Week6

Name: NEERAJ KUMAR
Regno.: 220905536
Roll no.: 57
Title: RECURSIVE DESCENT PARSER FOR SIMPLE GRAMMARS

Lab Exercise: Write a recursive descent parser for the following simple grammars.

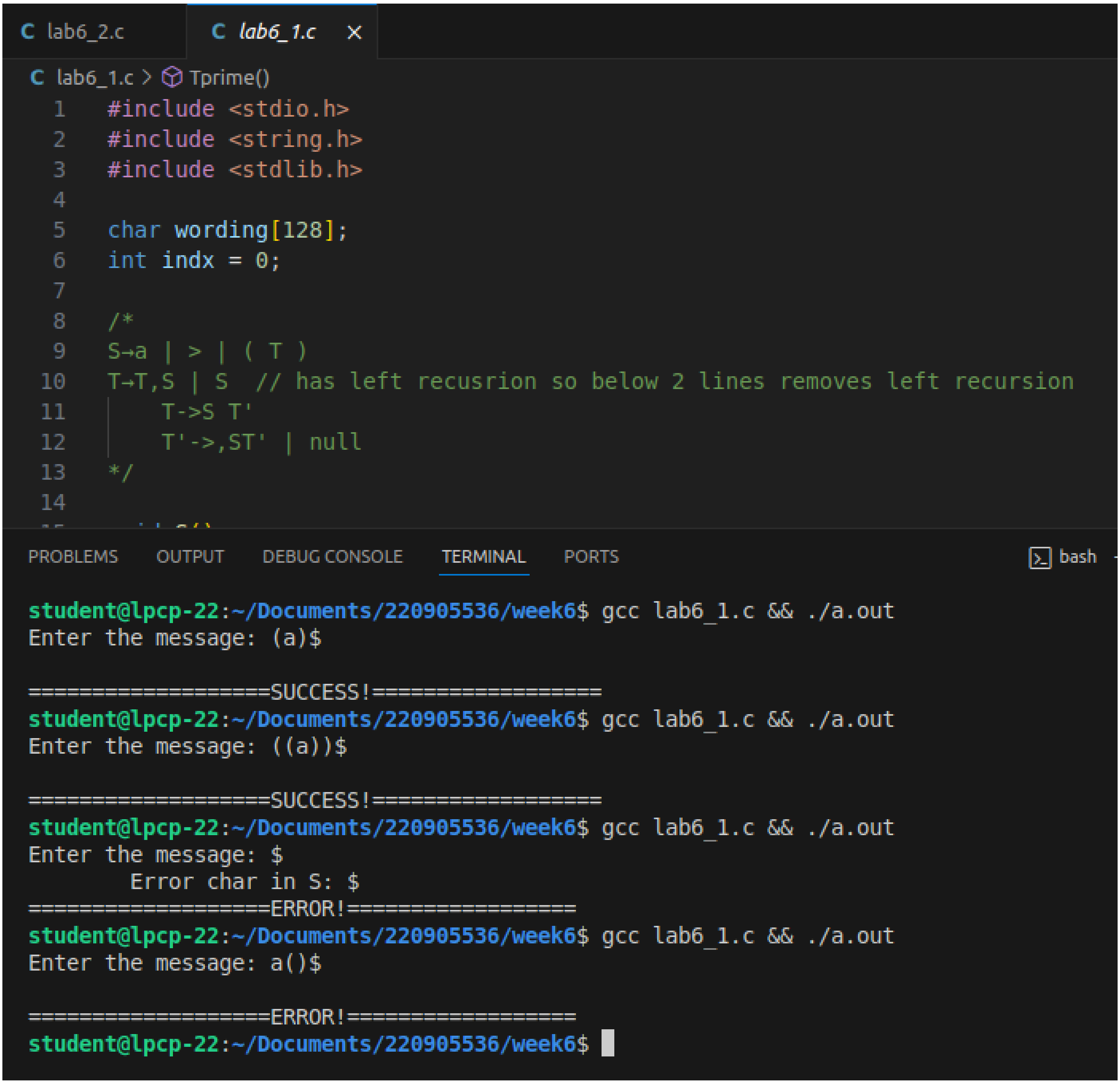
1. S→a | > | (T)
   T→T, S|S

   Source Code:

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

char wording[128];
int indx = 0;

/*
S→a | > | ( T )
T→T,S | S  // has left recusrion so below 2 lines removes left recursion
   T->S T'
   T'->,ST' | null
*/

void S();
void T();
void Tprime();

void valid()
{
   printf("\n==================SUCCESS!=================\n");
   exit(0);
}

void invalid()
{
   printf("\n==================ERROR!=================\n");
   exit(0);
}

void Tprime()
{
   if (wording[indx] == ',')
   {
      indx++;
      S();
      Tprime();
   }
   else if (wording[indx] == '$')
   {
      return ;
```

```
    }
}

void T()
{
    S();
    Tprime();
}

void S()
{
    if (wording[indx] == 'a' || wording[indx] == '>')
    {
        indx++;
    }
    else if (wording[indx] == '(')
    {
        indx++;
        T();
        if (wording[indx] == ')')
        {
            indx++;
            return;
        }
    }
    else
    {
        printf("\tError char in S: %c\t", wording[indx]);
        invalid();
    }
}

int main()
{
    printf("Enter the message: ");
    scanf("%s", wording);
    S();
    if (wording[indx] == '$')
    {
        valid();
    }
    else
    {
        invalid();
    }
    return 0;
}
```

Output:



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

char wording[128];
int indx = 0;

/*
S→a | > | ( T )
T→T,S | S  // has left recusrion so below 2 lines removes left recursion
    T->S T'
    T'->,ST' | null
*/

```

```
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_1.c && ./a.out
Enter the message: (a)$

===================SUCCESS!===================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_1.c && ./a.out
Enter the message: ((a))$

===================SUCCESS!===================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_1.c && ./a.out
Enter the message: $
        Error char in S: $
===================ERROR!===================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_1.c && ./a.out
Enter the message: a()$

===================ERROR!===================
student@lpcp-22:~/Documents/220905536/week6$ 
```

2. S→UVW
   U →(S) | aSb | d
   V → aV | ϵ
   W → cW | ϵ

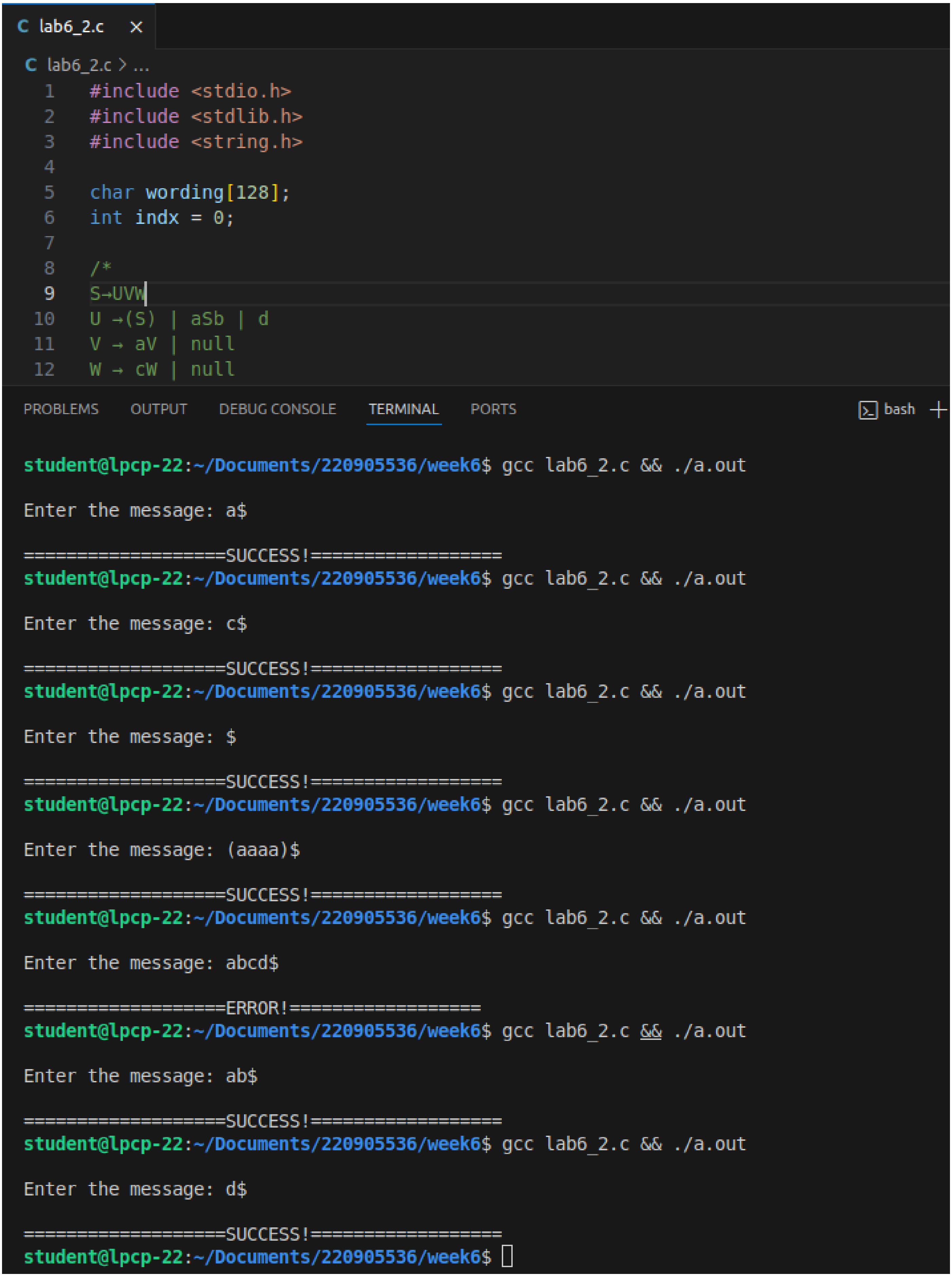
   Source Code:

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

char wording[128];
int indx = 0;

/*
S→UVW
U →(S) | aSb | d
V → aV | null
W → cW | null
```

```
*/

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

void valid()
{
   printf("\n===================SUCCESS!==================\n");
   exit(0);
}

void invalid()
{
   printf("\n===================ERROR!==================\n");
   exit(0);
}

void W()
{
   if (wording[indx] == 'c')
   {
      indx++;
      W();
   }
   else if (wording[indx] == '$')
   {
      return;
   }
}

void V()
{
   if (wording[indx] == 'a')
   {
      indx++;
      V();
   }
   else if (wording[indx] == '$')
   {
      return;
   }
}

void U()
{
   if (wording[indx] == '(')
   {
      indx++;
      S();
      if (wording[indx] == ')')
      {
         indx++;
         return;
```

```
        }
    }
    else if (wording[indx] == 'a')
    {
        indx++;
        S();
        if (wording[indx] == 'b')
        {
            indx++;
            return;
        }
    }
    else if (wording[indx] == 'd')
    {
        indx++;
        return;
    }
}

void S()
{
    if (wording[indx] == '$')
    {
        return;
    }
    U();
    V();
    W();
}

int main()
{
    printf("\nEnter the message: ");
    scanf("%s", wording);
    S();
    if (wording[indx] == '$')
    {
        valid();
    }
    else
    {
        invalid();
    }
    return 0;
}
```

Output:



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

char wording[128];
int indx = 0;

/*
S→UVW
U →(S) | aSb | d
V → aV | null
W → cW | null
```

```
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_2.c && ./a.out

Enter the message: a$

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_2.c && ./a.out

Enter the message: c$

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_2.c && ./a.out

Enter the message: $

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_2.c && ./a.out

Enter the message: (aaaa)$

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_2.c && ./a.out

Enter the message: abcd$

===================ERROR!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_2.c && ./a.out

Enter the message: ab$

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_2.c && ./a.out

Enter the message: d$

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ 
```

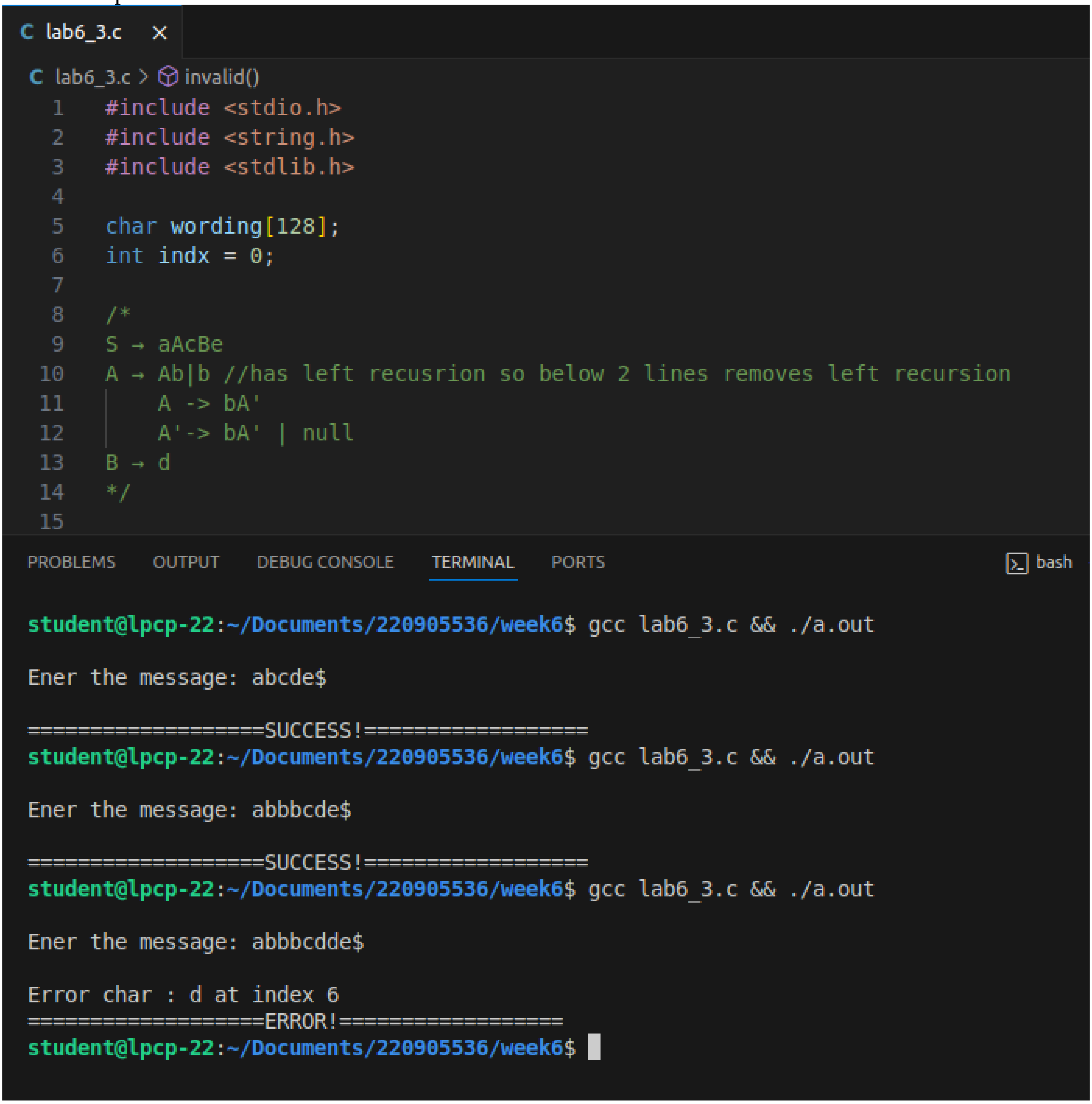
3. S→aAcBe
   A→Ab|b
   B→d

Source Code:

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

char wording[128];
int indx = 0;

/*
S → aAcBe
A → Ab|b //has left recusrion so below 2 lines removes left recursion
   A -> bA'
   A'-> bA' | null
B → d
*/

void S();
void A();
void Aprime();
void B();

void valid()
{
   printf("\n===================SUCCESS!==================\n");
   exit(0);
}

void invalid()
{
   printf("\nError char : %c at index %d", wording[indx], indx);
   printf("\n===================ERROR!==================\n");
   exit(0);
}

void A()
{
   if (wording[indx] == 'b')
   {
      indx++;
      Aprime();
   }
   else if (wording[indx] == '$')
   {
      return;
   }
}

void Aprime()
{
   if (wording[indx] == 'b')
   {
      indx++;
      Aprime();
   }
   else if (wording[indx] == '$')
```

```
    {
        return;
    }
}

void B()
{
    if (wording[indx] == 'd')
    {
        indx++;
        return;
    }
    else
    {
        invalid();
    }
}

void S()
{
    if (wording[indx] == 'a')
    {
        indx++;
        A();
        if (wording[indx] == 'c')
        {
            indx++;
            B();
            if (wording[indx] == 'e')
            {
                indx++;
                return;
            }
        }
        else
        {
            invalid();
        }
    }
    else
    {
        invalid();
    }
}

int main()
{
    printf("\nEner the message: ");
    scanf("%s", wording);
    S();
    if (wording[indx] == '$')
    {
        valid();
    }
    else
    {
        invalid();
    }
    return 0;
}
```

Output:



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

char wording[128];
int indx = 0;

/*
S → aAcBe
A → Ab|b //has left recusrion so below 2 lines removes left recursion
    A -> bA'
    A'-> bA' | null
B → d
*/

```

```
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_3.c && ./a.out

Ener the message: abcde$

===================SUCCESS!===================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_3.c && ./a.out

Ener the message: abbbcde$

===================SUCCESS!===================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_3.c && ./a.out

Ener the message: abbbcdde$

Error char : d at index 6
===================ERROR!===================
student@lpcp-22:~/Documents/220905536/week6$
```

4. S →(L) | a
   L → L,S | S

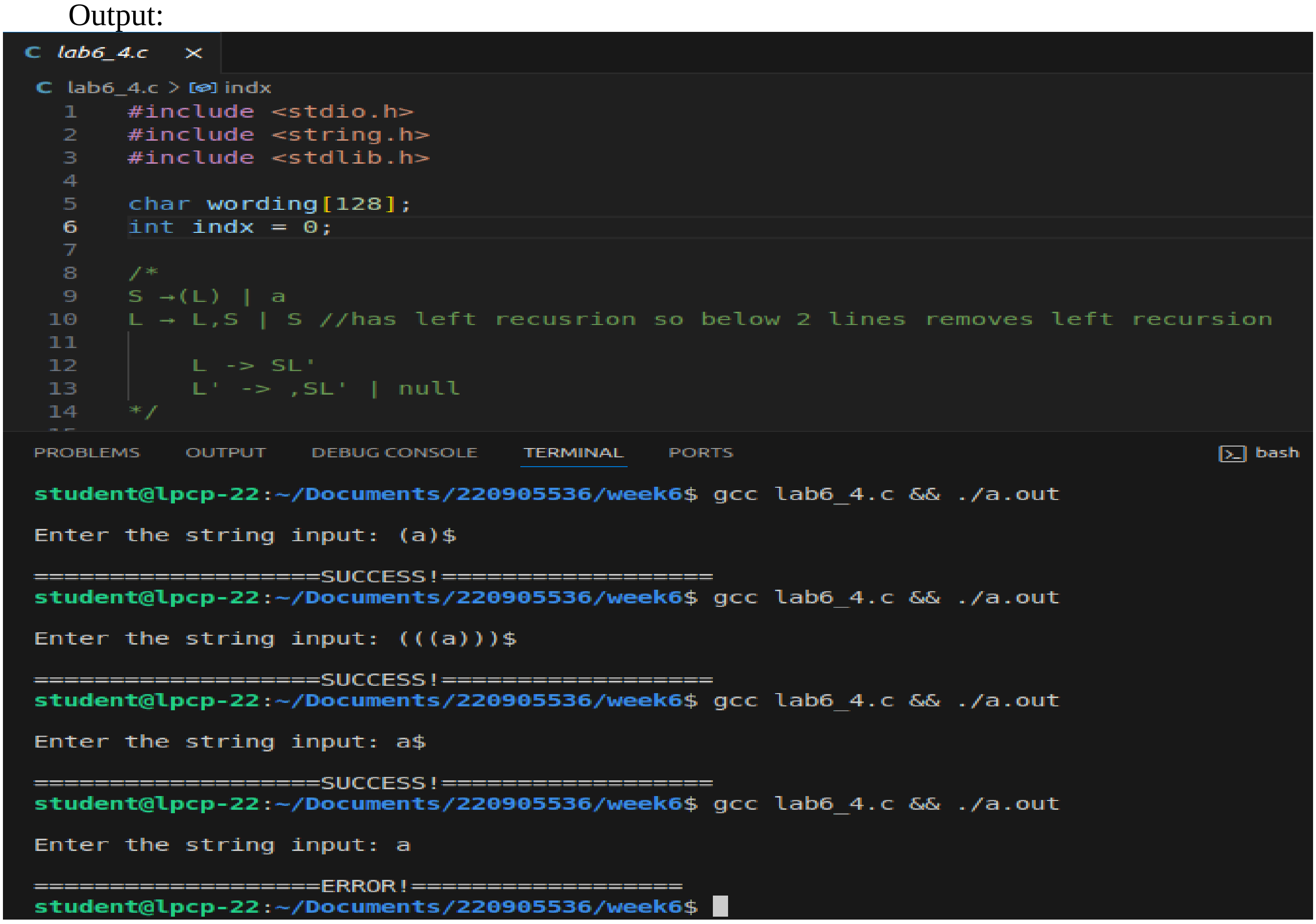
Source Code:

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

char wording[128];
```

```
int indx = 0;

/*
S →(L) | a
L → L,S | S //has left recusrion so below 2 lines removes left recursion

   L -> SL'
   L' -> ,SL' | null
*/

void S();
void L();
void Lprime();

void valid()
{
   printf("\n===================SUCCESS!==================\n");
   exit(0);
}

void invalid()
{
   printf("\n===================ERROR!==================\n");
   exit(0);
}

void Lprime()
{
   if (wording[indx] == ',')
   {
      indx++;
      S();
      Lprime();
   }
   else if (wording[indx] == '$')
   {
      return;
   }
}

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

void S()
{
   if (wording[indx] == 'a')
   {
      indx++;
   }
   else if (wording[indx] == '(')
   {
      indx++;
```

```
            L();
            if (wording[indx]==')')
            {
                indx++;
                return;
            }
        }
        else
        {
            printf("\tError char in S: %c\t", wording[indx]);
            invalid();
        }
    }

    int main()
    {
        printf("\nEnter the string input: ");
        scanf("%s", wording);
        S();
        if (wording[indx] == '$')
        {
            valid();
        }
        else
        {
            invalid();
        }
        return 0;
    }
```

Output:



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

char wording[128];
int indx = 0;

/*
S →(L) | a
L → L,S | S //has left recusrion so below 2 lines removes left recursion

    L -> SL'
    L' -> ,SL' | null
*/
```

```
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_4.c && ./a.out

Enter the string input: (a)$

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_4.c && ./a.out

Enter the string input: (((a)))$

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_4.c && ./a.out

Enter the string input: a$

===================SUCCESS!==================
student@lpcp-22:~/Documents/220905536/week6$ gcc lab6_4.c && ./a.out

Enter the string input: a

===================ERROR!==================
student@lpcp-22:~/Documents/220905536/week6$ 
```