Program

```
#include<stdio.h>
#define MAX 20
#include<string.h>
#include<stdlib.h>

struct symbol
{
	char sym[10];
	int addr;
}S[MAX];

struct litab
{
	char lit[10];
	int addr;
}L[MAX];

void print_file(char *);
void print_symtab();
void print_littab();

char optab[]
[6]={"STOP","ADD","SUB","MULT","MOVER","MOVEM","COMP","BC","DIV","READ","PRIN
T"};
char regtab[][5]={"AREG","BREG","CREG","DREG"};
char adtab[][7]={"START","END","ORIGIN","EQU","LTORG"};
char condtab[][4]={"LT","LE","EQ","GT","GE","ANY"};

FILE *fs,*ft;

char buffer[80],source[80],tok1[10],tok2[10],tok3[10],tok4[10],tok5[10];
int lc=0,sc=0,poolcnt=0,litcnt=0;
int pooltab[10];

int search_optab(char *s)
{
	int i;
	for(i=0;i<11;i++)
	{
		if(strcmp(optab[i],s)==0)
		{
			return i;
		}
	}
	return -1;
}
```

```
int search_regtab(char *s)
{
	int i;
	for(i=0;i<4;i++)
	{
		if(strcmp(regtab[i],s)==0)
		{
			return i+1;
		}
	}
	return -1;
}

int search_condtab(char *s)
{
	int i;
	for(i=0;i<6;i++)
	{
		if(strcmp(condtab[i],s)==0)
		{
			return (i+1);
		}
	}
	return -1;
}

int search_adtab(char *s)
{
	int i;
	for(i=0;i<5;i++)
	{
		if(strcmp(adtab[i],s)==0)
		{
			return i+1;
		}
	}
	return -1;
}

int search_symtab(char *s)
{
	int i;
	for(i=0;i<sc;i++)
	{
		if(strcmp(S[i].sym,s)==0)
		{
			return i;
		}
```

```
        }
        return -1;
}

int search_littab(char *s)
{
        int i;
        for(i=pooltab[poolcnt];i<litcnt;i++)
        {
                if(strcmp(L[i].lit,s)==0)
                {
                        return i;
                }
        }
        return -1;
}

void pass1()
{
        int p;
        int n,i=0,j=0,k=0,o=0,f=0;

   char h[20],m[20];
        fs=fopen(source,"r");
        if(fs==NULL)
        {
                printf("\n File doesn't Exists ... \n");
                exit(0);
        }

        ft=fopen("id.txt","w");
        while(fgets(buffer,80,fs))
        {
                n=sscanf(buffer,"%s%s%s%s",tok1,tok2,tok3,tok4);
                switch(n)
                {
                        case 1: //ltorg,end
                                        i=search_adtab(tok1);
                                        if(i==2)
                                        {
                                           fprintf(ft," (AD, %02d)\n",i);
                                                for(j=pooltab[poolcnt];j<litcnt;j++)
                                                {
                                                        L[j].addr=lc++;
                                                        fprintf(ft," (DL, 01) \t (C, 0%c)\n",L[j].lit[2]);
                                                }
                                                lc--;
                                                pooltab[++poolcnt]=litcnt;
                                                break;
```

```
                }

                if(i==5)
                {
                        for(j=pooltab[poolcnt];j<litcnt;j++)
                        {
                                L[j].addr=lc++;
                                fprintf(ft," (DL, 01) \t (C, 0%c)\n",L[j].lit[2]);
                        }
                        lc--;
                        pooltab[++poolcnt]=litcnt;
                        break;
                }

case 2: //start
                i=search_adtab(tok1);
                if(i==1)
                {
                        lc=atoi(tok2)-1;
                        fprintf(ft," (AD, %02d) \t (C, 0%s)\n",i,tok2);
                        break;
                }

                //ORIGIN A+10
                if(i==3)
                {
                        f=0;
                        h[0]=tok2[0];
                        h[1]='\0';
                        p=search_symtab(h);

                        for(o=2;o<strlen(tok2);o++)
                                m[f++]=tok2[o];
                        m[f]='\0';

                        switch(tok2[1])
                        {
                                case '+':  lc=S[p].addr+atoi(m)-1;
                                                printf("%d",lc);
                                                break;
                                case '-': lc=S[p].addr-atoi(m)-1;
                        }
                }

case 3: i=search_optab(tok1);
                if(i>=1 && i<=8)
                {
                        tok2[strlen(tok2)-1]='\0';
                        k=search_regtab(tok2);
```

```
                                        //mover areg,='5'
                                        if(tok3[0]=='=')
                                        {
                                                j=search_littab(tok3);
                                                if(j==-1)
                                                {
                                                        strcpy(L[litcnt].lit,tok3);
                                                        fprintf(ft," (IS, %02d) \t 0%d \t (L,
%02d)\n",i,k,litcnt);
                                                        litcnt++;
                                                }
                                                else
                                                {
                                                        fprintf(ft," (IS, %02d) \t 0%d \t (L,
%02d)\n",i,k,j);
                                                }
                                                break;
                                        }
                                        else//mover areg,A
                                        {
                                                p=search_symtab(tok3);
                                                if(p==-1)
                                                {
                                                        strcpy(S[sc].sym,tok3);
                                                        fprintf(ft," (IS, %02d) \t 0%d \t (S,
%02d)\n",i,k,sc);
                                                        sc++;
                                                }
                                                else
                                                {
                                                        fprintf(ft," (IS, %02d) \t 0%d \t (S,
%02d)\n",i,k,p);
                                                }
                                                break;
                                        }
                                }

                                //A DS 2
                                if(strcmp(tok2,"DS")==0)
                                {
                                        p=search_symtab(tok1);
                                        if(p==-1)
                                        {
                                                strcpy(S[sc].sym,tok1);
                                                S[sc].addr=lc;
                                                fprintf(ft," (DL, 02) \t (C, 0%s)\n",tok3);
                                                sc++;
                                        }
```

```
		else
		{
			S[p].addr=lc;
			fprintf(ft," (DL, 02) \t (C, 0%s)\n",tok3);
		}
		lc=lc+atoi(tok3)-1;
		break;
}

// D EQU A-1
i=search_adtab(tok2);
if(i==4)
{
		p=search_symtab(tok1);
		if(p==-1)
		{
			f=0;
			strcpy(S[sc].sym,tok1);
			h[0]=tok3[0];
			h[1]='\0';
			p=search_symtab(h);

			for(o=2;o<strlen(tok3);o++)
				m[f++]=tok3[o];
			m[f]='\0';

			switch(tok3[1])
			{
				case '+': S[sc].addr=S[p].addr+atoi(m);
						break;
				case '-': S[sc].addr=S[p].addr-atoi(m);
			}
			fprintf(ft," (AD, 04) \t (C, 0%d)\n",S[sc].addr);
			sc++;
		}
		else
		{
			strcpy(S[sc].sym,tok1);
			p=search_symtab(tok3);
			h[0]=tok3[0];
			h[1]='\0';
			p=search_symtab(h);

			for(o=2;o<strlen(tok3);o++)
				m[f++]=tok3[o];
			m[f]='\0';

			switch(tok3[1])
			{
```

```
                                                case '+': S[sc].addr=S[p].addr+atoi(m);
                                                          break;
                                                case '-': S[sc].addr=S[p].addr-atoi(m);
                                        }
                                        fprintf(ft," (AD, 04) \t (C, 0%d)\n",S[sc].addr);
                                }
                        }
                }
                lc++;
        }
        fcloseall();
}

void print_file(char *target)
{
        FILE *fp;
        fp=fopen(target,"r");

        if(fp==NULL)
        {
                printf("\n File doen't Exists ... \n");
                exit(0);
        }

        printf("\n");
        while(fgets(buffer,80,fp))
        {
                printf("%s",buffer);
        }
        fclose(fp);
}

void print_littab()
{
        int i;
        printf("\n LITERAL \t ADDRESS\n");

        for(i=0;i<litcnt;i++)
        {
                printf("  %s  \t   %d\n",L[i].lit,L[i].addr);
        }
}

void print_symtab()
{
        int p=0;
        printf("\n SYMBOL \t ADDRESS\n");

        while(p<sc)
```

```
        {
                printf("   %s    \t   %d\n",S[p].sym,S[p].addr);
                p=p+1;
        }
}

void main()
{
        printf("\n Enter Source File Name:  ");
        scanf("%s",source);

        printf("\n Using Variant - I \n");
        printf("\n ** Source Code ** \n");
        print_file(source);

        pass1();

        printf("\n\n ** Literal Table ** ");
        print_littab();

        printf("\n\n ** Symbol Table ** ");
        print_symtab();

        printf("\n\n ** Intermediate Code ** ");
        print_file("id.txt");
        printf("\n");
}
```

Output

```
abc@ubuntu:~/SEM2/SP$ gcc -o out p1.c -w
abc@ubuntu:~/SEM2/SP$ ./out

 Enter Source File Name:  source2

 Using Variant - I

 ** Source Code **

     START 200
MOVER AREG, ='5'
MOVEM BREG, ='1'
  ADD BREG, A
            LTORG
  SUB CREG, B
    A DS 2
      ORIGIN A+10
    B DS 2
    D EQU A+10
```

```
        END
215

 ** Literal Table **
 LITERAL      ADDRESS
  ='5'      203
  ='1'      204


 ** Symbol Table **
 SYMBOL      ADDRESS
   A         206
   B         216
   D         216


 ** Intermediate Code **
 (AD, 01)        (C, 0200)
 (IS, 04)        01      (L, 00)
 (IS, 05)        02      (L, 01)
 (IS, 01)        02      (S, 00)
 (DL, 01)        (C, 05)
 (DL, 01)        (C, 01)
 (IS, 02)        03      (S, 01)
 (DL, 02)        (C, 02)
 (DL, 02)        (C, 02)
 (AD, 04)        (C, 0216)
 (AD, 02)

abc@ubuntu:~/SEM2/SP$
```