

```

1  #include <stdlib.h>
2  #include <stdio.h>
3  #include <string.h>
4  #include<windows.h>
5  #include <time.h>
6
7  //struct definition for scores
8  typedef struct mem{
9      int score;
10     char name[10];
11 }mem;
12
13 //function to clear the concole
14 void clear() {
15     system("clear");
16 }
17
18 //function to wait for an enter to continue, then clears concole
19 void cont() {
20     char wait;
21     printf("Press enter to continue...");
22     do {
23         wait = getchar();
24     } while (wait != '\n');
25     clear(); // Clear the console after waiting for Enter
26 }
27
28 //function to print current scores from a file
29 int currentScores(FILE* file, mem* store){
30     int num = 0;
31     //open file in readmode and check if successful
32     freopen("scores.txt", "r", file);
33     if(file == NULL){
34         printf("Failed to open file.\n");
35         exit(0);
36     }
37     printf("Current Scores:\n");
38     for(int i = 0; i < 10; i++){
39         if(fscanf(file, "%d %s", &store[i].score, store[i].name) >= 2){
40             printf("%2d - %10s: Score of %d\n", i+1, store[i].name, store[i].score);
41             num++;
42         }else{
43             printf("%2d -           : Score of -\n", i+1);
44         }
45     }
46 }
47
48 //function for random symbol printing
49 int symbols(int num){
50     int times = num/3 + 3;
51     char guess[11], actual[11] = "";
52     //limit number of symbols to 10
53     if(times > 10){
54         times = 10;
55     }
56     //time for each symbol slowly goes down per # of symbol
57     int timer = 5000/times;
58     srand(time(NULL));
59     //initail print with a delay
60     printf("\n          w \n\n a          d\n\n          s\n\n");
61     Sleep(500);
62     clear();
63     //loop for random symbols for amount of times
64     for(int i = 0; i < times; i++){
65         int randomNum = rand() % 4;
66         switch(randomNum) {
67             case 0:
68                 printf("          -\n          - w -\n          -\n a          d\n\n          s\n\n %d of\n\n", i+1, times);

```

```

69         strcat(actual, "w");
70         break;
71     case 1:
72         printf("\n      w\n  -\n- a -      d\n  -\n      s\n\n      %d of %d\n", i+1,
73             times);
74         strcat(actual, "a");
75         break;
76     case 2:
77         printf("\n      w\n\n a      d\n      -\n      - s -\n      -\n      %d of
78             %d\n", i+1, times);
79         strcat(actual, "s");
80         break;
81     case 3:
82         printf("\n      w\n      -\n a      - d -\n      -\n      s\n\n
83             %d of %d\n", i+1, times);
84         strcat(actual, "d");
85         break;
86     }
87     Sleep(timer);
88     clear();
89 }
90 //ask for sequence
91 printf("\n      w \n\n a      d\n\n      s\n\n");
92 printf("Enter the sequence: ");
93 scanf(" %s", guess);
94 //if you are right you return 1, or correct. If not, 0 or wrong
95 if(strcmp(guess, actual) == 0){
96     return 1;
97 }
98 return 0;
99 }
100 //game function. Returns a score
101 int playGame(){
102     int play = 1;
103     int score = 0;
104     printf("\n\nMemorize the following sequence: \n");
105     cont();
106     //loop until you get one wrong.
107     while(play){
108         if(score > 0){
109             clear();
110             printf("\nCorrect! Your score is now %d!\nMemorize the following sequence: \n"
111                 , score);
112             getchar();
113             cont();
114         }
115         play = symbols(score);
116         //this only happens if symbols returns 0;
117         if(!play){
118             clear();
119             printf("\nIncorrect. Your final score is %d\n", score);
120             break;
121         }
122         score++;
123     }
124     //returns your final score
125     return score;
126 }
127 //function for bubble sorting the scores and printing them to the file
128 void updateScores(FILE* file, mem* store){
129     // Sort the scores in descending order
130     for(int i = 0; i < 10; i++){
131         for(int j = 0; j < 10 - i; j++){
132             if(store[j].score < store[j + 1].score){
133                 mem temp = store[j];
134                 store[j] = store[j + 1];
135                 store[j + 1] = temp;

```

```

134     }
135 }
136 }
137 //reopen file in write mode, and check if successful
138 freopen("scores.txt", "w", file);
139 if(file == NULL){
140     printf("Failed to open file.\n");
141     exit(0);
142 }
143 // Update the scores in the file
144 for(int i = 0; i <= 10; i++){
145     if(store[i].score > 0 && strcmp(store[i].name, "") != 0){
146         fprintf(file, "%d %s\n", store[i].score, store[i].name);
147     }
148 }
149 }
150
151 int main(void){
152
153     //welcome message
154     clear();
155     printf("\nWelcome to the Memory Game!\nThis is a game to test your memory\n");
156     printf("There will be\n\n      w  \n\n  a      d\n\n      s\n\n\nthat will be
157     circled on your screen\n");
158     printf("Enter the sequence back correctly to earn points!\n");
159     cont();
160
161     //open or create the file and check if successful
162     FILE* highScores;
163     highScores = fopen("scores.txt", "a+");
164     if(highScores == NULL){
165         printf("Failed to open file.\n");
166         exit(0);
167     }
168
169     //struct to store the scores
170     mem scores[11] = {0};
171
172     //print current scores in the file
173     currentScores(highScores, scores);
174     cont();
175
176     //plays the game and your score gets stored at the end of the stuct array
177     scores[10].score = playGame();
178
179     //name gets put into end of stuct array
180     printf("\nEnter your name for the scoreboard: ");
181     scanf(" %10s", &scores[10].name);
182
183     //update the score chart and print new score chart
184     clear();
185     updateScores(highScores, scores);
186     printf("\nUpdated score chart:\n");
187     currentScores(highScores, scores);
188
189     //close the file
190     fclose(highScores);
191     return 0;
192 }

```