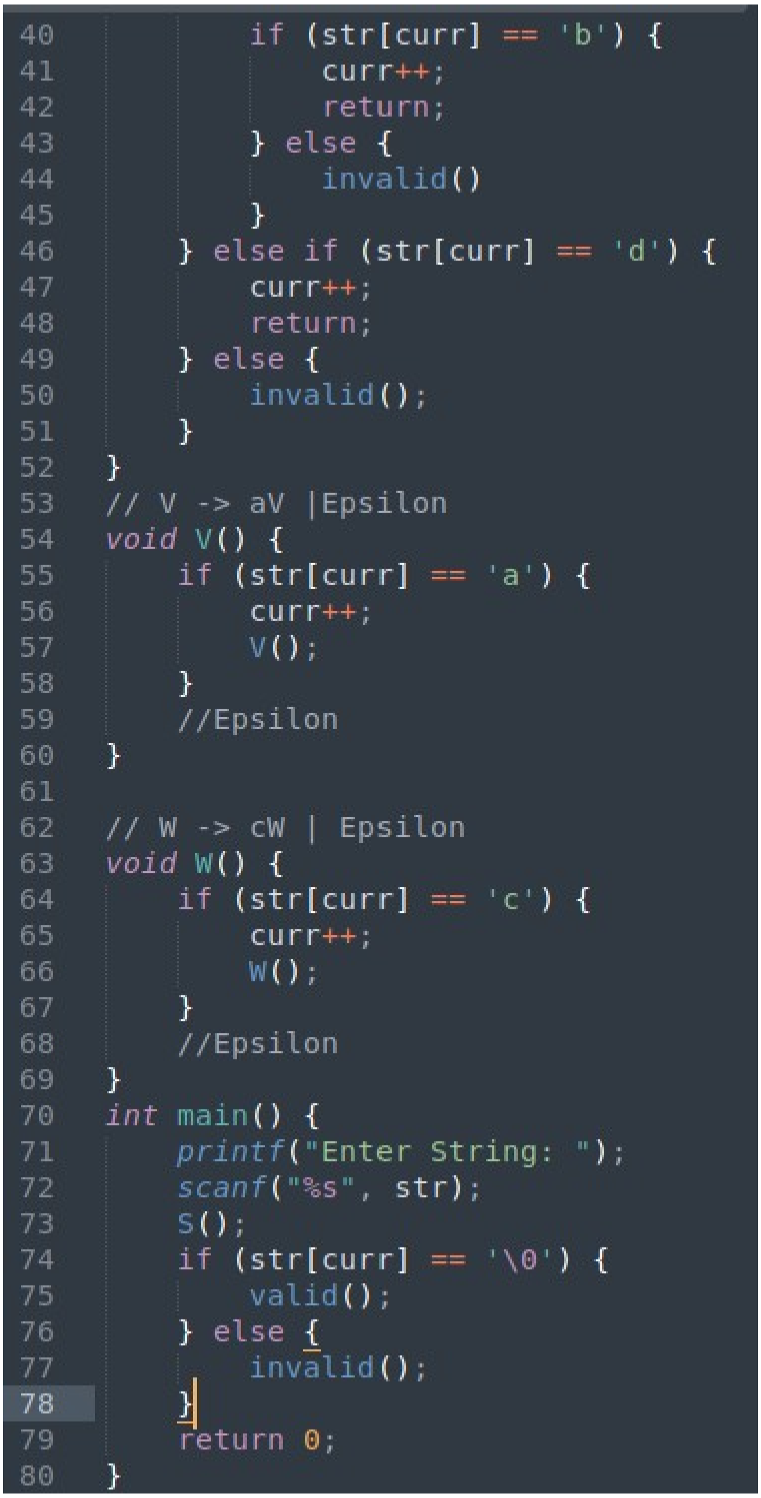
```
student@oslab-02:~/220905128/lab6$ cc q1.c
student@oslab-02:~/220905128/lab6$ ./a.out
Enter String: (A,A)
-----------------ERROR!----------------
student@oslab-02:~/220905128/lab6$ ./a.out
Enter String: (a,a)
----------------SUCCESS!---------------
```

Q2)
S→UVW
U →(S) | aSb | d
V → aV |
W → cW |

CODE:

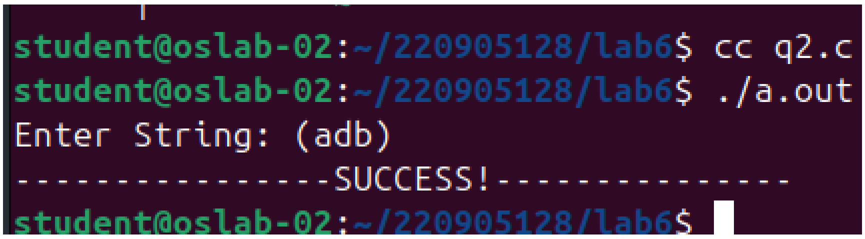


```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int curr = 0;
char str[100];
void S();
void U();
void V();
void W();
void invalid();
void valid();
void invalid() {
    printf("-----------------ERROR!----------------\n");
    exit(0);
}
void valid() {
    printf("----------------SUCCESS!---------------\n");
    exit(0);
}
// S -> UVW
void S() {
    U();
    V();
    W();
}
// U -> (S) | aSb | d
void U() {
    if (str[curr] == '(') {
        curr++;
        S();
        if (str[curr] == ')') {
            curr++;
            return;
        } else {
            invalid();
        }
    } else if (str[curr] == 'a') {
        curr++;
        S();
        if (str[curr] == 'b') {
            curr++;
```



```
        if (str[curr] == 'b') {
            curr++;
            return;
        } else {
            invalid()
        }
    } else if (str[curr] == 'd') {
        curr++;
        return;
    } else {
        invalid();
    }
}
// V -> aV |Epsilon
void V() {
    if (str[curr] == 'a') {
        curr++;
        V();
    }
    //Epsilon
}

// W -> cW | Epsilon
void W() {
    if (str[curr] == 'c') {
        curr++;
        W();
    }
    //Epsilon
}
int main() {
    printf("Enter String: ");
    scanf("%s", str);
    S();
    if (str[curr] == '\0') {
        valid();
    } else {
        invalid();
    }
    return 0;
}
```

OUTPUT:

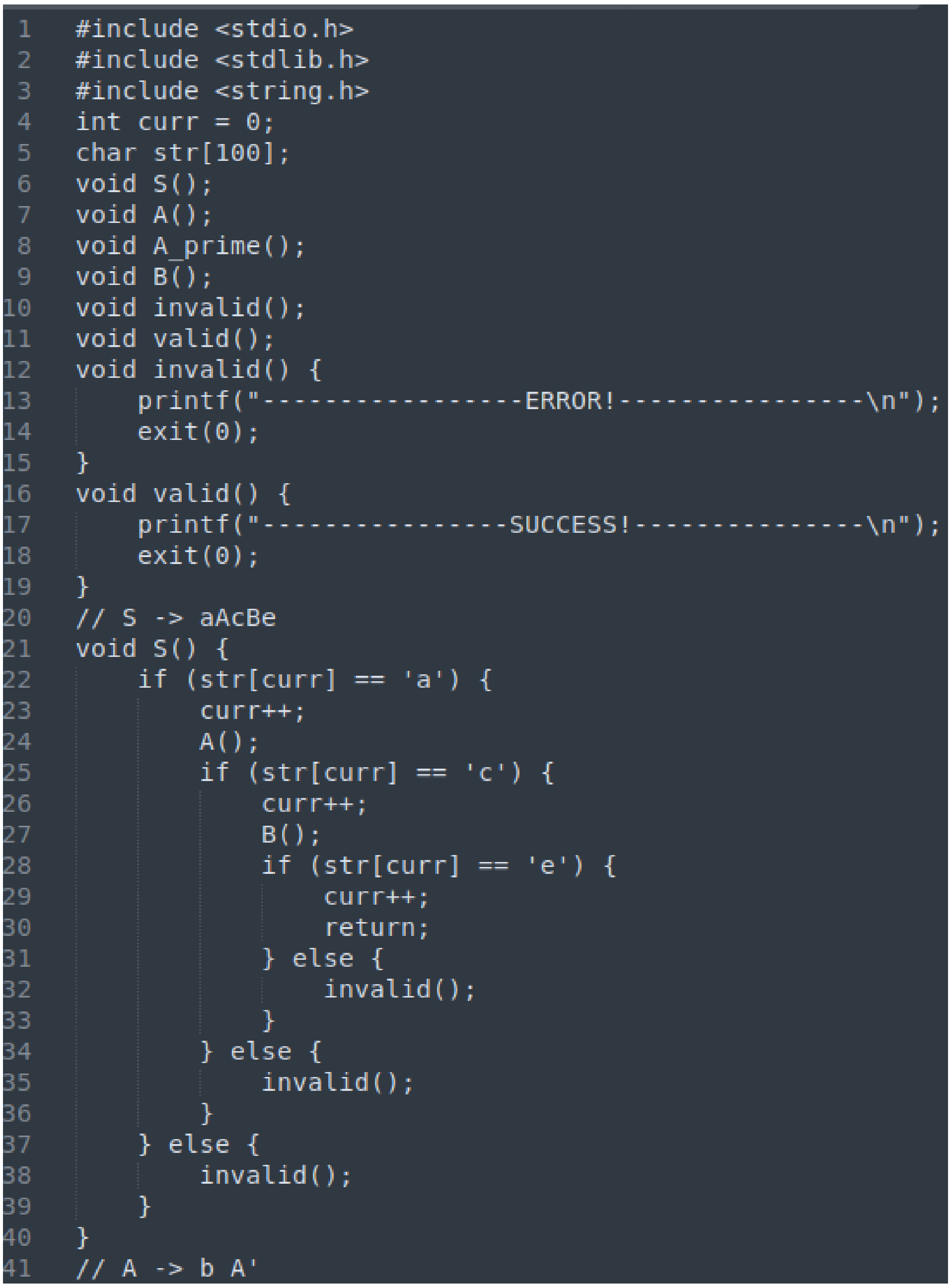


```
student@oslab-02:~/220905128/lab6$ cc q2.c
student@oslab-02:~/220905128/lab6$ ./a.out
Enter String: (adb)
----------------SUCCESS!---------------
student@oslab-02:~/220905128/lab6$
```

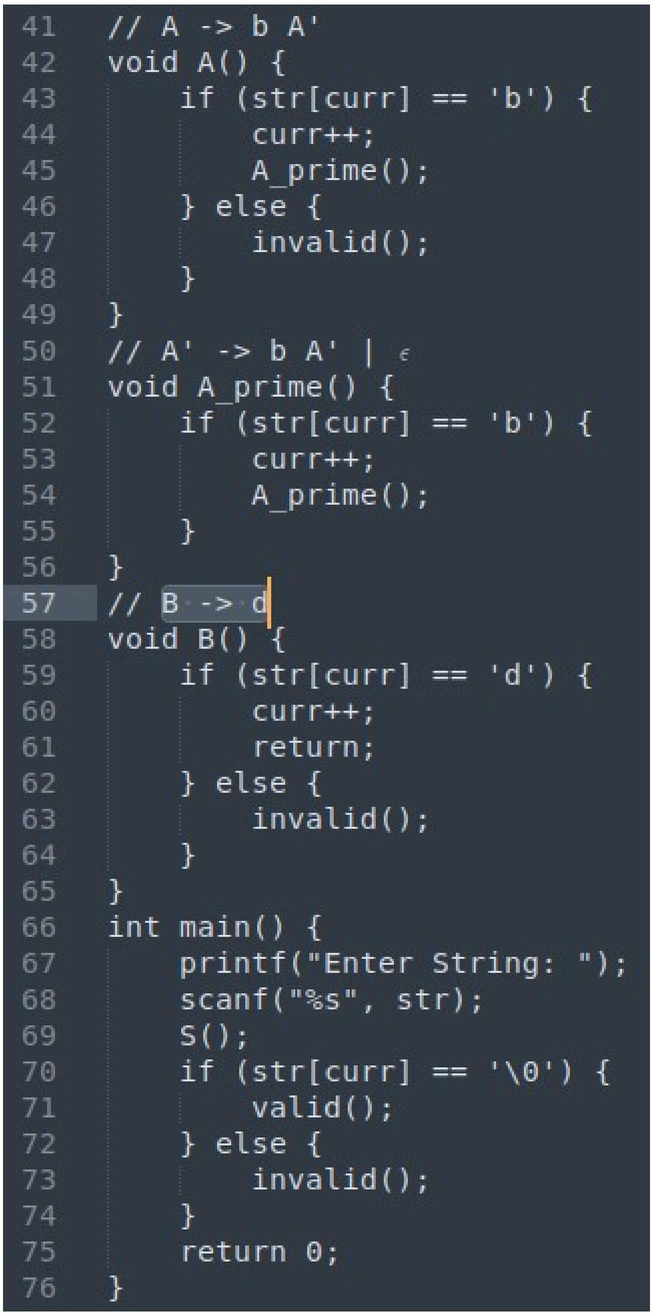
Q3)
S → aAcBe
A → Ab|b
B → d

After removing Left Recursion ,
S -> aAcBe
A -> b A'
A' -> b A' |Epsilon
B -> d

CODE:

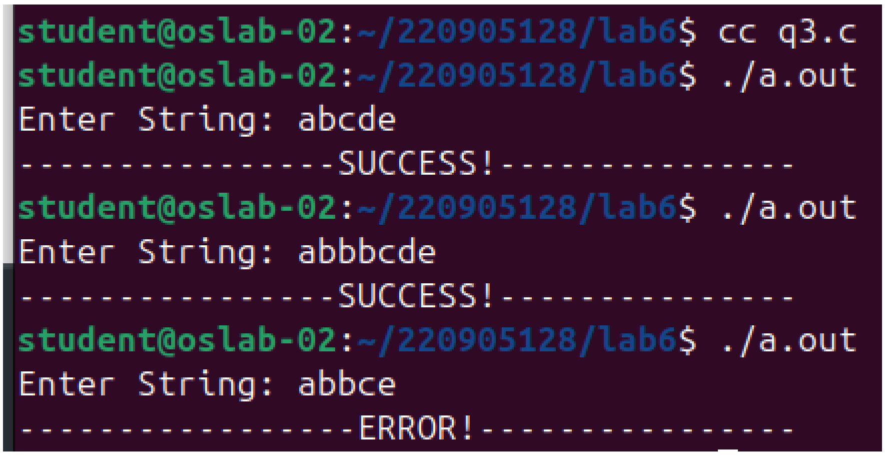


```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int curr = 0;
char str[100];
void S();
void A();
void A_prime();
void B();
void invalid();
void valid();
void invalid() {
    printf("-----------------ERROR!----------------\n");
    exit(0);
}
void valid() {
    printf("----------------SUCCESS!---------------\n");
    exit(0);
}
// S -> aAcBe
void S() {
    if (str[curr] == 'a') {
        curr++;
        A();
        if (str[curr] == 'c') {
            curr++;
            B();
            if (str[curr] == 'e') {
                curr++;
                return;
            } else {
                invalid();
            }
        } else {
            invalid();
        }
    } else {
        invalid();
    }
}
// A -> b A'
```



```c
// A -> b A'
void A() {
    if (str[curr] == 'b') {
        curr++;
        A_prime();
    } else {
        invalid();
    }
}
// A' -> b A' | ϵ
void A_prime() {
    if (str[curr] == 'b') {
        curr++;
        A_prime();
    }
}
// B -> d
void B() {
    if (str[curr] == 'd') {
        curr++;
        return;
    } else {
        invalid();
    }
}
int main() {
    printf("Enter String: ");
    scanf("%s", str);
    S();
    if (str[curr] == '\0') {
        valid();
    } else {
        invalid();
    }
    return 0;
}
```

OUTPUT:

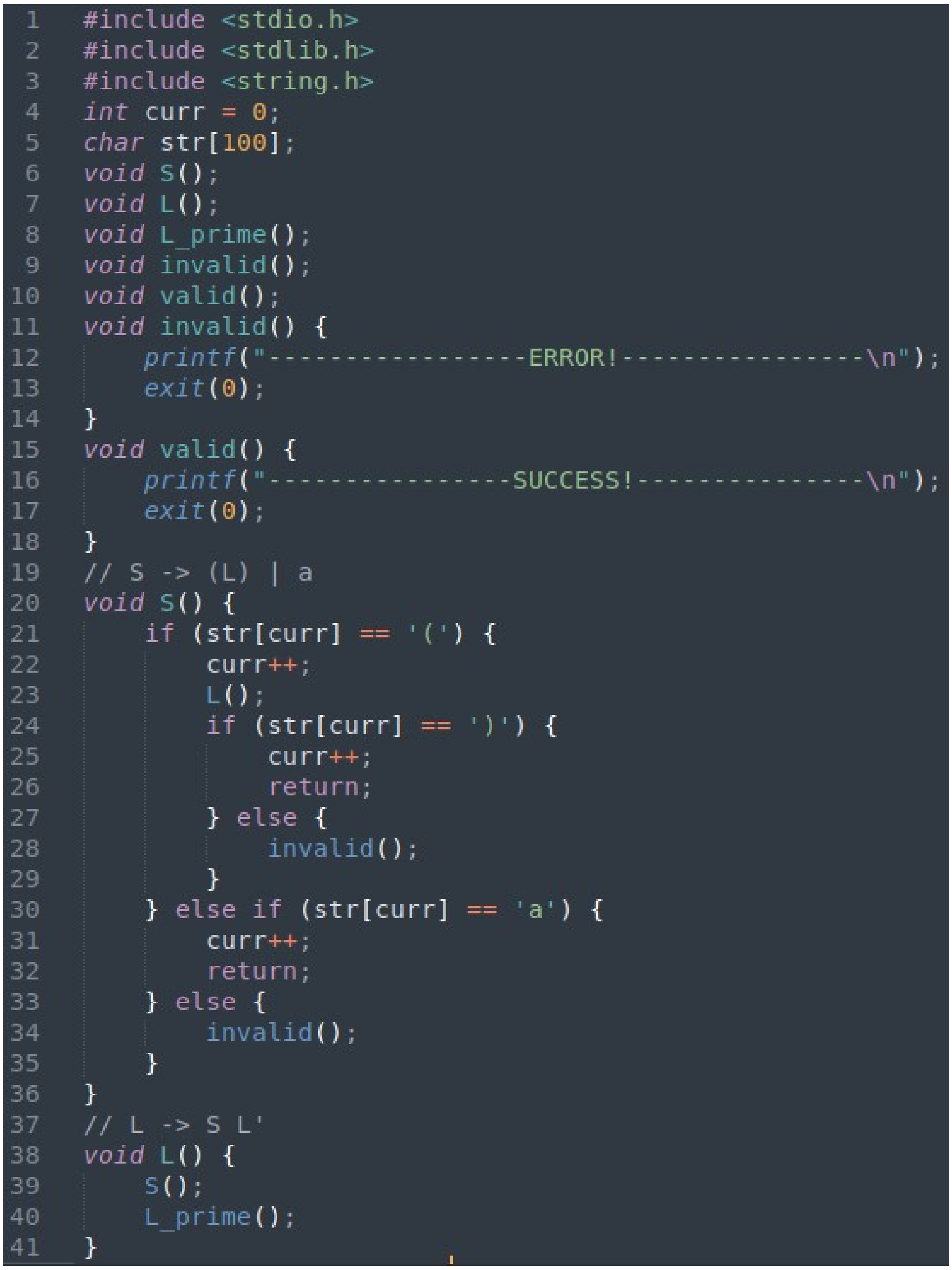


```
student@oslab-02:~/220905128/lab6$ cc q3.c
student@oslab-02:~/220905128/lab6$ ./a.out
Enter String: abcde
----------------SUCCESS!---------------
student@oslab-02:~/220905128/lab6$ ./a.out
Enter String: abbbcde
----------------SUCCESS!---------------
student@oslab-02:~/220905128/lab6$ ./a.out
Enter String: abbce
-----------------ERROR!----------------
```

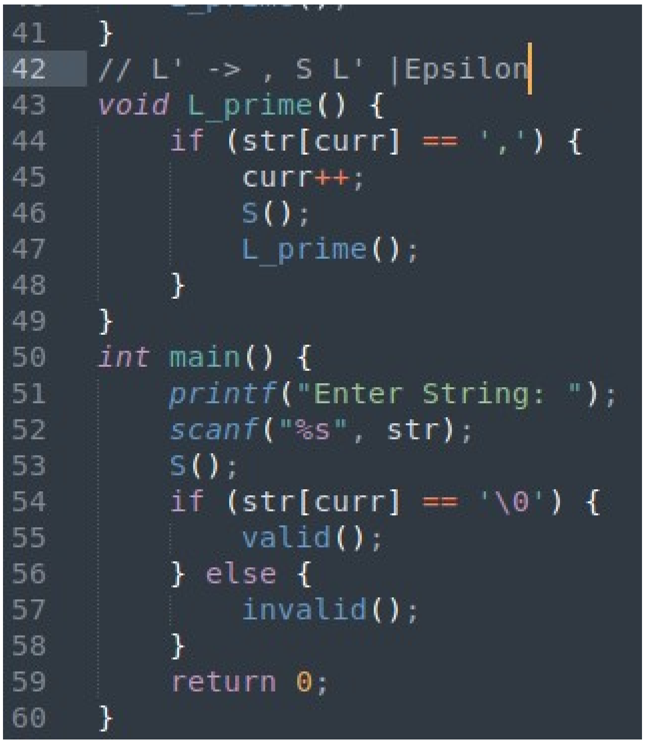
Q4)
S →(L) | a
L → L,S | S

After removing Left Recursion ,
S -> (L) | a
L -> S L'
L' -> , S L' | Epsilon

CODE:

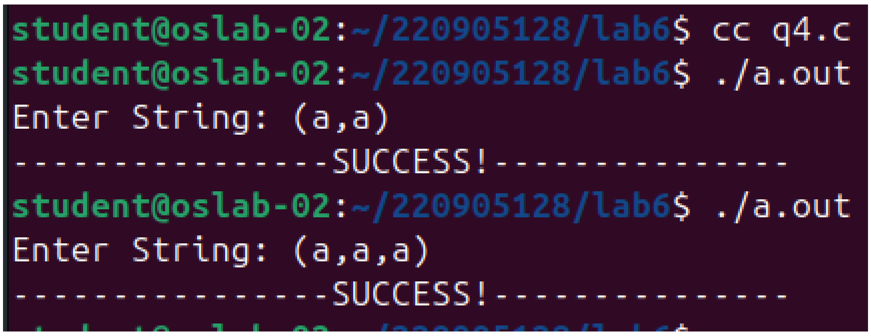


```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int curr = 0;
char str[100];
void S();
void L();
void L_prime();
void invalid();
void valid();
void invalid() {
    printf("-----------------ERROR!----------------\n");
    exit(0);
}
void valid() {
    printf("----------------SUCCESS!---------------\n");
    exit(0);
}
// S -> (L) | a
void S() {
    if (str[curr] == '(') {
        curr++;
        L();
        if (str[curr] == ')') {
            curr++;
            return;
        } else {
            invalid();
        }
    } else if (str[curr] == 'a') {
        curr++;
        return;
    } else {
        invalid();
    }
}
// L -> S L'
void L() {
    S();
    L_prime();
}
```



```c
}
// L' -> , S L' |Epsilon
void L_prime() {
    if (str[curr] == ',') {
        curr++;
        S();
        L_prime();
    }
}
int main() {
    printf("Enter String: ");
    scanf("%s", str);
    S();
    if (str[curr] == '\0') {
        valid();
    } else {
        invalid();
    }
    return 0;
}
```

OUTPUT:



```
student@oslab-02:~/220905128/lab6$ cc q4.c
student@oslab-02:~/220905128/lab6$ ./a.out
Enter String: (a,a)
----------------SUCCESS!---------------
student@oslab-02:~/220905128/lab6$ ./a.out
Enter String: (a,a,a)
----------------SUCCESS!---------------
```