Date: 15/09/2025

Experiment No: 06

Aim: To implementation of intermediate code generation.

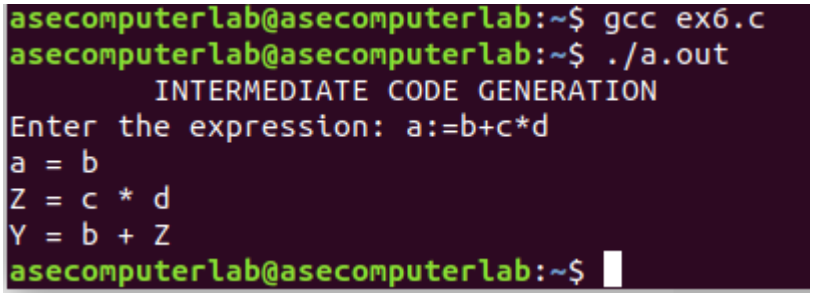
Code:

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

int tmpch = 'Z'; // For temporary variable naming
char str[100];

struct Operation {
    int pos;
    char op;
} ops[20];

int opCount = 0;

void findOperators() {
    int i;
    // Operator precedence order: */ +-
    for (i = 0; str[i] != '\0'; i++) {
        if (str[i] == ':' && str[i+1] == '=') {
            ops[opCount].pos = i;
            ops[opCount].op = '=';
            opCount++;
            i++; // Skip the '='
        }
    }

    for (i = 0; str[i] != '\0'; i++) {
        if (str[i] == '*' || str[i] == '/') {
            ops[opCount].pos = i;
            ops[opCount].op = str[i];
            opCount++;
        }
    }

    for (i = 0; str[i] != '\0'; i++) {
        if (str[i] == '+' || str[i] == '-') {
            ops[opCount].pos = i;
            ops[opCount].op = str[i];
            opCount++;
        }
    }
}
```

```c
void fleft(int pos, char *left) {
    int i = pos - 1;
    int j = 0;
    while (i >= 0 && str[i] != '+' && str[i] != '-' && str[i] != '*' && str[i] != '/' && str[i] != ':' && str[i] != '=') {
        i--;
    }
    i++;
    while (i < pos) {
        left[j++] = str[i++];
    }
    left[j] = '\0';
}

void fright(int pos, char *right) {
    int i = pos + 1;
    if (str[pos] == ':' && str[pos+1] == '=') {
        i++; // skip '=' in ':='
    }

    int j = 0;
    while (str[i] != '\0' && str[i] != '+' && str[i] != '-' && str[i] != '*' && str[i] != '/' && str[i] != ':' && str[i] != '=') {
        right[j++] = str[i++];
    }
    right[j] = '\0';
}

void replace(int pos, char tmp) {
    int i = pos - 1;
    while (i >= 0 && str[i] != '+' && str[i] != '-' && str[i] != '*' && str[i] != '/' && str[i] != ':' && str[i] != '=') {
        i--;
    }
    i++;
    str[i++] = tmp;

    int j = pos + 1;
    if (str[pos] == ':' && str[pos+1] == '=') {
        j++; // skip '=' in ':='
    }

    while (str[j] != '\0' && str[j] != '+' && str[j] != '-' && str[j] != '*' && str[j] != '/' && str[j] != ':' && str[j] != '=') {
        str[i++] = ' ';
        j++;
```

```c
    while (str[j] != '\0' && str[j] != '+' && str[j] != '-' && str[j] != '*' && str[j] != '/' && str[j] != ':' && str[j] != '=') {
        str[i++] = ' ';
        j++;
    }
    str[i] = '\0';
}

void generateCode() {
    char left[10], right[10];
    for (int i = 0; i < opCount; i++) {
        int pos = ops[i].pos;
        char op = ops[i].op;

        fleft(pos, left);
        fright(pos, right);

        if (op == '=') {
            printf("%s = %s\n", left, right);
        } else {
            printf("%c = %s %c %s\n", tmpch, left, op, right);
            replace(pos, tmpch);
            tmpch--;
        }
    }
}

int main() {
    printf("\tINTERMEDIATE CODE GENERATION\n");
    printf("Enter the expression: ");
    scanf("%s", str);

    findOperators();
    generateCode();

    return 0;
}
```

Output:



Results: The program for the implementation of intermediate code generation has been executed successfully.