EXPERIMENT 11

Intermediate code generation – Quadruple, Triple, Indirect triple

Name : Rahul Goel
Reg. No : RA1911030010094

Aim: Intermediate code generation – Quadruple, Triple, Indirect triple

Algorithm:-

The algorithm takes a sequence of three-address statements as input. For each three address statements of the form a:= b op c perform the various actions. These are as follows: 1. Invoke a function getreg to find out the location L where the result of computation b op c should be stored.

2. Consult the address description for y to determine y'. If the value of y currently in memoryand register both then prefer the register y' . If the value of y is not already in L then generate the instruction MOV y' , L to place a copy of y in L.
3. Generate the instruction OP z' , L where z' is used to show the current location of z. if z is in both then prefer a register to a memory location. Update the address descriptor of x to indicate that x is in location L. If x is in L then update its descriptor and remove x from allother descriptors.
4. If the current value of y or z have no next uses or not live on exit from the block or in register then alter the register descriptor to indicate that after execution of x : = y op z those register will no longer contain y or z.

Code:

```
#include<iostream
>#include<stdio.h>
#include<ctype.h>
#include<stdlib.h>
#include<string.h>
void small();
```

```
void dove(int i);
```

```
int p[5]={0,1,2,3,4},c=1,i,k,l,m,pi;
char sw[5]={'=','-','+','/','*'},j[20],a[5],b[5],ch[2];
int main()
{
printf("Enter the expression:");
scanf("%s",j);
printf("\tThe Intermediate code is:\n");
small();
return 0;
}
void dove(int i)
{ a[0]=b[0]='\
0';
if(!isdigit(j[i+2])&&!isdigit(j[i-2]))
{
a[0]=j[i-1];
b[0]=j[i+1];
}
if(isdigit(j[i+2])){
a[0]=j[i-1];
b[0]='t';
b[1]=j[i+2];
}
if(isdigit(j[i-2]))
{ b[0]=j[i+
1];
a[0]='t';
a[1]=j[i-2];
b[1]='\0';
}
if(isdigit(j[i+2]) &&isdigit(j[i-2]))
{ a[0]='
t';
b[0]='t';
a[1]=j[i-2];
b[1]=j[i+2];
sprintf(ch,"%d",c);
j[i+2]=j[i-2]=ch[0];
}
if(j[i]=='*')
printf("\tt%d=%s*%s\n",c,a,b);
if(j[i]=='/') printf("\tt%d=%s/
```

```
%s\n",c,a,b);if(j[i]=='+')
```

```
printf("\tt%d=%s+
%s\n",c,a,b);if(j[i]=='-')printf("\tt%d=%s-
%s\n",c,a,b); if(j[i]=='=')
printf("\t%c=t%d",j[i-1],--c);
sprintf(ch,"%d",c);
j[i]=ch[0]
;c++;
small();
}
void small()
{
pi=0;l=0;
for(i=0;i<strlen(j);i++)
{
for(m=0;m<5;m+
+)if(j[i]==sw[m])
if(pi<=p[m])
{
pi=p[m]
;l=1;
k=i;
}
}
if(l==1)
dove(k)
;else
exit(0);}
```

Output:-

main.cpp

```cpp
#include<iostream>
#include<stdio.h>
#include<ctype.h>
#include<stdlib.h>
#include<string.h>
void small();
void dove(int i);

int p[5]={0,1,2,3,4},c=1,i,k,l,m,pi;
char sw[5]={'=','-','+','/','*'},j[20],a[5],b[5],ch[2];
int main()
{
printf("Enter the expression:"); scanf("%s",j);
printf("\tThe Intermediate code is:\n"); small();
return 0;
}
void dove(int i)
{ a[0]=b[0]='\0';
if(!isdigit(j[i+2])&&!isdigit(j[i-2]))
{
a[0]=j[i-1];
b[0]=j[i+1];
}
if(isdigit(j[i+2])){
```

input

```
Enter the expression:a=b-c*d
        The Intermediate code is:
        t1=c*d
        t2=b-t1
        a=t2

...Program finished with exit code 0
```

Result:-

The program was successfully compiled and run.