**Assembler pass 1**

#include <stdio.h>

#include<stdlib.h>

#include<string.h>

FILE \*fr;

FILE \*fw;

int main ()

{

int lc;

char buff[10];

int count, scount;

count=0; scount=0;

char lit[10], sym[10];

int litadd[10];

int i,flag=0,index;

fr = fopen ("input.txt", "r");

fw = fopen("ic.txt", "w");

if (fr < 0 || fw < 0)

{

printf ("Error while opening a file");

exit (0);

}

fscanf (fr, "%s", buff); //start will be read

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

{

fprintf(fw,"(AD,01)");

fscanf(fr,"%d",&lc); //100 will be read

fprintf(fw,"(C,%d)\n",lc); // (C,100)

NEXT:

fscanf (fr, "%s", buff); //ADD willl be read or END

fprintf(fw,"%d ",lc); // To print the address

if(strcmp(buff,"END")==0)

{

fprintf(fw,"(AD,02)");

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

{

litadd[i]=lc++;

}

//exit(0);

goto last;

}

if(strcmp(buff,"ADD")==0)

{

fprintf(fw,"(IS,01)");

}

if(strcmp(buff,"SUB")==0)

{

fprintf(fw,"(IS,02)");

}

if(strcmp(buff,"MULT")==0)

{

fprintf(fw,"(IS,03)");

}

if(strcmp(buff,"DIV")==0)

{

fprintf(fw,"(IS,04)");

}

fscanf (fr, "%s", buff); //to scan register AREG, BREG, CREG, DREG,

if(strcmp(buff,"AREG,")==0)

{

fprintf(fw," 1 ");

}

if(strcmp(buff,"BREG,")==0)

{

fprintf(fw," 2 ");

}

if(strcmp(buff,"CREG,")==0)

{

fprintf(fw," 3 ");

}

if(strcmp(buff,"DREG,")==0)

{

fprintf(fw," 4 ");

}

fscanf (fr, "%s", buff); //TO SCAN SECOND OPERAND AS CONSTANT ' ', literal =' '

if(buff[0]=='\'')

{

fprintf(fw,"(C,%s)\n",buff); // (C,'5')

}

else if(buff[0]=='=')

{

flag=0;

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

{

if(buff[2]==lit[i])

{

flag=1;

index=i;

break;

}

}

if(flag==0) // if new literal is found

{

fprintf(fw,"(L,%d)\n",count);

lit[count]=buff[2]; // =''

count++;

}

else // already present

{

fprintf(fw,"(L,%d)\n",index);

}

}

else

{

flag=0;

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

{

if(buff[0]==sym[i])

{

flag=1;

index=i;

break;

}

}

if(flag==0) // if new literal is found

{

fprintf(fw,"(S,%d)\n",scount);

sym[scount]=buff[0]; // =''

scount++;

}

else // already present

{

fprintf(fw,"(S,%d)\n",index);

}

}

lc++;

goto NEXT;

}

else

{

printf("Invlaid Assembly Program\n");

}

last:

printf("Literal Table is\n");

printf("SN\t Name\t Address\n");

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

{

printf("%d\t%c\t%d\n",i,lit[i],litadd[i]);

}

return 0;

}





