Wordsearch

computer applications project

Parth Behani

10 ICSE

2017

# Contents

[Certificate 2](#_Toc499895574)

[Certificate 3](#_Toc499895575)

[Introduction 4](#_Toc499895576)

[project details 5](#_Toc499895577)

[Variable description 6](#_Toc499895578)

[Function description 7](#_Toc499895579)

[Screen shot 8](#_Toc499895580)

[System specifications 10](#_Toc499895581)

[Source code 11](#_Toc499895582)

[Improvements 36](#_Toc499895583)

# Certificate

*This is to certify that the project report*

*“Wordsearch”*

*Submitted by*

*Parth Behani*

*Has been submitted in complete fulfillment for the requirement of the Council for the Indian School Certificate Examinations, New Delhi, for the Indian Certificate of Secondary Education class 10 Computer Applications Internal Assessment*

*Rema Nair Parth Behani*

*External Examiner Student*

*Satish Jayarajan*

*Principal*

# Certificate

*This is to certify that the project report*

*“Wordsearch”*

*Submitted by*

*Parth Behani*

*To Mallya Aditi International School, Yelahanka is my original work and that no part of this project has been submitted for the award of any other degree, diploma, or the similar title and that the work has not been previously published in any journal or magazine.*

*Roopa Pulapaka Parth Behani*

*Internal Examiner Student*

*Satish Jayarajan*

*Principal*

# Introduction

Wordsearch is a single player game where the player searches for a set words hidden in a board of random letters. The game is completed when all the words hidden in the board are found.

“Wordsearch” is a user-friendly program with a text based interface. It allows the user to play premade wordsearch games in two levels of difficulty. It also has the capability for the user to enter either 5 three letter words or 5 four letter words, upon which a customized wordsearch board will be generated by the program. The user can then play this custom game.

# project details

**Date of Commencement:** 6th April 2017

**Date of Completion:** 21st May 2017

**Name of Developer:** Parth Behani

**Name of Coordinator:** Roopa Pulapaka

**Approved By:** Roopa Pulapaka,

Rema Nair

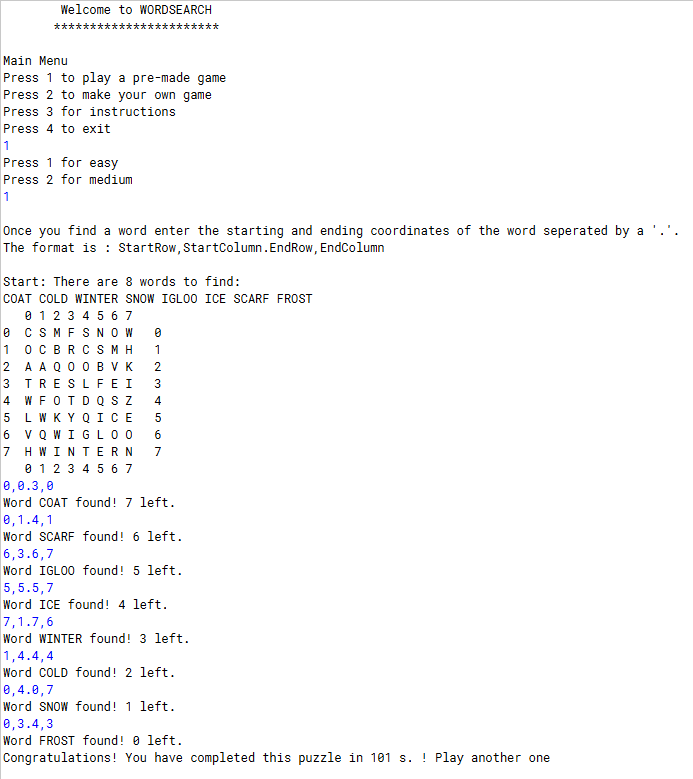
# Variable description

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| sc | static Scanner | Used for all user inputs |
| ws\_easy | static char[ ][ ] | Contains the easy premade wordsearch board in a premade array |
| ws\_medium | static char[ ][ ] | Contains the medium premade wordsearch board in a premade array |
| ws\_3n | static char[ ][ ] | Stores 3 letter user entered custom wordsearch board after generation |
| ws\_4n | static char[ ][ ] | Stores 4 letter user entered custom wordsearch board after generation |
| easy\_words | static String[ ] | Stores solution words to premade easy wordsearch |
| medium\_words | static String[ ] | Stores solution words to premade medium wordsearch |
| words3 | static String[ ] | Stores words to be found in 3 letter user entered custom wordsearch |
| words3\_answers | static String[ ] | Stores co-ordinate answers in 3 letter user entered custom wordsearch |
| words4 | static String[ ] | Stores words to be found in 4 letter user entered custom wordsearch |
| words4\_answers | static String[ ] | Stores co-ordinate answers in 4 letter user entered custom wordsearch |
| alpha | static String | Contains the alphabet |

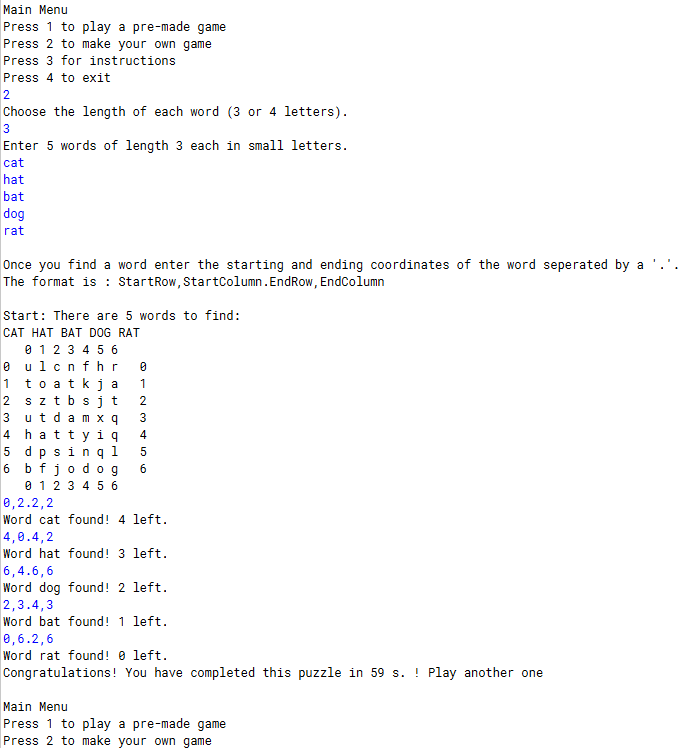
# Function description

|  |  |
| --- | --- |
| **Name** | **Description** |
| Main( ) | Prints opening screen and starts game by calling Menu( ) |
| Menu( ) | Creates game menu |
| Instructions( ) | Prints game instructions |
| Pre\_made( ) | Gives options between easy and medium premade games |
| Easy( ) | Initializes easy premade game |
| Medium( ) | Initializes medium premade game |
| Custom( ) | Gives options between 3 and 4 letter word user entered custom wordsearch game |
| Accept( ) | Accepts words from user for used entered custom wordsearch game |
| RandomL( ) | Generates randomized playing board for user entered custom wordsearch game |
| N3( ) | Places 3 letter user entered words into custom randomized wordsearch board |
| N4( ) | Places 4 letter user entered words into custom randomized wordsearch board |
| Print( ) | Prints wordsearch board |
| Answers( ) | Accepts and checks answers for wordsearch games |

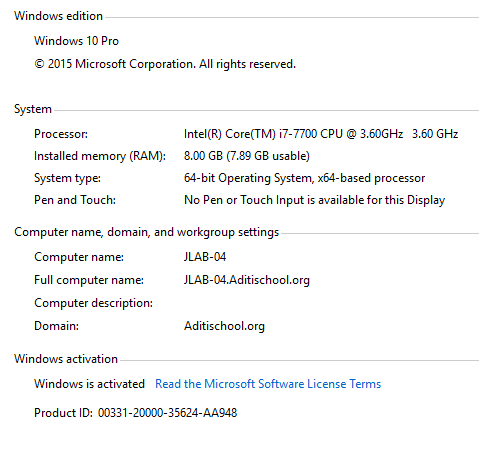
# Screen shot

Premade Game:

Custom Game:



# System specifications



# Source code

import java.util.\*;

class Wordsdearch {

static Scanner sc = new Scanner (System.in);

static char ws\_easy[][]= {{'C','S','M','F','S','N','O','W'},{'O','C','B','R','C','S','M','H'},{'A','A','Q','O','O',' B','V','K'},{'T','R','E' ,'S','L','F','E','I'},{'W','F','O','T','D','Q','S','Z',},{'L','W','K','Y','Q','I','C','E'},{' V','Q','W','I','G','L','O', 'O'},{'H','W','I','N','T','E','R','N'}};

static char ws\_medium[][]= {{'X','D','I','Y','Z','P','A','N','H','S'},{'T','N','I','E','C','O','R','O','A','V',},{'N',' U','Z','L','S','B','C','S',' Y','G'},{'E','O','B','E','K','O','H','M','T','C'},{'D','S','C','L','X','K','A','I','L','A'}, {'N','N','Y','U','M','N ','I','R','E','R'},{'I','U','H','K','Y','U','C','C','V','P'},{'P','M','Z','U','Y','D','G','F',' O','E'},{'C','R','O','S', 'S','B','A','R','N','T'},{'A','W','Y','Y','F','O','N','D','U','E'}};

static char ws\_3n[][]= new char[7][7];

static char ws\_4n[][]= new char[9][9];

static String easy\_words[] = {"COAT", "COLD", "WINTER", "SNOW", "IGLOO", "ICE", "SCARF", "FROST"};

static String medium\_words[] = {"unsound", "ukulele", "carpet", "indent", "crossbar", "dunk", "fondue", "crimson", "novelty", "archaic"};

static String words3[]= new String [5];

static String words3\_answers[]= new String [5];

static String words4[]= new String [5];

static String words4\_answers[]= new String [5];

static String alpha = "abcdefghijklmnopqrstuvwxyz";

public static void main (String [] args){

System.out.println(" Welcome to WORDSEARCH ");

System.out.println(" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* ");

System.out.println();

Menu();

}

public static void Menu (){

int x = 0;

while(true){

Scanner scc = new Scanner (System.in);

try {

System.out.println("Main Menu");

System.out.println("Press 1 to play a pre-made game");

System.out.println("Press 2 to make your own game");

System.out.println("Press 3 for instructions");

System.out.println("Press 4 to exit");

x = scc.nextInt();

break;

}

catch(Exception e){

scc.close();

System.out.println("Enter only integers");

}

}

switch (x){

case 1:

Pre\_made();

break;

case 2:

Custom();

break;

case 3:

Instructions();

Menu();

break;

case 4:

System.exit(0);

break;

default:

System.out.println("Invalid input. Try again");

Menu();

break;

}

}

public static void Instructions(){

System.out.println();

System.out.println("Once you find a word enter the starting and ending coordinates of the word seperated by a '.'.");

System.out.println("The format is : StartRow,StartColumn.EndRow,EndColumn");

System.out.println();

}

public static void Pre\_made (){

int x = 0;

while(true){

Scanner scc = new Scanner (System.in);

try {

System.out.println("Press 1 for easy");

System.out.println("Press 2 for medium");

x = scc.nextInt();

break;

}

catch(Exception e){

scc.close();

System.out.println("Enter only integers");

}

}

switch (x){

case 1:

Easy();

break;

case 2:

Medium();

break;

default:

System.out.println("Invalid input. Try again");

Pre\_made();

break;

}

}

public static void Easy (){

Print(8,8,8,1);

Answers(8,1);

}

public static void Medium (){

Print(10,10,10,2);

Answers(10,2);

}

public static void Custom (){

int x = 0;

while(true){

Scanner scc = new Scanner (System.in);

try {

System.out.println("Choose the length of each word (3 or 4 letters).");

x = scc.nextInt();

break;

}

catch(Exception e){

scc.close();

System.out.println("Enter only integers");

}

}

switch (x){

case 3:

Accept(3,11);

RandomL(7,7,11);

N3();

Print(7,7,5,11);

Answers(5,11);

break;

case 4:

Accept(4,12);

RandomL(9,9,12);

N4();

Print(9,9,5,12);

Answers(5,12);

break;

default:

System.out.println("Invalid input. Try again");

Custom();

break;

}

}

public static void Accept(int length, int type){

System.out.println("Enter 5 words of length " + length + " each in small letters.");

for (int i=0; i<5; i++){

switch(type){

case 11:

words3[i]=sc.next();

if(words3[i].length()!=length){

System.out.println("Wrong length. Try again.");

i--;

continue;

}

for(int j=0; j< length; j++){

if(words3[i].charAt(j)>=97 && words3[i].charAt(j)<=122){

continue;

}

System.out.println("Invalid character present. Try again.");

i--;

break;

}

break;

case 12:

words4[i]=sc.next();

if(words4[i].length()!=length){

System.out.println("Wrong length. Try again.");

i--;

continue;

}

for(int j=0; j< length; j++){

if(words4[i].charAt(j)>=97 && words4[i].charAt(j)<=122){

continue;

}

System.out.println("Invalid character present. Try again.");

i--;

break;

}

break;

}

}

}

public static void RandomL (int x, int y, int type){

Random ran = new Random();

for (int i=0; i<x; i++){

for (int j=0; j<y; j++){

switch (type){

case 11:

ws\_3n[i][j]=alpha.charAt(ran.nextInt(26));

break;

case 12:

ws\_4n[i][j]=alpha.charAt(ran.nextInt(26));

break;

}

}

}

}

public static void N3 (){

Random ran = new Random();

int r1 = ran.nextInt(3);

int r2 = ran.nextInt(3);

int r3 = ran.nextInt(2);

int r4 = ran.nextInt(2);

int r5 = ran.nextInt(2);

if (r2==0){

r2=3;

}

else if (r2==1){

r2=4;

}

else {

r2=5;

}

if (r3==0){

r3=3;

}

else {

r3=4;

}

if (r4==0){

r4=5;

}

else {

r4=6;

}

if (r5==0){

r5=5;

}

else {

r5=6;

}

words3\_answers[0]="0,"+r1+".2,"+r1;

words3\_answers[1]=r2+",0."+r2+",2";

words3\_answers[2]="2,"+r3+".4,"+r3;

words3\_answers[3]=r4+",4."+r4+",6";

words3\_answers[4]="0,"+r5+".2,"+r5;

for (int i=0; i<7; i++){

for (int j=0; j<7; j++){

if(j==r1){

if (i==0||i==1||i==2){

ws\_3n[i][j]=words3[0].charAt(i);

}

}

if (i==r2){

if (j==0||j==1||j==2){

ws\_3n[i][j]=words3[1].charAt(j);

}

}

if (j==r3){

if (i==2||i==3||i==4){

ws\_3n[i][j]=words3[2].charAt(i-2);

}

}

if (i==r4){

if (j==4||j==5||j==6){

ws\_3n[i][j]=words3[3].charAt(j-4);

}

}

if (j==r5){

if (i==0||i==1||i==2){

ws\_3n[i][j]=words3[4].charAt(i);

}

}

}

}

}

public static void N4(){

Random ran = new Random();

int r1 = ran.nextInt(4);

int r2 = ran.nextInt(3);

int r3 = ran.nextInt(5);

int r4 = ran.nextInt(2);

int r5 = ran.nextInt(5);

if (r3==0){

r3=4;

}

else if (r3==1){

r3=5;

}

else if (r3==2){

r3=6;

}

else if (r3==3){

r3=7;

}

else {

r3=8;

}

if (r4==0){

r4=7;

}

else {

r4=8;

}

if (r5==0){

r5=4;

}

else if (r5==1){

r5=5;

}

else if (r5==2){

r5=6;

}

else if (r5==3){

r5=7;

}

else {

r5=8;

}

words4\_answers[0]=r1+",0."+r1+",3";

words4\_answers[1]="4,"+r2+".7,"+r2;

words4\_answers[2]=r3+",3."+r3+",6";

words4\_answers[3]="5,"+r4+".8,"+r4;

words4\_answers[4]="0,"+r5+".3,"+r5;

//0 for forwards and 1 for backwards

int rd1 = ran.nextInt(2);

int rd2 = ran.nextInt(2);

int rd3 = ran.nextInt(2);

int rd4 = ran.nextInt(2);

int rd5 = ran.nextInt(2);

for (int i=0; i<9; i++){

for (int j=0; j<9; j++){

if(i==r1){

if (j==0||j==1||j==2||j==3){

if (rd1 == 0){

ws\_4n[i][j]=words4[0].charAt(j);

}

else {

words4\_answers[0]=r1+",3."+r1+",0";

ws\_4n[i][j]=words4[0].charAt(3-j);

}

}

}

if (j==r2){

if (i==4||i==5||i==6||i==7){

if (rd2==0){

ws\_4n[i][j]=words4[1].charAt(i-4);

}

else {

words4\_answers[1]="7,"+r2+".4,"+r2;

ws\_4n[i][j]=words4[1].charAt(7-i);

}

}

}

if (i==r3){

if (j==3||j==4||j==5||j==6){

if (rd3==0){

ws\_4n[i][j]=words4[2].charAt(j-3);

}

else {

words4\_answers[2]=r3+",6."+r3+",3";

ws\_4n[i][j]=words4[2].charAt(6-j);

}

}

}

if (j==r4){

if (i==5||i==6||i==7||i==8){

if (rd4==0){

ws\_4n[i][j]=words4[3].charAt(i-5);

}

else {

words4\_answers[3]="8,"+r4+".5,"+r4;

ws\_4n[i][j]=words4[3].charAt(8-i);

}

}

}

if (j==r5){

if (i==0||i==1||i==2||i==3){

if (rd5==0){

ws\_4n[i][j]=words4[4].charAt(i);

}

else{

words4\_answers[4]="3,"+r5+".0,"+r5;

ws\_4n[i][j]=words4[4].charAt(3-i);

}

}

}

}

}

}

public static void Print (int x, int y, int words, int mode){

System.out.println();

System.out.println("Once you find a word enter the starting and ending coordinates of the word seperated by a '.'.");

System.out.println("The format is : StartRow,StartColumn.EndRow,EndColumn");

System.out.println();

System.out.println("Start: There are "+words+" words to find: ");

for (int i=0; i<words; i++){

switch (mode){

case 1:

System.out.print(easy\_words[i].toUpperCase()+" ");

break;

case 2:

System.out.print(medium\_words[i].toUpperCase()+" ");

break;

case 11:

System.out.print(words3[i].toUpperCase()+" ");

break;

case 12:

System.out.print(words4[i].toUpperCase()+" ");

break;

}

}

System.out.println();

System.out.print(" ");

for (int i=0; i<y; i++){

System.out.print(i+" ");

}

for (int i=0; i<x; i++){

System.out.println();

if (i<10) {

System.out.print(i+" ");

}

else {

System.out.print(i+" ");

}

for (int j=0; j<y; j++){

switch (mode){

case 1:

System.out.print(ws\_easy[i][j]+ " ");

break;

case 2:

System.out.print(ws\_medium[i][j]+ " ");

break;

case 11:

System.out.print(ws\_3n[i][j]+ " ");

break;

case 12:

System.out.print(ws\_4n[i][j]+ " ");

break;

}

}

if (i<10){

System.out.print(" "+i);

}

else {

System.out.print(" "+i);

}

}

System.out.println();

System.out.print(" ");

for (int i=0; i<y; i++){

System.out.print(i+" ");

}

}

public static void Answers (int num, int mode){

long startTime = System.currentTimeMillis();

System.out.println();

String a[] = {"0,0.3,0", "1,4.4,4", "7,1.7,6", "0,4.0,7", "6,3.6,7", "5,5.5,7", "0,1.4,1", "0,3.4,3"};

String b[] = {"6,1.0,1", "7,3.1,3", "3,9.8,9", "6,0.1,0","8,0.8,7", "7,5.4,5", "9,4.9,9", "6,7.0,7", "8,8.2,8", "0,6.6,6"} ;

boolean correct = false;

boolean quitter = false;

int left = num;

while (left>0){

String x = sc.next();

if(x.equalsIgnoreCase("e")|| x.equalsIgnoreCase("exit")){

System.out.println("Do you really want to exit? [y/n]");

boolean valid = false;

while (valid == false){

String e = sc.next();

if (e.equalsIgnoreCase("y")||e.equalsIgnoreCase("yes")){

System.out.println("Exiting...");

quitter = true;

left = -1;

valid = true;

}

else if (e.equalsIgnoreCase("n")||e.equalsIgnoreCase("no")){

System.out.println("Continuing...");

valid = true;

}

else {

System.out.println("Invalid input. Try again");

}

}

correct = true;

}

for (int i=0; i<num; i++){

switch (mode){

case 1:

if (a[i].equals(x)){

System.out.println("Word "+easy\_words[i]+" found! "+(--left)+ " left.");

a[i]="";

correct = true;

}

break;

case 2:

if (b[i].equals(x)){

System.out.println("Word "+medium\_words[i]+" found! "+(--left)+ " left.");

b[i]="";

correct = true;

}

break;

case 11:

if (words3\_answers[i].equals(x)){

System.out.println("Word "+words3[i]+" found! "+(--left)+ " left.");

words3\_answers[i]="";

correct = true;

}

break;

case 12:

if (words4\_answers[i].equals(x)){

System.out.println("Word "+words4[i]+" found! "+(--left)+ " left.");

words4\_answers[i]="";

correct = true;

}

break;

}

}

if (correct==false){

System.out.println("Wrong entry, try again.");

}

else {

correct = false;

}

}

long Time = (System.currentTimeMillis() - startTime)/1000;

if (quitter==false){

System.out.println("Congratulations! You have completed this puzzle in " +Time + " s. ! Play another one");

}

System.out.println();

Menu();

}

}

# Improvements

* The program could have a graphical interface.
* There could be more pre-made wordsearch puzzles.
* The custom generated wordsearch puzzles mechanism could be improved to accept longer words and words of different lengths in the same puzzle.
* The custom generated wordsearch puzzles could have capabilities to let certain words overlap at common letter.