```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

// This structure stores one entry in the high score table
struct highscore {
      char name[20];
      int numofwords;
};

// This function reads the word list from the wordlist.txt file
bool readwordlist(char wordlist[][20])
{


      // Task 1. Read the word list from the text file
      //
      // Open the word list file for reading
      //
      FILE *fp = fopen("wordlist.txt", "r");
      if (fp == NULL) {
            printf(" file does not exist.");
                  return false;
      }
      // Read the words using a loop
      //
      for (int i = 0; i < 849; i++) {
            char word[20];
            fscanf(fp, "%s", word);
            strcpy(wordlist[i], word);
      }
      // Close the word list file
      //
      fclose(fp);
      return true;
}

// This function takes the target word and then print the letters in
// a random order.
void printshuffleword(char word[])
{
      char wordcopy[20];
      int letternum;
      int index;

      // Copy the word into another variable
      strcpy(wordcopy, word);

      // Pick and print a letter from the copy randomly
      letternum = 0;
      while (letternum < strlen(word)) {
            // Pick a letter randomly
            index = rand() % strlen(word);

            // Print the letter if it has not been printed
            if (wordcopy[index] != '\0') {
                  printf("%c", wordcopy[index]);
                  wordcopy[index] = '\0';
```

```
                letternum++;
            }
        }
    }

    // This function reads the high score table from a text file into an array.
    void readhighscoretable(highscore table[])
    {


        // Task 2. Read the high score table from the text file
        //
        // Open the high score table file for reading
        //
        FILE *fp = fopen("highscoretable.txt", "r");
        if (fp == NULL) {
            printf(" file does not exist.");
        }
        // Read the high score entries using a while loop
        //
        // Read the high score entries and put them in the table
        int i = 0;
        while (fscanf(fp, "%s %d", table[i].name, &table[i].numofwords) != EOF) {
            i++;
        }
        // Fill the rest of the table array with 0 number of words
        for (i; i < 10; i++) {
            table[i].numofwords = 0;
        }
        // Close the word list file
        //
        fclose(fp);

    }

    // This function inserts a new highscore entry into the table array.
    void inserthighscoretable(highscore table[], int numofwords)
    {
        int i, j;
        char name[20];

        // Ask for the player's name
        printf("Please enter your name: ");
        scanf("%s", name);

        // Find the position of the player in the table
        i = 0;
        while (table[i].numofwords > numofwords && i < 10) {
            i++;
        }

        // Update the table
        if (i < 10) {
            // Push down all the rest of the entries
            for (j = 9; j > i; j--) {
                strcpy(table[j].name, table[j - 1].name);
                table[j].numofwords = table[j - 1].numofwords;
            }
```

```
            // Insert the new entry at the ith position
            strcpy(table[i].name, name);
            table[i].numofwords = numofwords;
      }
}

// This function saves the highscore table to the text file.
void savehighscoretable(highscore table[])
{


      // Task 3. Save the high score table into the text file
      //
      // Open the high score table file for writing
      //
      FILE *fp = fopen("highscoretable.txt", "w");
      if (fp == NULL) {
            printf(" file does not exist.");
      }
      // Write the high score entries using a loop
      //
      for(int i=0;i<10;i++)
            fprintf(fp, "%s %d\n", table[i].name, table[i].numofwords);
      // Close the word list file
      //
      fclose(fp);

}

// This function prints the highscore table.
void printhighscoretable(highscore table[])
{
      int i;

      // Print the title if the highscore table contains at least one entry
      if (table[0].numofwords > 0) {
            printf("\nHighscore Table\n");
            printf("-------------------------\n");
      }

      // Print all valid entries of the highscore table
      i = 0;
      while (table[i].numofwords > 0 && i < 10) {
            printf("%-3d%-20s%3d\n", i + 1, table[i].name, table[i].numofwords);
            i++;
      }
      printf("\n");
}

int main()
{
      // This array stores the 849 words available in the game
      char wordlist[849][20];

      // This is the current number of words the player has guessed correctly
      int numofwords = 0;

      // This stores the target word index
      int targetwordindex;
```

```
	// This stores the guessed word
	char guess[20];

	// This stores the number of guesses the player has used for one word
	int numofguesses;

	// This is the high score table
	highscore table[10];

	// Read the word list file
	if (!readwordlist(wordlist)) {
		return 0;
	}

	// Randomize the random numbers
	srand(time(0));

	printf("Welcome to the word building game!\n");
	printf("---------------------------------\n\n");

	printf("In this game, you have to build a secret word using the letters given
to you!\n\n");
	printf("You have got three chances for each word. If you use up your
chances,\nthe game will be over.\n\n");

	do {
		// Pick a word randomly from the word list
		targetwordindex = rand() % 849;

		printf("Word %d - ", numofwords + 1, wordlist[targetwordindex]);

		// Print the shuffled letters in the target word
		printshuffleword(wordlist[targetwordindex]);
		printf("\n\n");

		// Get the guess from the player
		numofguesses = 0;
		do {
			// Give hints for second and last chance
			if (numofguesses == 1) {
				printf("Hint: the word starts with '%c'.\n",
					wordlist[targetwordindex][0]);
			}
			else if (numofguesses == 2) {
				printf("Hint: the word ends with '%c'.\n",
					wordlist[targetwordindex]
[strlen(wordlist[targetwordindex]) - 1]);
			}

			printf("Please enter your guess: ");
			scanf("%s", guess);

			// Increase the number of guesses
			numofguesses++;

			// The loop stops if either the player guesses correctly or the
			// player guesses three times already
		} while (strcmp(wordlist[targetwordindex], guess) != 0 && numofguesses
```

```
< 3);

            // Increase the number of words if the player guesses correctly
            if (strcmp(wordlist[targetwordindex], guess) == 0) {
                  numofwords++;
            }

            printf("\n");
      } while (strcmp(wordlist[targetwordindex], guess) == 0);

      // Print the game over message
      printf("Game over! You have guessed %d word(s) correctly.\n\n", numofwords);

      // Read the high score table
      readhighscoretable(table);

      // If your score is higher than the minimum score in the table
      if (numofwords > 0 && numofwords >= table[9].numofwords) {
            // Insert your entry in the table
            inserthighscoretable(table, numofwords);

            // Save the high score table
            savehighscoretable(table);
      }

      // Print the high score table
      printhighscoretable(table);
}
```