/\* Assembler PASS-II \*/

#include<stdio.h>

#include<string.h>

#define MAX 20

struct sym\_tab

{

char sym[MAX];

int adr,l;

}s[MAX];

int sc,loc;

struct lit\_tab

{

char lit[MAX];

int adr;

}l[MAX];

int lc;

int pt[MAX]={0},pc;

struct mne\_op\_tab

{

char mne[MAX],clas[MAX];

int op,l;

}m[MAX]={{"START","AD",1,0},

{"END","AD",2,0},

{"ORIGIN","AD",3,0},

{"EQU","AD",4,0},

{"LTORG","AD",5,0},

{"DC","DL",1,0},

{"DS","DL",2,0},

{"STOP","IS",0,1},

{"ADD","IS",1,1},

{"SUB","IS",2,1},

{"MULT","IS",3,2},

{"MOVER","IS",4,1},

{"MOVEM","IS",5,1},

{"BC","IS",6,1},

{"DIV","IS",7,2},

{"READ","IS",8,1},

{"PRINT","IS",9,1},

};

struct regs

{

char reg[MAX];

int val;

}r[4]={{"AREG",1},

{"BREG",2},

{"CREG",3},

{"DREG",4}};

void disp(char fname[MAX])

{

char s[80];

FILE \*fp;

fp=fopen(fname,"r");

while(fgets(s,80,fp))

printf("%s",s);

fcloseall();

}

void pass1()

{

FILE \*fp1,\*fp2;

int i,j,k=0,n;

char s1[80],w1[80],w2[80],w3[80],w4[80],str[80],temp[80];

fp1=fopen("a1.txt","r");

fp2=fopen("b.txt","w");

while(fgets(s1,80,fp1))

{

n=sscanf(s1,"%s%s%s%s",w1,w2,w3,w4);

if(n==4)

{

strcpy(s[sc].sym,w1);

s[sc].adr=loc;

s[sc++].l=1;

n--;

strcpy(w1,w2);

strcpy(w2,w3);

strcpy(w3,w4);

}

if(n==3)

{

if(strcmp(w2,"EQU")==0)

{

strcpy(s[sc].sym,w1);

for(i=0;i<sc;i++)

if(strcmp(w3,s[i].sym)==0)

break;

s[sc].adr=s[i].adr;

s[sc].l=1;

for(i=0;i<MAX;i++)

if(strcmp(w2,m[i].mne)==0)

break;

sprintf(str,"%d (%s,%d) - (S,%d)\n",loc,m[i].clas,m[i].op,sc);

sc++;

}

else if(strcmp(w2,"DS")==0 || strcmp(w2,"DC")==0)

{

for(i=0;i<sc;i++)

if(strcmp(w1,s[i].sym)==0)

break;

s[i].adr=loc;

for(j=0;j<MAX;j++)

if(strcmp(w2,m[j].mne)==0)

break;

sprintf(str,"%d (%s,%d) - (C,%s)\n",loc,m[j].clas,m[j].op,w3);

if(strcmp(w2,"DS")==0)

{

s[i].l=atoi(w3);

loc+=atoi(w3);

}

else

{

s[i].l=1;

loc+=1;

}

}

else

{

for(i=0;i<MAX;i++)

if(strcmp(w1,m[i].mne)==0)

break;

w2[4]='\0';

for(j=0;j<4;j++)

if(strcmp(w2,r[j].reg)==0)

break;

sprintf(str,"%d (%s,%d) %d ",loc,m[i].clas,m[i].op,r[j].val);

loc+=m[i].l;

if(w3[0]=='=')

{

for(i=pt[pc];i<lc;i++)

if(strcmp(w3,l[i].lit)==0)

break;

if(i==lc)

{

strcpy(l[lc].lit,w3);

sprintf(temp,"(L,%d)\n",lc-pt[pc]);

lc++;

}

else

{

sprintf(temp,"(L,%d)\n",i-pt[pc]);

}

}

else

{

for(i=0;i<sc;i++)

if(strcmp(w3,s[i].sym)==0)

break;

if(i==sc)

{

strcpy(s[sc].sym,w3);

sprintf(temp,"(S,%d)\n",sc);

sc++;

}

else

sprintf(temp,"(S,%d)\n",i);

}

strcat(str,temp);

}

}

if(n==2)

{

if(strcmp(w1,"START")==0)

{

for(i=0;i<MAX;i++)

if(strcmp(m[i].mne,w1)==0)

break;

sprintf(str,"%d (%s,%d) - (C,%s)\n",loc,m[i].clas,m[i].op,w2);

loc=atoi(w2);

}

if(strcmp(w1,"ORIGIN")==0)

{

for(i=0;i<MAX;i++)

if(strcmp(m[i].mne,w1)==0)

break;

j=k=0;

sprintf(str,"%d (%s,%d) - ",loc,m[i].clas,m[i].op);

while(!ispunct(w2[j]))

temp[k++]=w2[j++];

temp[k]='\0';

for(k=0;k<sc;k++)

if(strcmp(temp,s[k].sym)==0)

break;

loc=s[k].adr;

if(w2[j]=='+')

{

k=0;

while(w2[j]!='\0')

temp[k++]=w2[j++];

temp[k]='\0';

loc+=atoi(temp);

}

if(w2[j]=='-')

{

k=0;

while(w2[j]!='\0')

temp[k++]=w2[j++];

temp[k]='\0';

loc+=atoi(temp);

}

sprintf(temp,"(C,%d)\n",loc);

strcat(str,temp);

}

}

if(n==1)

{

if(strcmp(w1,"LTORG")==0 || strcmp(w1,"END")==0)

{

for(i=0;i<MAX;i++)

if(strcmp(m[i].mne,w1)==0)

break;

sprintf(str,"%d (%s,%d) - ",loc,m[i].clas,m[i].op);

for(i=pt[pc];i<lc;i++)

{

l[i].adr=loc;

sprintf(temp,"%s\n",l[i].lit);

strcat(str,temp);

fputs(str,fp2);

loc++;

sprintf(str,"%d - - ",loc);

}

if(strcmp(w1,"LTORG")==0)

pt[++pc]=lc;

}

continue;

}

fputs(str,fp2);

}

fcloseall();

}

void show\_tab()

{

int i;

printf("\n------------SYMTAB------------------\n");

printf("Symbol\tAddress\tLength\n");

for(i=0;i<sc;i++)

printf("%s\t%d\t%d\n",s[i].sym,s[i].adr,s[i].l);

printf("\n------------LITTAB------------------\n");

printf("Literal\tAddress\n");

for(i=0;i<lc;i++)

printf("%s\t%d\n",l[i].lit,l[i].adr);

printf("\n------------POOLTAB------------------\n");

printf("LITTAB\_PTR\n");

for(i=0;i<=pc;i++)

printf("%d\n",pt[i]);

}

int main()

{

disp("a1.txt");

pass1();

show\_tab();

printf("\nIntermediate Code:\n");

disp("b.txt");

return 1;

}