Write a C program to construct recursive descent parsing for the given grammar.

**Code:**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

char \*input;

int position = 0;

void E();

void E\_prime();

void T();

void T\_prime();

void F();

void error() {

printf("Parsing failed at position %d: %s\n", position, &input[position]);

exit(1);

}

void match(char expected) {

if(input[position] == expected)

position++;

else

error();

}

// Function to check for "id" token

void match\_id() {

if(strncmp(&input[position], "id", 2) == 0)

position += 2; // Move position forward by 2 (length of "id")

else

error();

}

void E() {

T();

E\_prime();

}

void E\_prime() {

if(input[position] == '+') {

match('+');

T();

E\_prime();

}

}

void T() {

F();

T\_prime();

}

void T\_prime() {

if(input[position] == '\*') {

match('\*');

F();

T\_prime();

}

}

void F() {

if(strncmp(&input[position], "id", 2) == 0) {

match\_id(); // Match "id" token

} else if(input[position] == '(') {

match('(');

E();

match(')');

} else {

error();

}

}

int main() {

char str[100];

printf("Enter an expression: ");

scanf("%s", str);

input = str;

E();

if(input[position] == '\0')

printf("Parsing successful!\n");

else

printf("Parsing failed at position %d\n", position);

return 0;

}

OUTPUT:

