Compiler Construction

Roll No :18bce144

Practical-10

Objective

- To implement Code Optimization techniques.

CODE

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
struct op
{ char l;
char r[20];
}
op[10],pr[10];
int main()
{ int a,i,k,j,n,z=0,m,q;
char *p,*l;
char temp,t;
char *tem;
printf("Enter the Number of Values:");
```

```
scanf("%d",&n);
for(i=0;i<n;i++)
{
printf("left: ");
op[i].l=getche();
printf("\tright: ");
scanf("%s",op[i].r);
}
printf("Intermediate Code\n") ;
for(i=0;i<n;i++)
{
printf("%c=",op[i].l);
printf("%s\n",op[i].r);
}
for(i=0;i<n-1;i++)
{ temp=op[i].l;
for(j=0;j<n;j++)
{
p=strchr(op[j].r,temp);
if(p)
{
pr[z].l=op[i].l;
```

```
strcpy(pr[z].r,op[i].r);
z++; }}}
pr[z].l=op[n-1].l;
strcpy(pr[z].r,op[n-1].r);
z++;
printf("After Dead Code Elimination\n");
for(k=0;k<z;k++) {
printf("%c\t=",pr[k].l);
printf("%s\n",pr[k].r);
}
for(m=0;m<z;m++) {
tem=pr[m].r;
for(j=m+1;j<z;j++) {
p=strstr(tem,pr[j].r);
if(p) {
t=pr[j].l;
pr[j].l=pr[m].l;
for(i=0;i<z;i++) {
l=strchr(pr[i].r,t) ;
if(l) {
a=l-pr[i].r;
printf("pos: %d\n",a);
```

```
pr[i].r[a]=pr[m].l; }}}}}
printf("Eliminate Common Expression\n");
for(i=0;i<z;i++)
{
printf("%c\t=",pr[i].l);
printf("%s\n",pr[i].r);
}
for(i=0;i<z;i++)
{
for(j=i+1;j<z;j++)
{
q=strcmp(pr[i].r,pr[j].r);
if((pr[i].l==pr[j].l)&&!q)
{
pr[i].l='\0';
strcpy(pr[i].r,'\0');
}}}
printf("Optimized Code\n");
for(i=0;i<z;i++)
{ if(pr[i].l!='\0')
{
printf("%c=",pr[i].l);
```

```
printf("%s\n",pr[i].r);
}
}
getch();
return 0;
}
```

Output file:

```
Enter the Number of Values:4
left: t right: a+b
left: u right: t*c
left: v right: u/f
left: j right: v-k
Intermediate Code
t=a+b
u=t*c
v=u/f
j=v-k
After Dead Code Elimination
t       =a+b
u       =t*c
v       =u/f
j       =v-k
Eliminate Common Expression
t       =a+b
u       =t*c
v       =u/f
j       =v-k
Optimized Code
t=a+b
u=t*c
v=u/f
j=v-k
```