**CSCI3334.01 Ana Cecilia Sánchez Ramos**

**PHASE 1**

**phase1.c**

int main(){

char ans[100]; // string that stores the whole line

char comm[10], param1[10], param2[20]; //command + parameters

int ans\_len, n; //string length + number of parameters

int tru = 1;

printf("Welcome!\n");

printf("Here's the available commands:\n\n");

displayCommands();

while(1){

printf("\nPlease enter a command + parameters: ");

fgets(ans, 100, stdin); /\* read in a line \*/

ans\_len = strlen(ans) - 1; /\* get rid of trailing newline character \*/

if (ans[ans\_len] == '\n'){

ans[ans\_len] = '\0';

}

breakLine(ans, comm, param1, param2, &n);

tru = strcmp(comm, "exit");

if(tru == 0 && param1[0] == '\0' && param2[0] == '\0'){

exit(0);

}

else if (tru == 0 && param1[0] != '\0'){

printf("No need for parameters! check \"help\" for more info\n");

}

else{

check(comm,param1,param2);

}

}

return 0;

}

**functions.h**

void displayCommands();

void breakLine(char \*, char \*, char \*, char \*, int \*) ;

void check(char \*comm, char \*param1, char \*param2);

**functions.c**

/\*FUNCTION-> display available commands\*/

void displayCommands(){

printf("load <filename> \nexecute\ndebug\ndump <start> <end>\nhelp\nassemble <filename>\ndirectory\nexit\n");

}

/\*FUNCTION-> break up string input\*/

void breakLine(char \*str, char \*comm, char \*param1, char \*param2, int \*n\_words) {

int str\_ct = 0, comm\_ct = 0, p1\_ct = 0, p2\_ct = 0;

memset(&comm[0],0,sizeof(comm));

memset(&param1[0],0,sizeof(param1));

memset(&param2[0],0,sizeof(param2));

\*n\_words = 0;

while(str[str\_ct] == ' '){ //check for spaces before command

str\_ct++;

}

while(str[str\_ct] != '\0' && str[str\_ct] != ' ') { //find command

comm[comm\_ct] = str[str\_ct];

str\_ct++;

comm\_ct++;

if(str[str\_ct] == '\0' || str[str\_ct] == ' '){ //null-terminate it

comm[comm\_ct] = '\0';

comm\_ct++;

(\*n\_words)++;

}

}

while(str[str\_ct] == ' '){

str\_ct++;

}

/\* find 1st parameter + store it\*/

if(str[str\_ct] != '\0' && str[str\_ct] !=' '){

while(str[str\_ct] != '\0' && str[str\_ct]!=' '){

param1[p1\_ct] = str[str\_ct];

p1\_ct++;

str\_ct++;

if(str[str\_ct] == '\0' || str[str\_ct] == ' '){

param1[p1\_ct] = '\0';

p1\_ct++;

(\*n\_words)++;

}

}

}

else { param1[0]='\0'; }

while(str[str\_ct] == ' '){

str\_ct++;

}

/\*find 2nd parameter + store it\*/

if(str[str\_ct] != '\0' && str[str\_ct] !=' ') {

while(str[str\_ct] != '\0' && str[str\_ct] != ' '){

param2[p2\_ct] = str[str\_ct];

p2\_ct++;

str\_ct++;

if(str[str\_ct] == '\0' || str[str\_ct] == ' '){

param2[p2\_ct] = '\0';

p2\_ct++;

(\*n\_words)++;

}

}

}

else{ param2[0]='\0'; }

}

/\*FUNCTION-> check user input\*/

void check(char \*comm, char \*param1, char \*param2){

char \* commands[8]; //create array to store commands

char\* input;

int cmp;

int index;

/\*fill array with commands\*/

commands[0] = "load";

commands[1] = "execute";

commands[2] = "debug";

commands[3] = "dump";

commands[4] = "help";

commands[5] = "assemble";

commands[6] = "directory";

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

for(int i = 0 ; i < 8; i++){ /\*loop through array to find command\*/

cmp=strcmp(comm,commands[i]);

if(cmp == 0) {

input = comm;

index = i;

break;

}

}

/\*check each case\*/

if(index == 0 || index == 5){ //load or assemble

if(param1[0] != '\0' && param2[0] == '\0'){

printf("command has been reached!\n");

}

else if(param1[0] == '\0'){

printf("You're missing parameters, check \"help\" for more info\n");

}

else{

printf("Too many parameters! check \"help\" for more info\n");

}

}

else if(index == 1 || index == 2){ //execute or debug

if(param1[0] == '\0' && param2[0] == '\0'){

printf("command has been reached!\n");

}

else{

printf("No need for parameters! check \"help\" for more info\n");

}

}

else if(index == 3){ //dump

if(param1[0] != '\0' && param2[0] != '\0'){

printf("command has been reached!\n");

}

else if(param1[0] == '\0' || (param1[0] !='\0' && param2[0] =='\0')){

printf("You're missing parameters, check \"help\" for more info\n");

}

else{

printf("Too many parameters! check \"help\" for more info\n");

}

}

else if(index == 4){ //help

if(param1[0] == '\0' && param2[0] == '\0'){

displayCommands();

}

else{

printf("No need for parameters! check \"help\" for more info\n");

}

}

else if(index == 6) { //directory

if(param1[0]=='\0' && param2[0] == '\0'){

system("ls");

}

else{

printf("No need for parameters! check \"help\" for more info\n");

}

}

else{

printf("Sorry, not a valid command!!! check \"help\" for more info\n");

}

}