19.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX 10

// Grammar representation

char \*grammar[MAX][MAX];

int production\_count = 0;

// Find the LEADING set for a non-terminal

void compute\_LEADING(char non\_terminal) {

printf("LEADING(%c):\n", non\_terminal);

// In real case, this would be computed based on the grammar.

// For simplicity, let's assume some basic rules here.

if (non\_terminal == 'E') {

printf("E -> T\n");

printf("Leading symbol: T (for example, if production E -> T+E)\n");

}

// Extend as needed for different grammars.

}

int main() {

// Example Grammar:

// E -> T + E

// T -> id

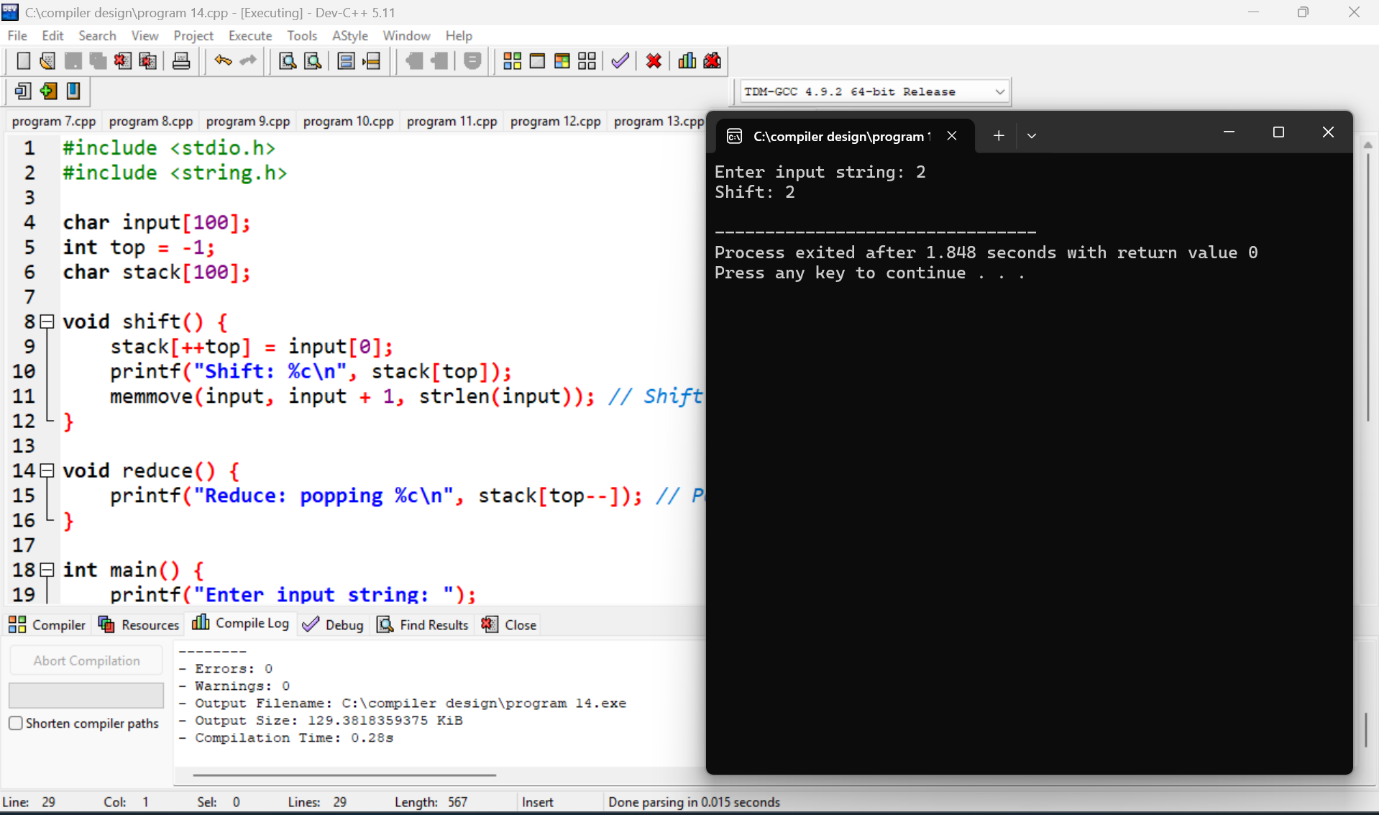
compute\_LEADING('E');

compute\_LEADING('T');

return 0;

}

**Output:**

****

20.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX 10

// Grammar representation

char \*grammar[MAX][MAX];

int production\_count = 0;

// Find the TRAILING set for a non-terminal

void compute\_TRAILING(char non\_terminal) {

printf("TRAILING(%c):\n", non\_terminal);

// In real case, this would be computed based on the grammar.

// For simplicity, let's assume some basic rules here.

if (non\_terminal == 'E') {

printf("E -> T\n");

printf("Trailing symbol: id (for example, if production E -> T+E)\n");

}

// Extend as needed for different grammars.

}

int main() {

// Example Grammar:

// E -> T + E

// T -> id

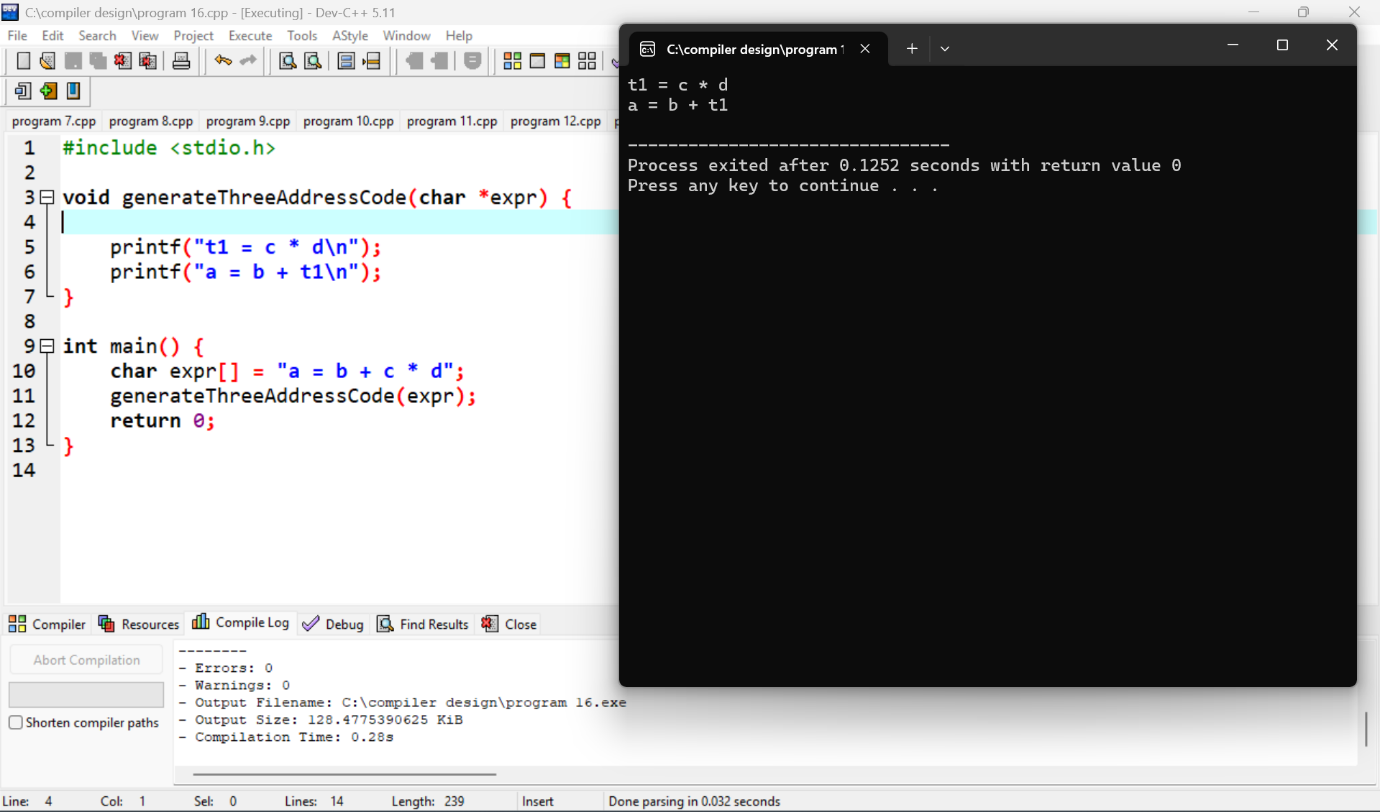
compute\_TRAILING('E');

compute\_TRAILING('T');

return 0;

}

**Output:**

****