PROGRAM

```
#include<stdio.h>
#include<string.h>
int tempVarCount = 1;
void genTemp(char newTemp[])
{
    FILE *looker = fopen("temp.txt", "w+");
    fprintf(looker, "t%d", tempVarCount);
    fseek(looker, 0, 0);
    fscanf(looker, "%s", newTemp);
    remove("temp.txt");
}

void push(char stack[], char element)
{
     int len = strlen(stack);
     stack[len] = element;
     stack[len + 1] = '\0';
}
void pop(char stack[], char handle[])
{
     int stackLen, handleLen, i;
     stackLen = strlen(stack);
     handleLen = strlen(handle);
     for(i=stackLen-1; i>=stackLen-handleLen; i--)
          stack[i] = '\0';
}

int isOperator(char input)
{
    if((input == '+')||(input == '-')||(input == '*')||(input == '/')||(input == '(')||(input == ')')||(input
                                                                                                    == '$'))
        return(1);
    else
        return(0);
}

int getHandle(char stack[], char handle[])
{
     int i, opCount = 0, stackLen = strlen(stack);
     for(i=stackLen-1; i>=0; i--)
     {
          if(isOperator(stack[i]) == 1)
```

```
                opCount++;
            if(opCount == 2)
            {
                strcpy(handle, &stack[i+1]);
                  break;
            }
        }
        if(((opCount == 2)&&(handle[0] != '(')))
            return(1);
        else
            return(0);
}

void getSubHandle(char stack[], char handle[])
{
      int i, stackLen = strlen(stack);
      for(i=stackLen-1; i>=0; i--)
      {
            if(isOperator(stack[i]) == 1)
            {
                strcpy(handle, &stack[i+1]);
                  break;
            }
      }
}

void genExpression(char stack[], char handle[])
{
      char newTemp[5];
      genTemp(newTemp);
      printf("%s = %s\n", newTemp, handle);
      pop(stack, handle);
      strcat(stack, newTemp);
      tempVarCount++;
}

void interMedCodeGen(char inputString[])
{
      char stack[100], handle[20];
      strcpy(stack, "$");
      strcat(inputString, "$");
      while((stack[strlen(stack)-1] != '$')||(inputString[0] != '$'))
      {
```

```
        if((inputString[0] == '+')||(inputString[0] == '-'))
        {
            if(getHandle(stack, handle) == 1)
                genExpression(stack, handle);
            else
            {
                push(stack, inputString[0]);
                strcpy(inputString, &inputString[1]);
            }
        }
        else if((inputString[0] == ')')||(inputString[0] == '$'))
        {
            if(getHandle(stack, handle) == 1)
                genExpression(stack, handle);
            else
            {
                getSubHandle(stack, handle);
                if(inputString[0] == '$')
                    pop(stack, handle);
                else if(inputString[0] == ')')
                {
                    pop(stack, handle);
                    pop(stack, "(");
                    strcat(stack, handle);
                    strcpy(inputString, &inputString[1]);
                }
            }
        }
        else
        {
            push(stack, inputString[0]);
            strcpy(inputString, &inputString[1]);
        }
    }
}
void main()
{
    char inputString[100];
    printf("Enter the expression for intermediate-code generation...\n");
    scanf("%s", inputString);
    interMedCodeGen(inputString);
    getch();
}
```

INPUT/OUTPUT-

```
Enter the expression for intermediate-code generation...
(a-(b+(c/d-a+b/d-a/e)/f)+g)/(b+(a-(c*d+a/b)*f)*d)
t1 = c/d
t2 = t1-a
t3 = b/d
t4 = t2+t3
t5 = a/e
t6 = t4-t5
t7 = t6/f
t8 = b+t7
t9 = a-t8
t10 = t9+g
t11 = c*d
t12 = a/b
t13 = t11+t12
t14 = t13*f
t15 = a-t14
t16 = t15*d
t17 = b+t16
t18 = t10/t17
```

```
Enter the expression for intermediate-code generation...
(23+(2+(4/3-5/6)*95)-45)/(45-(34+((4-5)+(5+6))/3)+7)
t1 = 4/3
t2 = 5/6
t3 = t1-t2
t4 = t3*95
t5 = 2+t4
t6 = 23+t5
t7 = t6-45
t8 = 4-5
t9 = 5+6
t10 = t8+t9
t11 = t10/3
t12 = 34+t11
t13 = 45-t12
t14 = t13+7
t15 = t7/t14
```

RESULT-