

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();
```

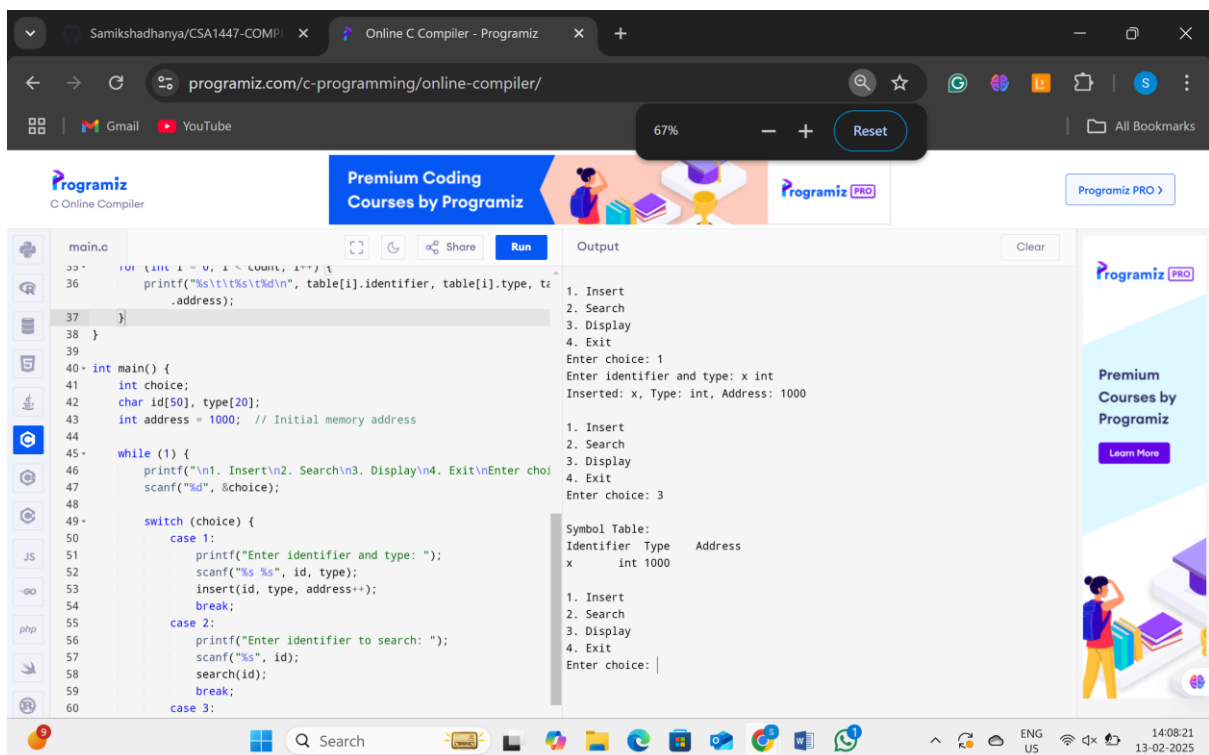
```
break;
```

case 4:

```
return 0;
```

```
}  
}  
}
```

OUTPUT:



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:

The screenshot shows a web browser window with the URL `programiz.com/c-programming/online-compiler/`. The page features a banner for "Premium Coding Courses by Programiz" and a "Programiz PRO" button. Below the banner, the online compiler interface is visible. On the left, the code editor shows the following C code in `main.c`:

```

6   if (str[left] == 'a' && str[right] == 'b')
7       return checkGrammar(str, left + 1, right - 1);
8   return 0;
9 }
10
11 int main() {
12     char input[50];
13     printf("Enter a string: ");
14     scanf("%s", input);
15
16     if (checkGrammar(input, 0, strlen(input) - 1))
17         printf("Valid according to the grammar.\n");
18     else
19         printf("Invalid according to the grammar.\n");
20
21     return 0;
22 }

```

On the right, the "Output" panel displays the execution results:

```

Enter a string: aabb
Valid according to the grammar.

=== Code Execution Successful ===

```

The bottom of the image shows a Windows taskbar with various application icons and a system clock indicating 14:10:17 on 13-02-2025.

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' |  $\epsilon$ 
```

```
void EPrime() {
```

```
    if (input[pos] == '+') { // If '+' is found
```

```
        match('+');
```

```
        T();
```

```
        EPrime();
```

```
    }
```

```
}
```

```
// T -> F T'
```

```
void T() {
```

```
    F();
```

```
    TPrime();
```

```
}
```

```
// T' -> * F T' |  $\epsilon$ 
```

```
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
```

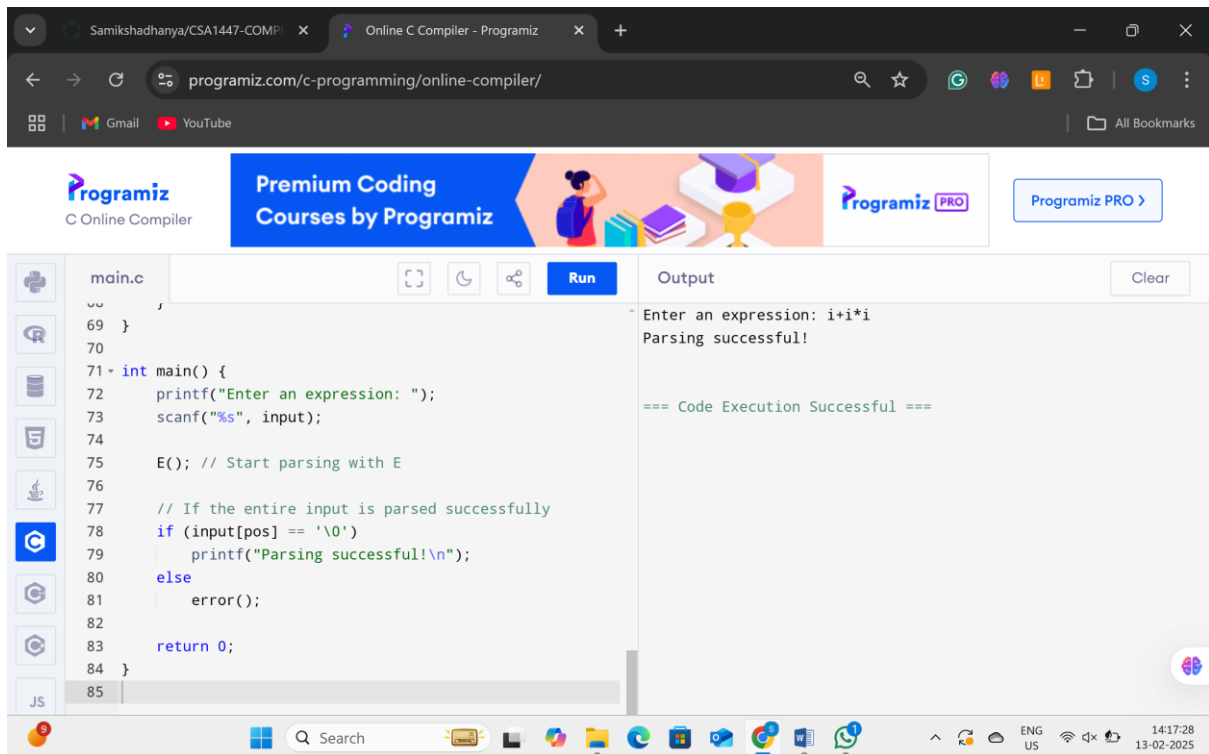
```
        match('(');
```

```
        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:



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--];
            }
        }
    }

    return values[valTop];
}

```

```

        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];
}
}

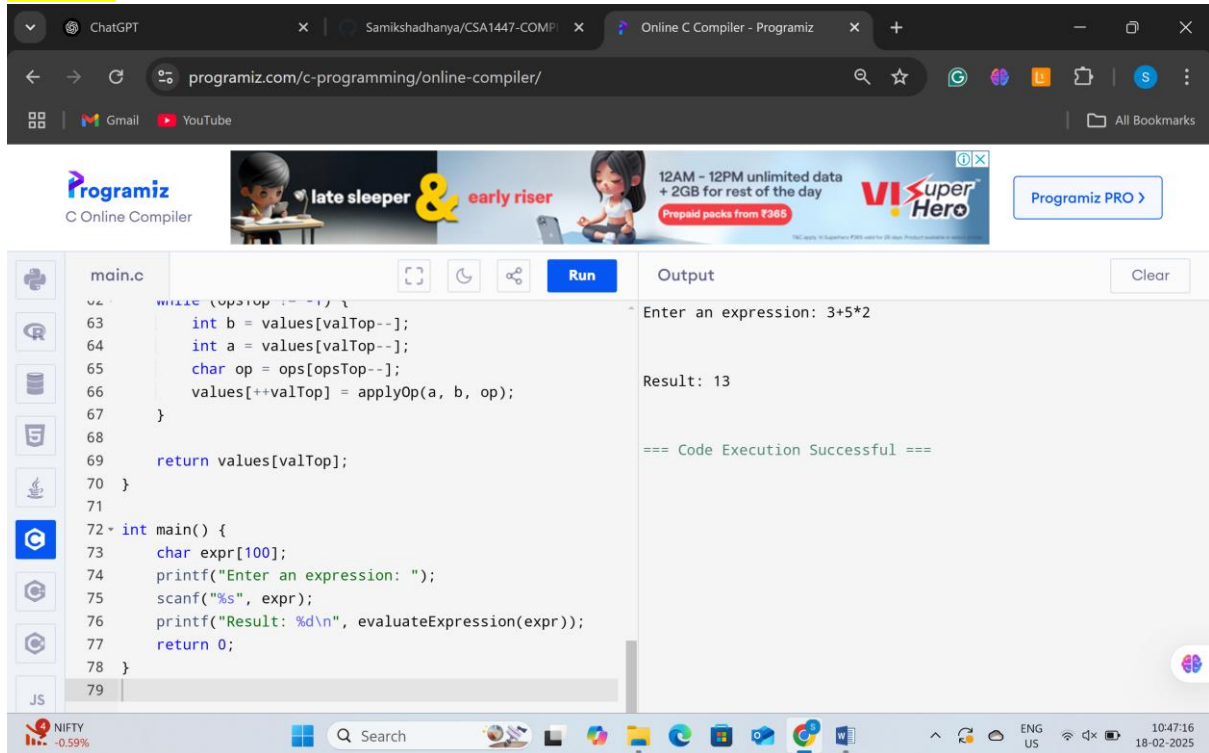
while (opsTop != -1) {
    int b = values[valTop--];
    int a = values[valTop--];
    char op = ops[opsTop--];
    values[++valTop] = applyOp(a, b, op);
}

return values[valTop];
}

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

```

OUTPUT:



The screenshot shows a web browser with the URL `programiz.com/c-programming/online-compiler/`. The page displays the Programiz Online C Compiler interface. The code editor on the left contains the following C code:

```
63 while (opTop != -1) {
64     int b = values[valTop--];
65     int a = values[valTop--];
66     char op = ops[opsTop--];
67     values[++valTop] = applyOp(a, b, op);
68 }
69 return values[valTop];
70 }
71
72 int main() {
73     char expr[100];
74     printf("Enter an expression: ");
75     scanf("%s", expr);
76     printf("Result: %d\n", evaluateExpression(expr));
77     return 0;
78 }
```

The output window on the right shows the execution results:

```
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:

The screenshot shows a web browser window with the URL `programiz.com/c-programming/online-compiler/`. The page features a header for "Programiz C Online Compiler" and a banner for "Premium Coding Courses by Programiz". The main area displays a C program in a text editor, which has been executed. The output window shows the following text:

```

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

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

=== Code Execution Successful ===

```

The C code in the editor is as follows:

```

1 #include <stdio.h>
2 #include <string.h>
3
4 void generateTAC(char expr[]) {
5     char tempVar = 'T';
6     int tempIndex = 1;
7     char op;
8     int i, len = strlen(expr);
9
10    printf("Three Address Code:\n");
11
12    for (i = 0; i < len; i++) {
13        if (expr[i] == '+' || expr[i] == '-' || expr[i]
            == '*' || expr[i] == '/') {
14            op = expr[i];
15            printf("%c%d = %c %c %c\n", tempVar,
                tempIndex, expr[i - 1], op, expr[i + 1]
            );

```

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:

The screenshot shows a web browser window with the URL `programiz.com/c-programming/online-compiler/`. The page title is "Programiz C Online Compiler". The code editor contains the following C code:

```

1 #include <stdio.h>
2
3 int main() {
4     char str[] = "Hello World\nThis is a test";
5     int chars = 0, words = 1, lines = 1;
6
7     for (int i = 0; str[i] != '\0'; i++) {
8         chars++;
9         if (str[i] == ' ') words++;
10        if (str[i] == '\n') lines++;
11    }
12
13    printf("Characters: %d\nWords: %d\nLines: %d\n", chars
14           , words, lines);
15    return 0;
16 }

```

The "Output" tab shows the following results:

```

Characters: 26
Words: 5
Lines: 2

=== Code Execution Successful ===

```

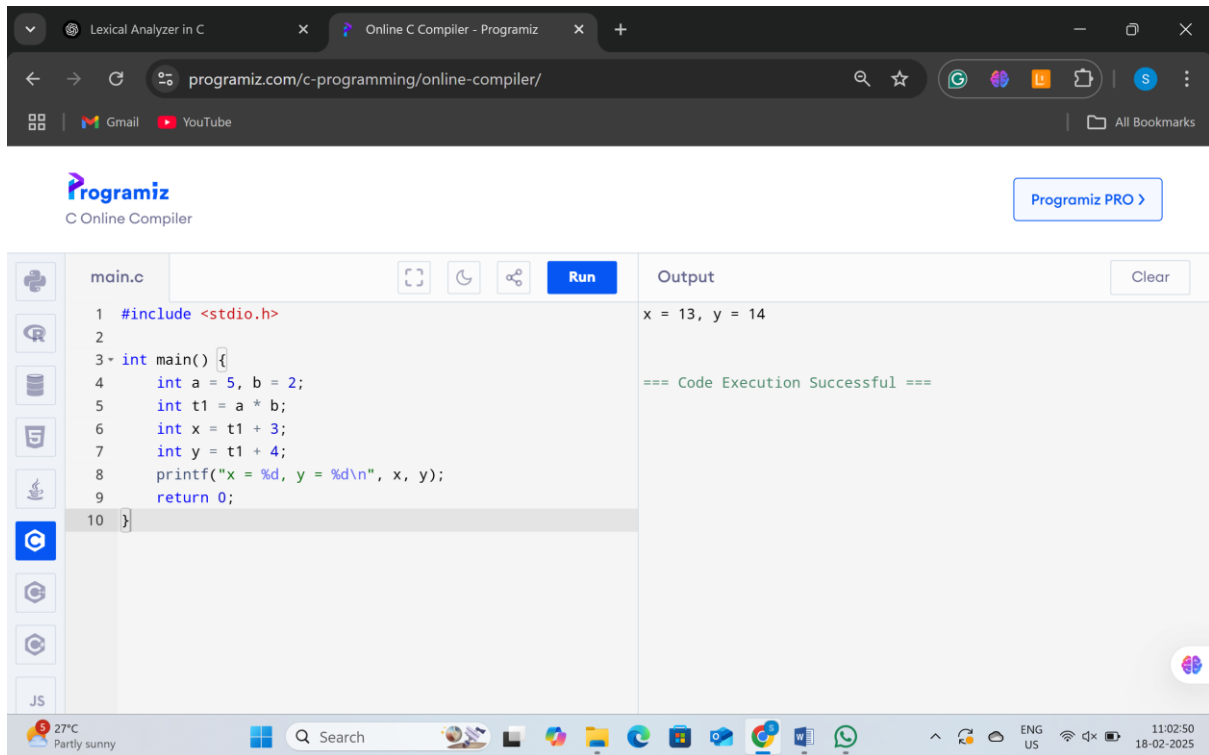
Experiment-14

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:



The screenshot shows a web browser window with the URL `programiz.com/c-programming/online-compiler/`. The page features the Programiz logo and a "Programiz PRO" button. The main content area is divided into two panels: a code editor on the left and an output panel on the right. The code editor contains the following C code:

```
main.c  
1 #include <stdio.h>  
2  
3 int main() {  
4     int a = 5, b = 2;  
5     int t1 = a * b;  
6     int x = t1 + 3;  
7     int y = t1 + 4;  
8     printf("x = %d, y = %d\n", x, y);  
9     return 0;  
10 }
```

The output panel displays the result of the program execution:

```
Output  
x = 13, y = 14  
  
=== Code Execution Successful ===
```

The browser's taskbar at the bottom shows the system time as 11:02:50 on 18-02-2025, with a temperature of 27°C and a "Partly sunny" weather forecast.