

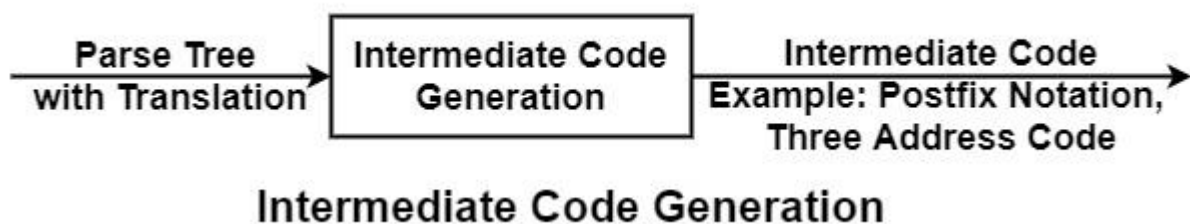
## Experiment No. 10

**Aim:** To implement intermediate code generator.

**Requirements:** Compatible version of Java.

### **Theory:**

Intermediate code can translate the source program into the machine program. Intermediate code is generated because the compiler can't generate machine code directly in one pass. Therefore, first, it converts the source program into intermediate code, which performs efficient generation of machine code further. The intermediate code can be represented in the form of postfix notation, syntax tree, directed acyclic graph, three address codes, Quadruples, and triples.



### **Example of Intermediate Code Generation –**

- **Three Address Code**– These are statements of form  $c = a \text{ op } b$ , i.e., in which there will be at most three addresses, i.e., two for operands & one for Result. Each instruction has at most one operator on the right-hand side.

Example of three address code for the statement



## Advantages of Intermediate Code Generation

- It is Machine Independent. It can be executed on different platforms.
- It creates the function of code optimization easy. A machine-independent code optimizer can be used to intermediate code to optimize code generation.
- It can perform efficient code generation.
- From the existing front end, a new compiler for a given back end can be generated.
- Syntax-directed translation implements the intermediate code generation, thus by augmenting the parser, it can be folded into the parsing

### Code:

```
import java.util.*;
import java.io.*;

class CodeGeneration {

    static Vector<String> threeadd = new Vector<>();
    static char tempVar='z';
    static int regcounter=0;
    static Map<String,Integer> reg = new HashMap<>();
    static Vector<String> machinecode = new Vector<>();

    public static void main(String []args) {

        String input;int i;
        Scanner sc = new Scanner(System.in);

        System.out.println("Enter expression: ");
        input = sc.nextLine();

        generateThreeAddressCode(input);

        System.out.println();
        System.out.println("Three Address Code: ");
        for(i=0;i<threeadd.size();i++) System.out.println(threeadd.get(i));

        generateMachineCode();

        System.out.println();
        System.out.println("Machine Code: ");
```

```

    for(i=0;i<machinecode.size();i++) System.out.println(machinecode.get(i));
}

public static void generateThreeAddressCode(String s) {

    String after="";
    if(s.contains("=") && s.length()>=3) {
        threeadd.add(s);
        return;
    }
    else if(s=="") return;

    if(s.contains("(")) {
        int start = s.indexOf("(");
        int end = s.indexOf(")");
        String temp = tempVar+"="+s.substring(start+1,end);
        after = s.substring(0,start)+tempVar+s.substring(end+1);
        tempVar--;
        threeadd.add(temp);
    }
    else if(s.contains("/")) {
        int c = s.indexOf("/");
        String temp = tempVar+"="+s.substring(c-1,c)+"/"+s.substring(c+1,c+2);
        after = s.substring(0,c-1)+tempVar+s.substring(c+2);
        tempVar--;
        threeadd.add(temp);
    }
    else if(s.contains("*")) {
        int c = s.indexOf("*");
        String temp = tempVar+"="+s.substring(c-1,c)+"*"+s.substring(c+1,c+2);
        after = s.substring(0,c-1)+tempVar+s.substring(c+2);
        tempVar--;
        threeadd.add(temp);
    }
    else if(s.contains("+")) {
        int c = s.indexOf("+");
        String temp = tempVar+"="+s.substring(c-1,c)+" "+s.substring(c+1,c+2);
        after = s.substring(0,c-1)+tempVar+s.substring(c+2);
        tempVar--;
        threeadd.add(temp);
    }
    else if(s.contains("-")) {
        int c = s.indexOf("-");
        String temp = tempVar+"="+s.substring(c-1,c)+"-"+s.substring(c+1,c+2);
        after = s.substring(0,c-1)+tempVar+s.substring(c+2);
        tempVar--;
        threeadd.add(temp);
    }
}

```

```

    }
    //System.out.println(after);
    generateThreeAddressCode(after);
}

```

```

public static void generateMachineCode() {

    int i,p=0,f=0;
    for(i=0;i<threeadd.size();i++) {
        String s = threeadd.get(i);
        String temp="";

        if(s.contains("+")) p = s.indexOf("+");
        else if(s.contains("-")) p = s.indexOf("-");
        else if(s.contains("*")) p = s.indexOf("*");
        else if(s.contains("/")) p = s.indexOf("/");
        else if(s.contains("=")) {
            p = s.indexOf("=");
            temp = "MOV "+reg.get(s.substring(p+1))+","+s.substring(0,1);
            machinecode.add(temp);
            temp="";
            f=1;
            continue;
        }

        if(reg.containsKey(s.substring(p-1,p)) == false) {
            reg.put(s.substring(p-1,p),regcounter);
            temp = "MOV "+s.substring(p-1,p)+","+regcounter;
            machinecode.add(temp);
            regcounter++;
        }
        //System.out.println(temp);
        temp="";
        if(reg.containsKey(s.substring(p+1,p+2)) == false) {
            reg.put(s.substring(p+1,p+2),regcounter);
            temp = "MOV "+s.substring(p+1,p+2)+","+regcounter;
            machinecode.add(temp);
            regcounter++;
        }
        //System.out.println(temp);
        temp="";

        if(s.contains("+")) temp = "ADD ";
        else if(s.contains("-")) temp = "SUB ";
        else if(s.contains("*")) temp = "MUL ";
        else if(s.contains("/")) temp = "DIV ";

        temp += reg.get(s.substring(p-1,p))+","+reg.get(s.substring(p+1,p+2));
        int r= reg.get(s.substring(p-1,p));
        reg.remove(s.substring(p-1,p));
    }
}

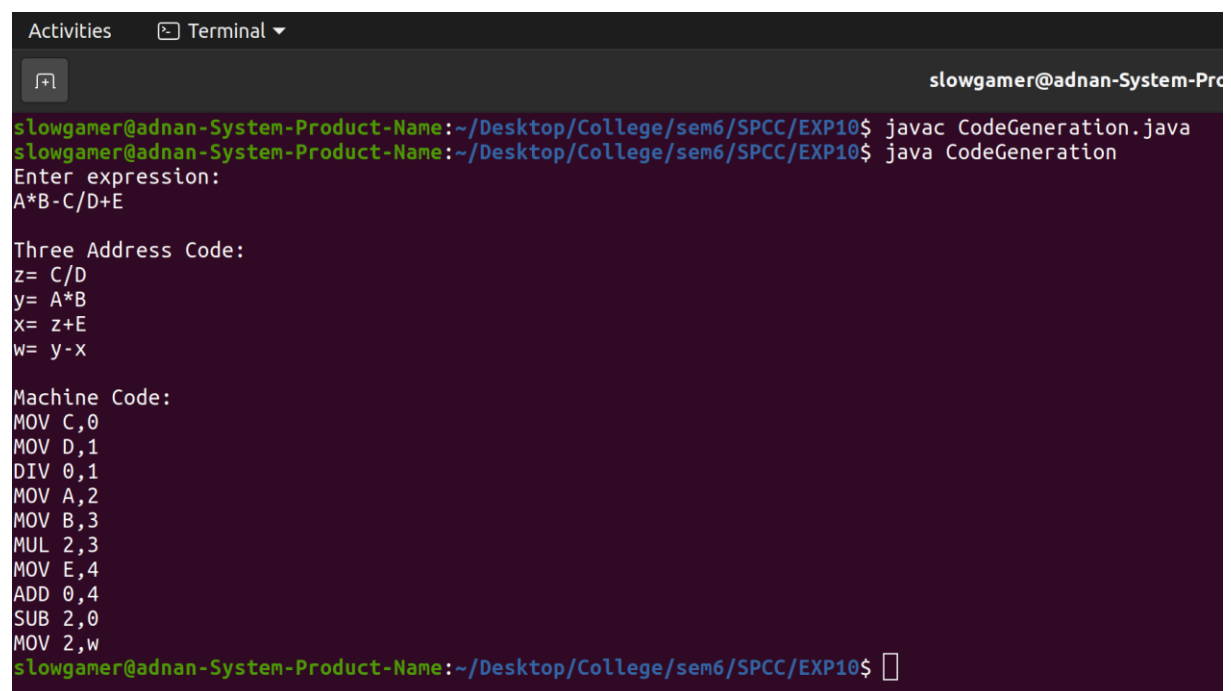
```

```

        reg.put(s.substring(0,1),r);
        machinecode.add(temp);
        //System.out.println(temp);
    }
    if(f==0) {
        String temp="",s=threeadd.get(i-1);
        temp = "MOV "+reg.get(s.substring(0,1))+","+s.substring(0,1);
        machinecode.add(temp);
    }
}
}
}

```

### **Output:**



```

slowgamer@adnan-System-Product-Name:~/Desktop/College/sem6/SPCC/EXP10$ javac CodeGenration.java
slowgamer@adnan-System-Product-Name:~/Desktop/College/sem6/SPCC/EXP10$ java CodeGeneration
Enter expression:
A*B-C/D+E

Three Address Code:
z= C/D
y= A*B
x= z+E
w= y-x

Machine Code:
MOV C,0
MOV D,1
DIV 0,1
MOV A,2
MOV B,3
MUL 2,3
MOV E,4
ADD 0,4
SUB 2,0
MOV 2,w
slowgamer@adnan-System-Product-Name:~/Desktop/College/sem6/SPCC/EXP10$ 

```

**Conclusion:** We have successfully implemented intermediate code generation program for arithmetic expression in JAVA.