**ASSIGNMENT: 3**

**Problem Statement:**

Write a recursiive program to implement Macro Processor. (AIF andAGO not expected, nested macro calls not expected) with Error Handling: Macro Duplicate Definition, Parameter Mismatching etc.

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

struct MNT

{

char mname[20];

int mdtp;

int pp,kpp,dpp;

}mnt[50];

struct MDT

{

char opcode[15],rest[35];

}mdt[30];

char arglist[15][15],apt[10][15];

int mdtp=0,mntp=0,arglistp=0;

char FName[20], TName[20];

char Buffer[80], temp[40],tok1[40];

int pp,kpp,dpp; **// no. of positional and keyword parameters**

FILE \*fp1, \*fp2;

void Print\_MNT() **//prints macro name table**

{ int i;

printf("\n\n----------MACRO NAME TABLE---------------------");

printf("\n#\tMName\t#MDTP\t#pp\t#kpp");

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

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

printf("\n%d\t%s\t%d\t%d\t%d",

i,mnt[i].mname,mnt[i].mdtp,mnt[i].pp,mnt[i].kpp);

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

}

void Print\_PNT() **//prints parameter name table**

{ int i;

printf("\n\n------PARAMETER NAME TABLE--------");

printf("\n#\tPName");

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

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

printf("\n%d\t%s\t%s",i,arglist[i],apt[i]);

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

}

void Print\_MDT() **//prints macro definition table**

{ int i;

printf("\n\n-----------MACRO DEFINITION TABLE--------------");

printf("\nOpcode\tRest");

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

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

printf("\n%d\t%s\t%s",i,mdt[i].opcode,mdt[i].rest);

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

}

int SearchMNT(char \*s) **//To search macro name**

{ int i;

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

if(strcmp(s,mnt[i].mname)==0)

return(i);

return(-1);

}

int SearchPNT(char \*s) **//To search parameter**

{ int i;

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

if(strcmp(arglist[i],s)==0)

return(i);

return(-1);

}

char \* nexttoken(char \*str , char \*token)

{

**//Separating tokens**

int i;

while(\*str==' ')

str++;

if(\*str==',' || \*str=='=')

{

\*token=\*str;

token++;

str++;

\*token='\0';

return(str);

}

while(isalnum(\*str) || \*str=='#' || \*str=='&')

{

\*token=\*str;

token++;

str++;

}

\*token='\0';

return(str);

}

void make\_arglist(char \*s) **//creates argument list**

{

int k;

pp=kpp=dpp=0; **//no. of postional and keyword parameters**

arglistp=0;

while(\*s)

{

s=nexttoken(s,temp);

k = SearchPNT(temp+1);

if(k==-1)

strcpy(arglist[arglistp++],temp+1);

else

{

printf("\nError: Multiple Declaration of Symbol %s in Argument List",temp);

exit(0);

}

s=nexttoken(s,temp);

if(\*temp=='=')

{

if(\*(temp+1) ==' '||\*(temp+1)==',')

kpp++;

else

{

dpp++;

}

break; **//handle keyword parameter**

}

pp++;

}

if(\*temp=='=') **//handle keyword parameter**

{

while(\*s)

{

s=nexttoken(s,temp);

if(\*temp!=',' )

strcpy(apt[arglistp-1],temp);

else

strcpy(apt[arglistp-1],"");

s=nexttoken(s,temp);

if(\*temp=='\0')

break;

k = SearchPNT(temp+1);

if(k==-1)

strcpy(arglist[arglistp++],temp+1);

else

{

printf("\nError: Multiple Declaration of Symbol %s in Argument List",temp);

exit(0);

}

}

}

}

void Expand(int n) **//Macro expansion function**

{

int a;

int MEC;

char \*pointer;

MEC = mnt[n].mdtp+1;

while(strcmp(mdt[MEC].opcode,"MEND")!=0)

{

fprintf(fp2,"+%s ",mdt[MEC].opcode);

pointer=mdt[MEC].rest;

pointer=nexttoken(pointer,tok1);

while(tok1[0]!='\0')

{

if(tok1[0]=='#')

{

a=atoi(tok1+1);

fprintf(fp2,"%s",apt[a]);

}

else

fprintf(fp2,"%s",tok1);

pointer=nexttoken(pointer,tok1);

}

fprintf(fp2,"\n");

MEC++;

}

}

int count(char \*pointer)

{

int cnt=0,i,len;

char arr1[10];

strcpy(arr1,pointer);

len=strlen(arr1);

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

{

if(arr1[i]==',')

{

cnt++;

}

}

cnt++;

return cnt;

}

void main()

{

int i=0,j=0,k=0,m,n,cnt=0;

char \*pointer; **//pointer for the array buffer**

printf("\nEnter Source File Name: ");

scanf("%s",FName);

printf("\nEnter Target File Name: ");

scanf("%s",TName);

if((fp1=fopen(FName,"r"))==NULL)

{ printf("\nCannot Open Source File...%s",FName);

exit(0);

}

if((fp2=fopen(TName,"w"))==NULL)

{

printf("\nCannot Create Intermediate File...%s",TName);

exit(0);

}

while(fgets(Buffer,80,fp1))

{

pointer=Buffer;

nexttoken(pointer,tok1);

if(strcmp(tok1,"MACRO")==0)

{

fgets(Buffer,80,fp1);  **//read the parameter line**

pointer=nexttoken(pointer,tok1);

m=SearchMNT(tok1);

if(m>=0)

{

printf("\n duplicate macro definition");

**//test case**

continue;

}

else

{

strcpy(mnt[mntp].mname,tok1);

mnt[mntp].mdtp = mdtp;

strcpy(mdt[mdtp].opcode,tok1);

strcpy(mdt[mdtp].rest,pointer);

mdtp++;

make\_arglist(pointer);

mnt[mntp].pp=pp;

mnt[mntp].kpp=(kpp+dpp);

mntp++;

}

while(fgets(Buffer,80,fp1)) **//store the body of the macro**

{

pointer=Buffer;

pointer=nexttoken(pointer,tok1);

if(strcmp(tok1,"MEND")==0)

{

strcpy(mdt[mdtp].opcode,"MEND");

strcpy(mdt[mdtp++].rest,"");

arglistp=0;;

break;

}

else

{

strcpy(mdt[mdtp].opcode,tok1);

strcpy(mdt[mdtp].rest,"");

pointer=nexttoken(pointer,tok1);

while(tok1[0]!='\0')

{

if(tok1[0]=='&')

{

k = SearchPNT(tok1+1);

if(k==-1)

{

printf("\nError: Parameter %s not found",tok1+1);

exit(0);

}

temp[0]='#';

temp[1]=k+48; **//convert to ASCII**

temp[2]='\0';

strcat(mdt[mdtp].rest,temp);

}

else

strcat(mdt[mdtp].rest,tok1);

pointer=nexttoken(pointer,tok1);

}

}

mdtp++;

}

}

else

{

k = SearchMNT(tok1);

if(k==-1)

fprintf(fp2,"%s",Buffer);

else

{

arglistp=0;

pointer=mdt[mnt[k].mdtp].rest;

make\_arglist(pointer);

pointer=nexttoken(pointer,tok1);

**//Handle positional parameters**

pointer=Buffer;

pointer=nexttoken(pointer,tok1); **//skip macro name**

cnt=count(pointer);

if((cnt)==(pp+kpp) && cnt<=(pp+kpp+dpp))

{

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

{

pointer=nexttoken(pointer,tok1);

strcpy(apt[i],tok1);

pointer=nexttoken(pointer,tok1); **//skip**

}

**//Handle keyword parameters**

pointer=nexttoken(pointer,tok1);

while(tok1[0]!='\0')

{

j=SearchPNT(tok1);

**//get location of the keyword parameter**

**//get the new value of the keyword parameter**

pointer=nexttoken(pointer,tok1);//skip =

pointer=nexttoken(pointer,tok1);

strcpy(apt[j],tok1);

pointer=nexttoken(pointer,tok1); **//skip**

pointer=nexttoken(pointer,tok1);

**//read next parameter**

} }

else

{

printf("\nParameter mismatch\n");

exit(0);

}

Print\_PNT();

Expand(k);

} **//macro expansion**

}

}

Print\_MNT();

Print\_MDT();

printf("\n\n Contents of source file = %s\n\n",FName);

fp1 =fopen(FName,"r");

while(!feof(fp1))

{

char c1 =fgetc(fp1);

printf("%c",c1);

}

fcloseall();

printf("\n\n Contents of target file = %s\n\n",TName);

fp1 =fopen(TName,"r");

while(!feof(fp1))

{

char c1 =fgetc(fp1);

printf("%c",c1);

}

fcloseall();

}

/\*

**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
OUTPUT:  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***  
student@lab7-comp23:~/Desktop$ gcc macro\_err.c  
student@lab7-comp23:~/Desktop$ ./a.out  
  
Enter Source File Name: test1.asm  
  
Enter Target File Name: a.txt  
  
  
------PARAMETER NAME TABLE--------  
# PName  
----------------------------------  
0 X N1  
1 Y N2  
2 REG AREG  
----------------------------------  
  
----------MACRO NAME TABLE---------------------  
# MName #MDTP #pp #kpp  
-----------------------------------------------  
0 INCR 0 2 1  
------------------------------------------------  
  
-----------MACRO DEFINITION TABLE--------------  
Opcode Rest  
-----------------------------------------------  
0 INCR &X, &Y,&REG=AREG  
  
1 MOVER #2,#0  
2 ADD #2,#1  
3 MOVEM #2,#0  
4 MEND   
-----------------------------------------------  
  
 Contents of source file = test1.asm  
  
MACRO  
INCR &X, &Y,&REG=AREG  
MOVER &REG,&X  
ADD &REG,&Y  
MOVEM &REG,&X  
MEND  
  
START 100  
READ N1  
READ N2  
INCR N1,N2  
STOP  
N1 DS 1  
N2 DS 1  
END  
  
  
 Contents of target file = a.txt  
  
  
START 100  
READ N1  
READ N2  
+MOVER AREG,N1  
+ADD AREG,N2  
+MOVEM AREG,N1  
STOP  
N1 DS 1  
N2 DS 1  
END  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
OUTPUT:ERROR HANDLING  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
  
  
student@lab7-comp23:~/Desktop$ gcc macro\_err.c  
student@lab7-comp23:~/Desktop$ ./a.out  
  
Enter Source File Name: test2.asm  
  
Enter Target File Name: a.txt  
 **Parameter mismatch**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
student@lab7-comp23:~/Desktop$ gcc macro\_err.c  
student@lab7-comp23:~/Desktop$ ./a.out  
  
Enter Source File Name: test.asm  
  
Enter Target File Name: a.txt  
  
 **duplicate macro definition**  
  
------PARAMETER NAME TABLE--------  
# PName  
----------------------------------  
0 X N1  
1 Y N2  
2 REG AREG  
----------------------------------  
  
----------MACRO NAME TABLE---------------------  
# MName #MDTP #pp #kpp  
-----------------------------------------------  
0 INCR 0 2 1  
------------------------------------------------  
  
-----------MACRO DEFINITION TABLE--------------  
Opcode Rest  
-----------------------------------------------  
0 INCR &X, &Y,&REG=AREG  
  
1 MOVER #2,#0  
2 ADD #2,#1  
3 MOVEM #2,#0  
4 MEND   
-----------------------------------------------  
  
Contents of source file = test.asm  
  
MACRO  
INCR &X, &Y,&REG=AREG  
MOVER &REG,&X  
ADD &REG,&Y  
MOVEM &REG,&X  
MEND  
  
MACRO  
INCR &A  
MOVER &REG,&A  
MEND  
  
  
START 100  
READ N1  
READ N2  
INCR N1,N2  
STOP  
N1 DS 1  
N2 DS 1  
END  
  
Contents of target file = a.txt  
  
MOVER &REG,&A  
MEND  
  
  
START 100  
READ N1  
READ N2  
+MOVER AREG,N1  
+ADD AREG,N2  
+MOVEM AREG,N1  
STOP  
N1 DS 1  
N2 DS 1  
END  
  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/