Name: Aman Kumar Rauniyar
Roll No: CH.EN.U4CSE22174

Lab-7

Title: To implementation of intermediate code generation.

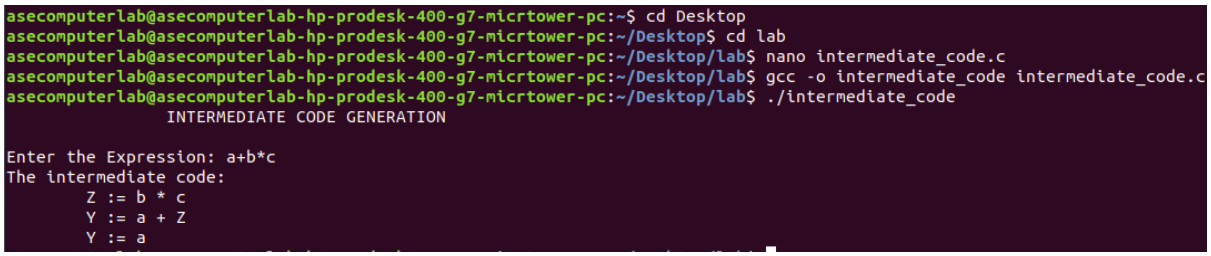
Code:

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

int i = 1, j = 0, no = 0, tmpch = 90;
char str[100], left[15], right[15];

struct exp {
    int pos;
    char op;
} k[15];

void findopr();
void explore();
void fleft(int);
void fright(int);

int main() {
    printf("\t\tINTERMEDIATE CODE GENERATION\n\n");
    printf("Enter the Expression: ");
    scanf("%s", str);
    printf("The intermediate code:\n");
    findopr();
    explore();
    return 0;
}
```

```
void explore() {
    i = 0;
    while (k[i].op != '\0') {
        fleft(k[i].pos);
        fright(k[i].pos);
        str[k[i].pos] = tmpch--;
        printf("\t%c := %s %c %s\n", str[k[i].pos], left, k[i].op, right);
        i++;
    }
    fright(-1);
    if (no == 0) {
        fleft(strlen(str));
        printf("\t%s := %s\n", right, left);
        exit(0);
    }
    printf("\t%s := %c\n", right, str[k[--i].pos]);
}

void fleft(int x) {
    int w = 0, flag = 0;
    x--;
    while (x != -1 && str[x] != '+' && str[x] != '*' && str[x] != '=' &&
           str[x] != '\0' && str[x] != '-' && str[x] != '/' && str[x] != ':') {
        if (str[x] != '$' && flag == 0) {
            left[w++] = str[x];
            left[w] = '\0';
            str[x] = '$';
            flag = 1;
        }
        x--;
    }
    // Reverse the left string because it is collected backwards
    int start = 0, end = w - 1;
    while (start < end) {
        char temp = left[start];
        left[start] = left[end];
        left[end] = temp;
        start++;
        end--;
    }
}
```

```
void fright(int x) {
    int w = 0, flag = 0;
    x++;
    while (x != -1 && str[x] != '+' && str[x] != '*' && str[x] != '\0' &&
           str[x] != '=' && str[x] != ':' && str[x] != '-' && str[x] != '/') {
        if (str[x] != '$' && flag == 0) {
            right[w++] = str[x];
            right[w] = '\0';
            str[x] = '$';
            flag = 1;
        }
        x++;
    }
}
```

Output:


asecomputerlab@asecomputerlab-hp-prodesk-400-g7-micrtower-pc:~$ cd Desktop
asecomputerlab@asecomputerlab-hp-prodesk-400-g7-micrtower-pc:~/Desktop$ cd lab
asecomputerlab@asecomputerlab-hp-prodesk-400-g7-micrtower-pc:~/Desktop/lab$ nano intermediate_code.c
asecomputerlab@asecomputerlab-hp-prodesk-400-g7-micrtower-pc:~/Desktop/lab$ gcc -o intermediate_code intermediate_code.c
asecomputerlab@asecomputerlab-hp-prodesk-400-g7-micrtower-pc:~/Desktop/lab$ ./intermediate_code
INTERMEDIATE CODE GENERATION
Enter the Expression: a+b*c
The intermediate code:
Z := b * c
Y := a + Z
Y := a