CSA 1447-COMPILER DESIGN FOR SYNTAX SMITH

PRACTICAL PROGRAMS 8 - 14

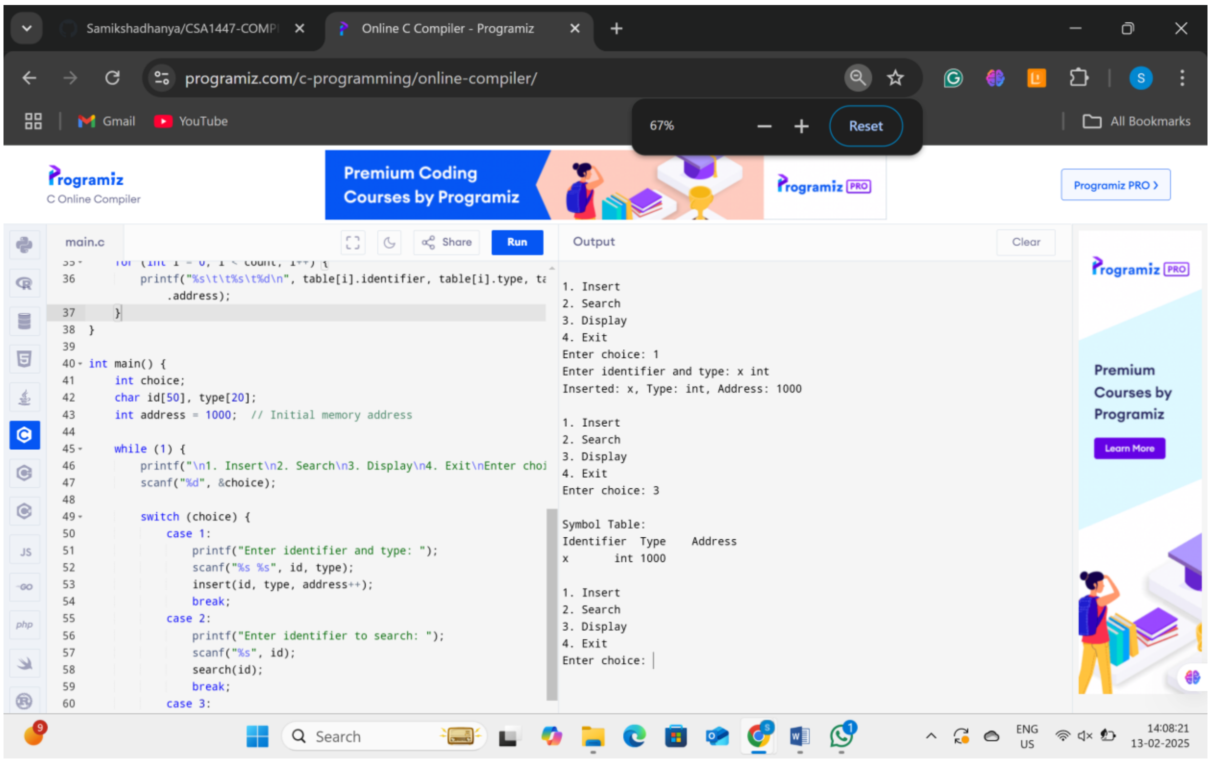
EXPERIMENT-8

PROGRAM:

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

#define MAX 100
struct SymbolTable {
   char identifier[50];
   char type[20];
   int address;
} table[MAX];

int count = 0;

void insert(char *id, char *type, int addr) {
   strcpy(table[count].identifier, id);
   strcpy(table[count].type, type);
   table[count].address = addr;
   count++;
   printf("Inserted: %s, Type: %s, Address: %d\n", id, type, addr);
}

void search(char *id) {
   for (int i = 0; i < count; i++) {
      if (strcmp(table[i].identifier, id) == 0) {
         printf("Found: %s, Type: %s, Address: %d\n", table[i].identifier, table[i].type, table[i].address);
         return;
      }
   }
```

```
    printf("Identifier not found.\n");
}

void display() {
    printf("\nSymbol Table:\n");
    printf("Identifier\tType\tAddress\n");
    for (int i = 0; i < count; i++) {
        printf("%s\t\t%s\t%d\n", table[i].identifier, table[i].type, table[i].address);
    }
}

int main() {
    int choice;
    char id[50], type[20];
    int address = 1000;  // Initial memory address

    while (1) {
        printf("\n1. Insert\n2. Search\n3. Display\n4. Exit\nEnter choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter identifier and type: ");
                scanf("%s %s", id, type);
                insert(id, type, address++);
                break;
            case 2:
                printf("Enter identifier to search: ");
                scanf("%s", id);
                search(id);
                break;
            case 3:
                display();
                break;
            case 4:
                return 0;
        }
    }
}
```

OUTPUT:


Samikshadhanya/CSA1447-COMP
Online C Compiler - Programiz
programiz.com/c-programming/online-compiler/
Gmail
YouTube
67%
Reset
All Bookmarks
Programiz
C Online Compiler
Premium Coding Courses by Programiz
Programiz PRO
main.c
Share
Run
Output
Clear
Premium Courses by Programiz
Learn More
Search
ENG US
14:08:21
13-02-2025

```
        for (int i = 0; i < count; i++) {
            printf("%s\t\t%s\t%d\n", table[i].identifier, table[i].type, ta
                .address);
        }
}

int main() {
    int choice;
    char id[50], type[20];
    int address = 1000;  // Initial memory address

    while (1) {
        printf("\n1. Insert\n2. Search\n3. Display\n4. Exit\nEnter choi
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                printf("Enter identifier and type: ");
                scanf("%s %s", id, type);
                insert(id, type, address++);
                break;
            case 2:
                printf("Enter identifier to search: ");
                scanf("%s", id);
                search(id);
                break;
            case 3:
```

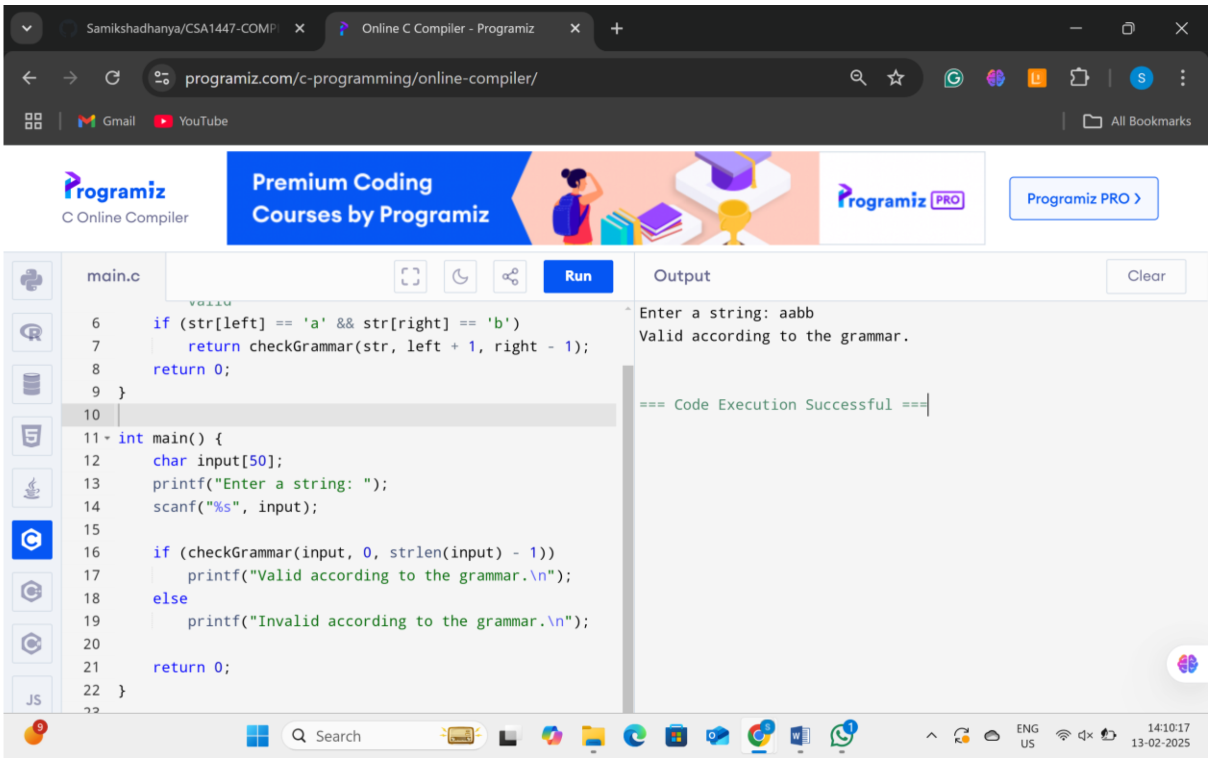
```
1. Insert
2. Search
3. Display
4. Exit
Enter choice: 1
Enter identifier and type: x int
Inserted: x, Type: int, Address: 1000

1. Insert
2. Search
3. Display
4. Exit
Enter choice: 3

Symbol Table:
Identifier  Type    Address
x       int 1000

1. Insert
2. Search
3. Display
4. Exit
Enter choice: 
```

EXPERIMENT-9

PROGRAM:

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

int checkGrammar(char *str, int left, int right) {
    if (left > right) return 1;  // Empty string is valid
    if (str[left] == 'a' && str[right] == 'b')
        return checkGrammar(str, left + 1, right - 1);
    return 0;
}

int main() {
    char input[50];
    printf("Enter a string: ");
    scanf("%s", input);

    if (checkGrammar(input, 0, strlen(input) - 1))
        printf("Valid according to the grammar.\n");
    else
        printf("Invalid according to the grammar.\n");

    return 0;
}
```

OUTPUT:



```
    if (str[left] == 'a' && str[right] == 'b')
        return checkGrammar(str, left + 1, right - 1);
    return 0;
}

int main() {
    char input[50];
    printf("Enter a string: ");
    scanf("%s", input);

    if (checkGrammar(input, 0, strlen(input) - 1))
        printf("Valid according to the grammar.\n");
    else
        printf("Invalid according to the grammar.\n");

    return 0;
}
```

```
Enter a string: aabb
Valid according to the grammar.

=== Code Execution Successful ===
```

Experiment- 10

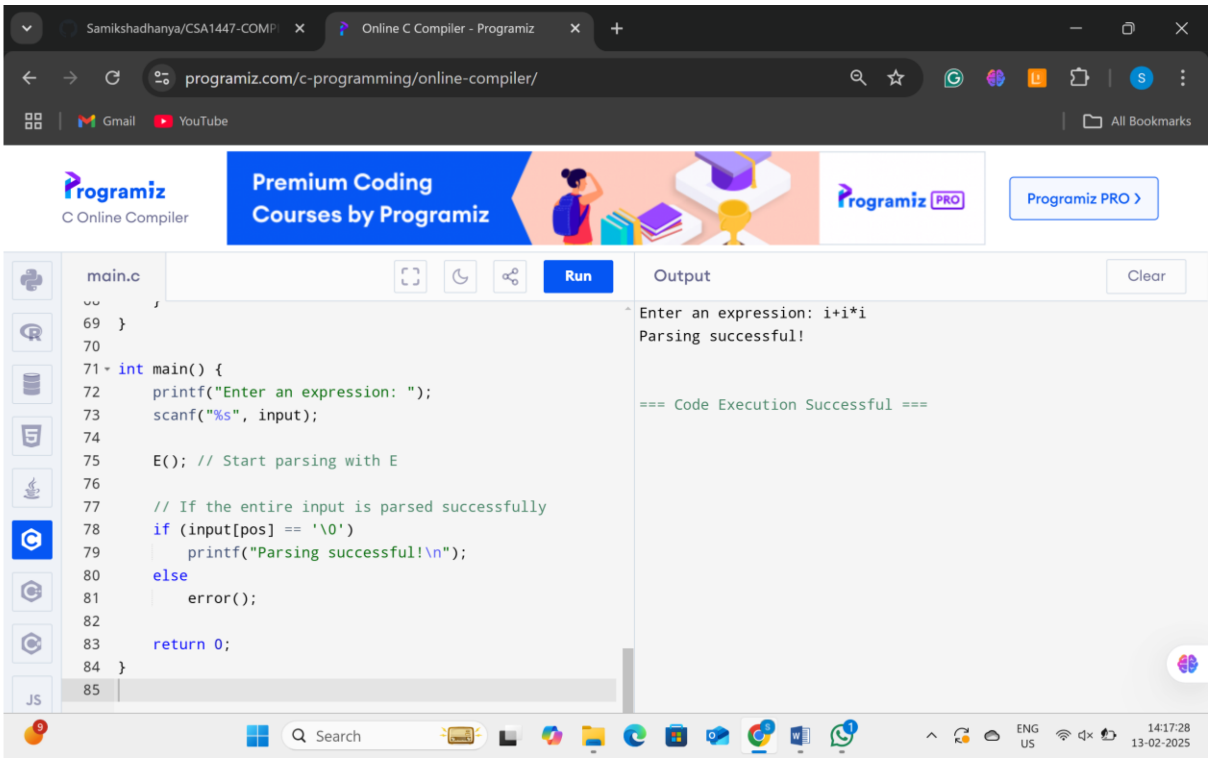
PROGRAM:

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>


char input[100];  // Input string

int pos = 0;      // Pointer to track parsing position


void E();  // Expression

void EPrime();

void T();  // Term

void TPrime();

void F();  // Factor


// Function to handle parsing errors

void error() {

   printf("Error in parsing!\n");

   exit(0);

}


// Function to match a character and move to the next

void match(char expected) {

   if (input[pos] == expected)

      pos++;

   else

      error();

}


// E -> T E'

void E() {

   T();

   EPrime();
```

```
}

// E' -> + T E' | ε
void EPrime() {
  if (input[pos] == '+') { // If '+' is found
    match('+');
    T();
    EPrime();
  }
}

// T -> F T'
void T() {
  F();
  TPrime();
}

// T' -> * F T' | ε
void TPrime() {
  if (input[pos] == '*') { // If '*' is found
    match('*');
    F();
    TPrime();
  }
}

// F -> (E) | id (assuming 'id' starts with 'i')
void F() {
  if (input[pos] == '(') { // If '(' is found
    match('(');
    E();
```

```
        match(')');
    } else if (input[pos] == 'i') { // Assuming 'id' is represented as 'i'
        match('i');
    } else {
        error();
    }
}

int main() {
    printf("Enter an expression: ");
    scanf("%s", input);

    E(); // Start parsing with E

    // If the entire input is parsed successfully
    if (input[pos] == '\0')
        printf("Parsing successful!\n");
    else
        error();

    return 0;
}
```

OUTPUT:



```
}

int main() {
    printf("Enter an expression: ");
    scanf("%s", input);

    E(); // Start parsing with E

    // If the entire input is parsed successfully
    if (input[pos] == '\0')
        printf("Parsing successful!\n");
    else
        error();

    return 0;
}
```

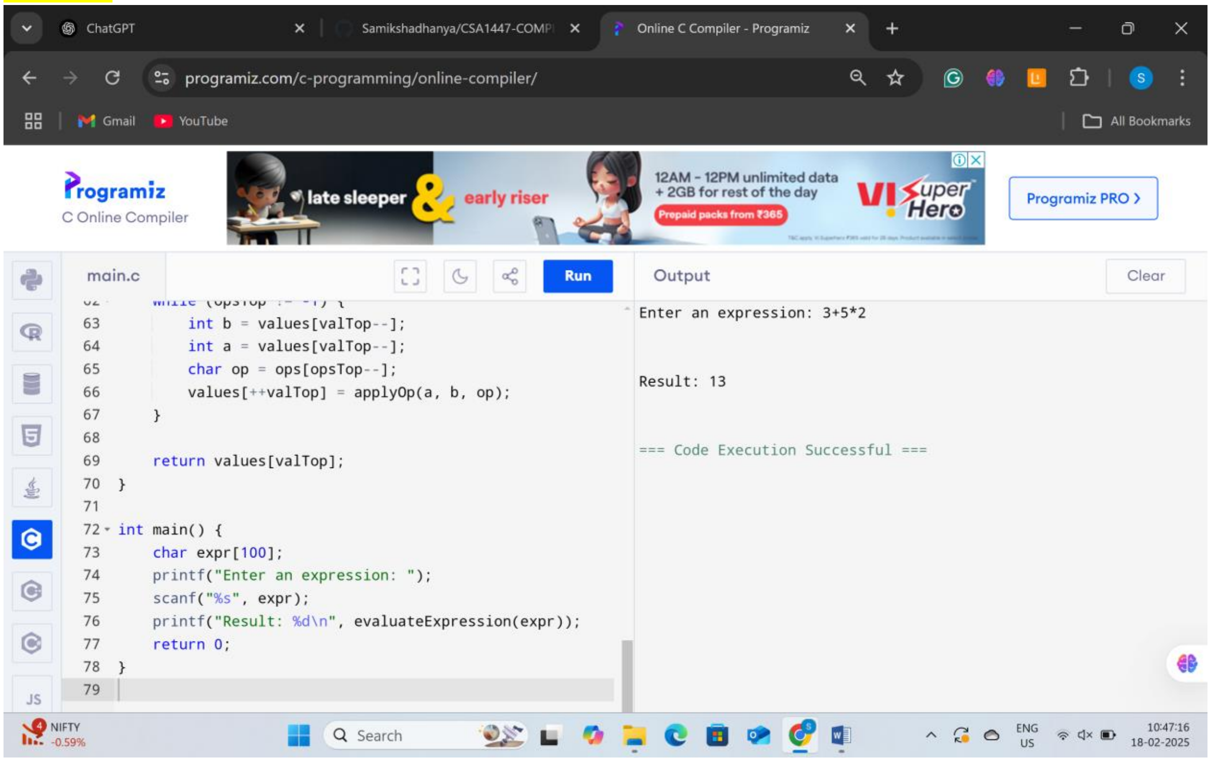
```
Enter an expression: i+i*i
Parsing successful!


=== Code Execution Successful ===
```

Experiment-11

PROGRAM:

```
#include <stdio.h>

#include <ctype.h>

#include <stdlib.h>


int precedence(char op) {

    if (op == '+' || op == '-') return 1;

    if (op == '*' || op == '/') return 2;

    if (op == '^') return 3;

    return 0;

}


int applyOp(int a, int b, char op) {

    switch (op) {

        case '+': return a + b;

        case '-': return a - b;

        case '*': return a * b;
        case '/': return a / b;
        case '^': {
            int res = 1;
            for (int i = 0; i < b; i++) res *= a;
            return res;
        }
    }
    return 0;
}

int evaluateExpression(char* expr) {
    int values[100], valTop = -1;
    char ops[100];
    int opsTop = -1;

    for (int i = 0; expr[i] != '\0'; i++) {
        if (isdigit(expr[i])) {
            int val = 0;
            while (isdigit(expr[i])) {
                val = (val * 10) + (expr[i] - '0');
                i++;
            }
            i--;
            values[++valTop] = val;
        } else if (expr[i] == '(') {
            ops[++opsTop] = expr[i];
        } else if (expr[i] == ')') {
            while (opsTop != -1 && ops[opsTop] != '(') {
                int b = values[valTop--];
                int a = values[valTop--];
                char op = ops[opsTop--];
                values[++valTop] = applyOp(a, b, op);
            }
            opsTop--;
        } else {
            while (opsTop != -1 && precedence(ops[opsTop]) >= precedence(expr[i])) {
                int b = values[valTop--];
                int a = values[valTop--];
                char op = ops[opsTop--];
                values[++valTop] = applyOp(a, b, op);
            }
            ops[++opsTop] = expr[i];
        }
    }

    while (opsTop != -1) {
        int b = values[valTop--];
        int a = values[valTop--];
        char op = ops[opsTop--];
        values[++valTop] = applyOp(a, b, op);
    }

    return values[valTop];
}

int main() {
    char expr[100];
    printf("Enter an expression: ");
    scanf("%s", expr);
    printf("Result: %d\n", evaluateExpression(expr));
    return 0;
}
```

OUTPUT:



```
    int b = values[valTop--];
    int a = values[valTop--];
    char op = ops[opsTop--];
    values[++valTop] = applyOp(a, b, op);
  }

  return values[valTop];
}

int main() {
  char expr[100];
  printf("Enter an expression: ");
  scanf("%s", expr);
  printf("Result: %d\n", evaluateExpression(expr));
  return 0;
}
```

```
Enter an expression: 3+5*2


Result: 13


=== Code Execution Successful ===
```

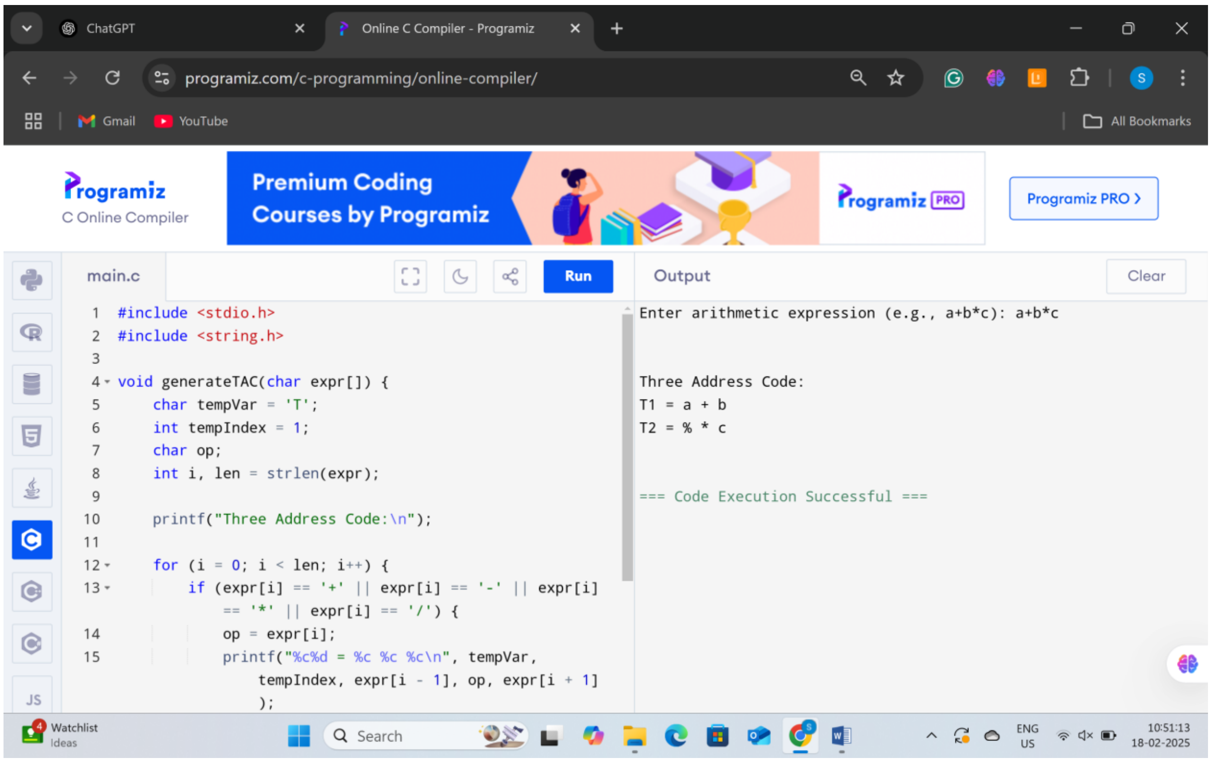
Experiment -12

PROGRAM:

```
#include <stdio.h>

#include <string.h>


void generateTAC(char expr[]) {

   char tempVar = 'T';

   int tempIndex = 1;

   char op;

   int i, len = strlen(expr);


   printf("Three Address Code:\n");


   for (i = 0; i < len; i++) {

      if (expr[i] == '+' || expr[i] == '-' || expr[i] == '*' || expr[i] == '/') {

         op = expr[i];

         printf("%c%d = %c %c %c\n", tempVar, tempIndex, expr[i - 1], op, expr[i + 1]);
```

```
            expr[i + 1] = tempVar + tempIndex - '0';
            tempIndex++;
        }
    }
}

int main() {
    char expr[50];
    printf("Enter arithmetic expression (e.g., a+b*c): ");
    scanf("%s", expr);
    generateTAC(expr);
    return 0;
}
```

OUTPUT:


ChatGPT
Online C Compiler - Programiz
programiz.com/c-programming/online-compiler/
Programiz
C Online Compiler
Premium Coding Courses by Programiz
Programiz PRO
main.c
Run
Output
Clear

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

void generateTAC(char expr[]) {
    char tempVar = 'T';
    int tempIndex = 1;
    char op;
    int i, len = strlen(expr);

    printf("Three Address Code:\n");

    for (i = 0; i < len; i++) {
        if (expr[i] == '+' || expr[i] == '-' || expr[i]
            == '*' || expr[i] == '/') {
            op = expr[i];
            printf("%c%d = %c %c %c\n", tempVar,
                tempIndex, expr[i - 1], op, expr[i + 1]
                );
```

```
Enter arithmetic expression (e.g., a+b*c): a+b*c


Three Address Code:
T1 = a + b
T2 = % * c


=== Code Execution Successful ===
```

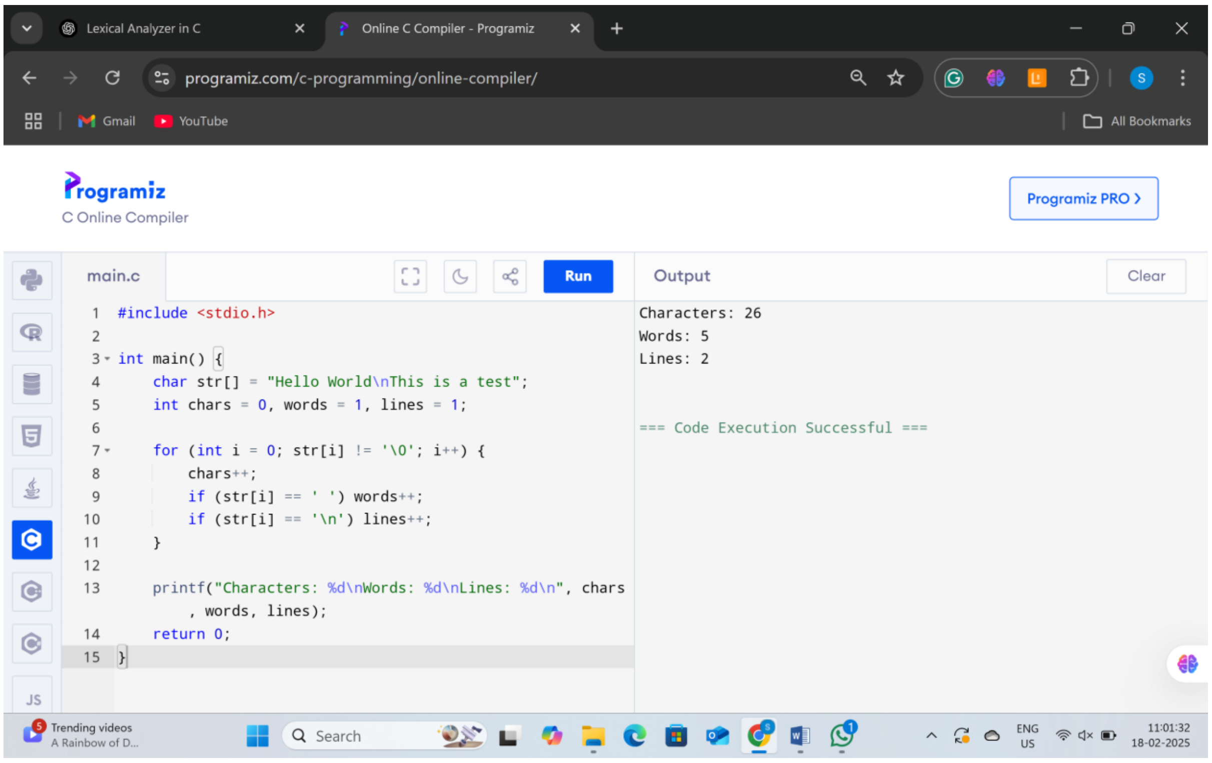
Experiment-13

PROGRAM:

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

```
int main() {
   char str[] = "Hello World\nThis is a test";
   int chars = 0, words = 1, lines = 1;

   for (int i = 0; str[i] != '\0'; i++) {
      chars++;
      if (str[i] == ' ') words++;
      if (str[i] == '\n') lines++;
   }

   printf("Characters: %d\nWords: %d\nLines: %d\n", chars, words, lines);
   return 0;
}
```

OUTPUT:



```
#include <stdio.h>

int main() {
    char str[] = "Hello World\nThis is a test";
    int chars = 0, words = 1, lines = 1;

    for (int i = 0; str[i] != '\0'; i++) {
        chars++;
        if (str[i] == ' ') words++;
        if (str[i] == '\n') lines++;
    }

    printf("Characters: %d\nWords: %d\nLines: %d\n", chars
        , words, lines);
    return 0;
}
```

```
Characters: 26
Words: 5
Lines: 2


=== Code Execution Successful ===
```

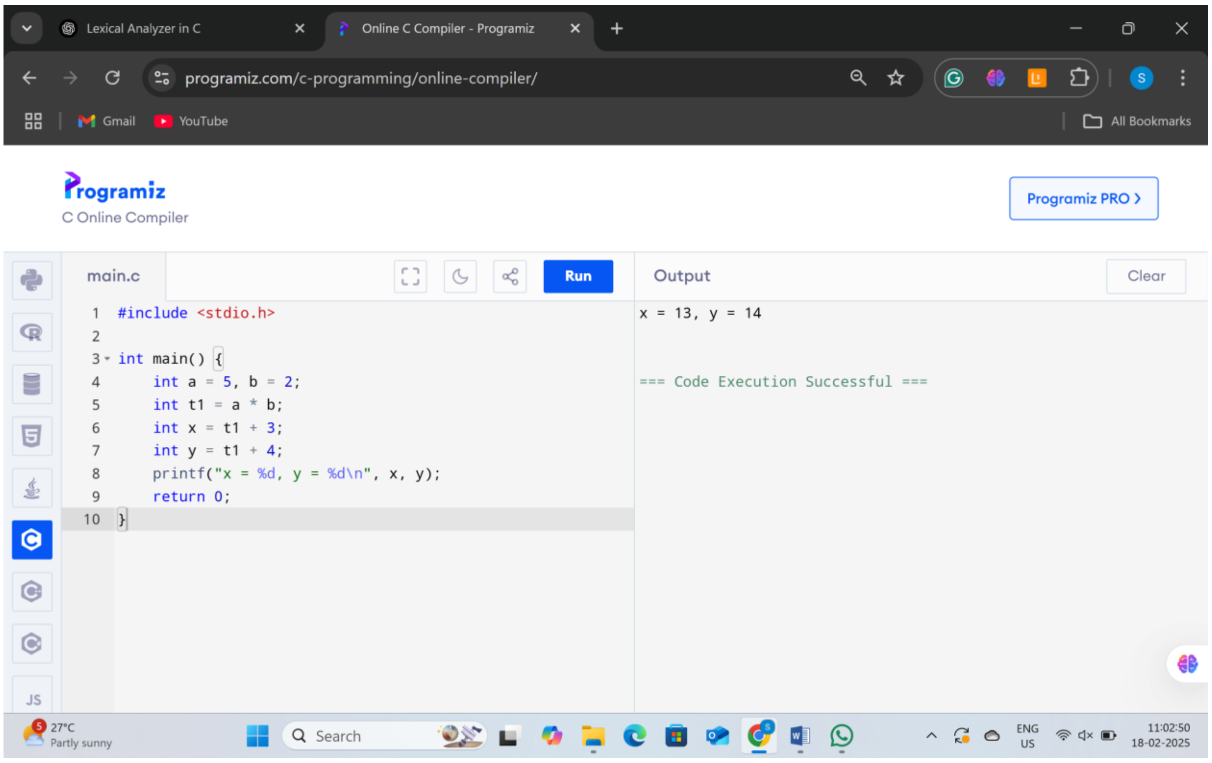
Experiment-14

PROGRAM:

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

```
int main() {
  int a = 5, b = 2;
  int t1 = a * b;
  int x = t1 + 3;
  int y = t1 + 4;
  printf("x = %d, y = %d\n", x, y);
  return 0;
}
```

OUTPUT:



```
#include <stdio.h>

int main() {
    int a = 5, b = 2;
    int t1 = a * b;
    int x = t1 + 3;
    int y = t1 + 4;
    printf("x = %d, y = %d\n", x, y);
    return 0;
}
```

```
x = 13, y = 14


=== Code Execution Successful ===
```