/\*

Name : Muhammad Nouman Zafar

Description : This is a cards game called WAR. where number of player choosen by the user and there are 13 rounds to complete the

game where each player select there cards and the Higest Unique card wins the game and the player gets all the points.

\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <time.h>

//constant strings with the name of the cards and teh name of the suits

const char suit[4][20] = { "Spades","Dimonds","Hearts","Clubs" };

const char cardFace[13][15] = { "Two","Three","Four","Five","Six","Seven","Eight","Nine","Ten","Jack","Queen","King","Ace" };

//defining the total cards and the number of decks

#define TOTAL\_CARD\_DECK 52

#define TOTAL\_DECK 3

typedef int Deck[TOTAL\_DECK][TOTAL\_CARD\_DECK];

//declaring the global variables so

int faceValue = 0;

int card;

char option;

char yOrn;

char save;

char startGame;

int fileNumber;

int l = 0;

int y = 0;

int specialRoundScore = 0;

int totalPlayers;

int pFaceValue;

int player;

//array for saving the dealt cards for each player

int playerOne[13];

int playerTwo[13];

int playerThree[13];

int playerFour[13];

int playerFive[13];

int playerSix[13];

int playerSeven[13];

int playerEight[13];

int playerNine[13];

int playerTen[13];

//arrays for scoring part

int perRoundSelectedCards[10];

int frequency[10];

int finalScores[10];

int selectedCard;

//for saving the selected cards by each player in each round

int p1SelectedCards[13];

int p2SelectedCards[13];

int p3SelectedCards[13];

int p4SelectedCards[13];

int p5SelectedCards[13];

int p6SelectedCards[13];

int p7SelectedCards[13];

int p8SelectedCards[13];

int p9SelectedCards[13];

int p10SelectedCards[13];

int p1FaceValue = 0;

int p2FaceValue = 0;

int p3FaceValue = 0;

int p4FaceValue = 0;

int p5FaceValue = 0;

int p6FaceValue = 0;

int p7FaceValue = 0;

int p8FaceValue = 0;

int p9FaceValue = 0;

int p10FaceValue = 0;

//scores of each player

int p1Scores = 0;

int p2Scores = 0;

int p3Scores = 0;

int p4Scores = 0;

int p5Scores = 0;

int p6Scores = 0;

int p7Scores = 0;

int p8Scores = 0;

int p9Scores = 0;

int p10Scores = 0;

int i, j, count, largestNumber, b, winnerScore, fileNumber1, ca = 0;

//function prototypes

void setUpDeck(Deck);

void cards(int card);

void printDeck(Deck);

void shuffleCards(Deck deck);

void dealDeck(Deck deck);

void playGame();

int playerMethod(int playerSelectedCards[13], int playerCards[13]);

void newGame();

//file pointer in order to access the files

//for rading and writing

FILE\* filePointer;

int main()

{

for (int opt = 0; opt < 1; opt++) {

//prompting the user

//for starting a new game or previously saved game

printf("Enter (N) to Open a New Game\nEnter (P) to previously Saved Game\nEnter (C) to Close\n");

scanf(" %c", &option);

//switch statement

//in case of n new game will be started

//p will start the previously started game

switch (option)

{

case 'n':

case 'N':

newGame();

break;

case 'p':

case 'P':

//reading from the file called war.txt

filePointer = fopen("war.txt", "r");

if (filePointer == NULL) {

printf("File could not be opend\n");

}

else {

//reading the fill /until its not the end of file

while (!feof(filePointer)) {

//reading the cards each and every single player has

//and the all teh cards selected by each player

//so player wont have to choose the same card again

for (int g = 0; g < 13; g++) {

fileNumber = fscanf(filePointer, "%d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d\n", &playerOne[g], &playerTwo[g], &playerThree[g],

&playerFour[g], &playerFive[g], &playerSix[g], &playerSeven[g], &playerEight[g], &playerNine[g]

, &playerTen[g], &p1SelectedCards[g], &p2SelectedCards[g], &p3SelectedCards[g], &p4SelectedCards[g], &p5SelectedCards[g]

, &p6SelectedCards[g], &p7SelectedCards[g], &p8SelectedCards[g], &p9SelectedCards[g], &p10SelectedCards[g]);

}

//reading the scores and the total number of player

fileNumber1 = fscanf(filePointer, "\n%d %d %d %d %d %d %d %d %d %d %d %d %d", &totalPlayers, &card, &specialRoundScore, &p1Scores, &p2Scores, &p3Scores, &p4Scores

, &p5Scores, &p6Scores, &p7Scores, &p8Scores, &p9Scores, &p10Scores);

//displaying the round number and the start the previos saved game

if (fileNumber > 0 && fileNumber1 > 0) {

for (int i = 0; i < 1; i++) {

printf("\n\nNumber of Players : %d", totalPlayers);

printf("\nNumber of rounds completed : %d", card);

printf("\n\n\tWant to complete the next Round ?");

printf("\n\t (Y)\tOR\t(N)\n");

scanf(" %c", &yOrn);

switch (yOrn)

{

case 'y':

case 'Y':

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

printf("\t\tROUND NUMBER %d", card + 1);

ca = card;

playGame();

break;

case 'n':

case 'N':

exit(0);

break;

default:

i--;

break;

}

}

}

}

//closing the file

fclose(filePointer);

}

break;

//if c is selected game would be closed

case 'c':

case 'C':

exit(0);

break;

default:

//if input is not valid ask the user again

//unlit its not the right one

opt--;

break;

}

}

}

//function for starting the new game

void newGame() {

Deck deck;

//asking the user how many players in total

printf("\nPlease Enter number of Players : ");

scanf("%d", &totalPlayers);

setUpDeck(deck);

shuffleCards(deck);

playGame();

}

//setup the Deck

void setUpDeck(Deck deck) {

//set up deck

//3 in total

//and each deck has 52 cards

for (int j = 0; j < TOTAL\_DECK; j++) {

for (int i = 0; i < TOTAL\_CARD\_DECK; i++) {

deck[j][i] = i;

}

}

}

//creating the cards

void cards(int card) {

//giving each card a value

if (strcmp(cardFace[card % 13], "Two") == 0) {

faceValue = 2;

}

else if (strcmp(cardFace[card % 13], "Three") == 0) {

faceValue = 3;

}

else if (strcmp(cardFace[card % 13], "Four") == 0) {

faceValue = 4;

}

else if (strcmp(cardFace[card % 13], "Five") == 0) {

faceValue = 5;

}

else if (strcmp(cardFace[card % 13], "Six") == 0) {

faceValue = 6;

}

else if (strcmp(cardFace[card % 13], "Seven") == 0) {

faceValue = 7;

}

else if (strcmp(cardFace[card % 13], "Eight") == 0) {

faceValue = 8;

}

else if (strcmp(cardFace[card % 13], "Nine") == 0) {

faceValue = 9;

}

else if (strcmp(cardFace[card % 13], "Ten") == 0) {

faceValue = 10;

}

else if (strcmp(cardFace[card % 13], "Jack") == 0) {

faceValue = 11;

}

else if (strcmp(cardFace[card % 13], "Queen") == 0) {

faceValue = 12;

}

else if (strcmp(cardFace[card % 13], "King") == 0) {

faceValue = 13;

}

else if (strcmp(cardFace[card % 13], "Ace") == 0) {

faceValue = 14;

}

else {

faceValue = 0;

}

printf(" %d\t |\t %d\t |\t%s of %s\t |\n", faceValue, card + 1, cardFace[card % 13], suit[card / 13]);

}

//printing the cards

//all the cards will be printed

void printDeck(Deck deck) {

for (int j = 0; j < TOTAL\_DECK; j++) {

for (int i = 0; i < TOTAL\_CARD\_DECK; i++) {

cards(deck[j][i]);

}

}

printf("\n");

}

void shuffleCards(Deck deck) {

//shuffling the deck /in order to give each player random cards

//and in every game cards will be different

time\_t t;

srand((unsigned)time(&t));

for (int j = 0; j < TOTAL\_DECK; j++) {

for (int i = 0; i < TOTAL\_CARD\_DECK; i++) {

int card = rand() % 52;

//temporarily hold the value of the card

int temp = deck[j][card];

deck[j][card] = deck[j][i];

deck[j][i] = temp;

}

}

//printDeck(deck);

dealDeck(deck);

}

void dealDeck(Deck deck) {

int i;

int j;

int p;

//dealing the cards to each played

//depends on how many players are playing the game at a time

if (totalPlayers > 2 && totalPlayers <= 10) {

//minimum number of player is 3

//and max is 10

//1 deck is only for 4 players and if number of player exceeds

//number of decks exceeds as well

for (p = 0; p < totalPlayers; p++) {

if (p == 0) {

for (i = 0; i < 13; i++)

{

playerOne[i] = deck[0][i];

}

}

else if (p == 1) {

for (i = 13, j = 0; i < 26, j < 13; i++, j++) {

playerTwo[j] = deck[0][i];

}

}

else if (p == 2) {

for (i = 26, j = 0; i < 39, j < 13; i++, j++) {

playerThree[j] = deck[0][i];

}

}

else if (p == 3) {

for (i = 39, j = 0; i < 52, j < 13; i++, j++) {

playerFour[j] = deck[0][i];

}

}

else if (p == 4) {

for (i = 0; i < 13; i++)

{

playerFive[i] = deck[1][i];

}

}

else if (p == 5) {

for (i = 13, j = 0; i < 26, j < 13; i++, j++) {

playerSix[j] = deck[1][i];

}

}

else if (p == 6) {

for (i = 26, j = 0; i < 39, j < 13; i++, j++) {

playerSeven[j] = deck[1][i];

}

}

else if (p == 7) {

for (i = 39, j = 0; i < 52, j < 13; i++, j++) {

playerEight[j] = deck[1][i];

}

}

else if (p == 8) {

for (i = 0; i < 13; i++)

{

playerNine[i] = deck[2][i];

}

}

else if (p == 9) {

for (i = 13, j = 0; i < 26, j < 13; i++, j++) {

playerTen[j] = deck[2][i];

}

}

}

}

else {

printf("\nNumber of players has to be in between 2 and 10\n");

newGame();

}

}

void playGame() {

//function for playing the game

//calling the function playerMethod

//which has two perameters

//one is for the cards selected by each player in each round

//and the other is for the the actual cards of each player

//adding the value of selected card

//and represent

//to see who is the winner of the round

if (totalPlayers > 2 && totalPlayers <= 10) {

for (card = ca; card < 13; card++) {

for (player = 0; player < totalPlayers; player++) {

if (player == 0) {

playerMethod(p1SelectedCards, playerOne);

p1FaceValue = pFaceValue;

perRoundSelectedCards[0] = p1FaceValue;

}

else if (player == 1) {

playerMethod(p2SelectedCards, playerTwo);

p2FaceValue = pFaceValue;

perRoundSelectedCards[1] = p2FaceValue;

}

else if (player == 2) {

playerMethod(p3SelectedCards, playerThree);

p3FaceValue = pFaceValue;

perRoundSelectedCards[2] = p3FaceValue;

}

else if (player == 3) {

playerMethod(p4SelectedCards, playerFour);

p4FaceValue = pFaceValue;

perRoundSelectedCards[3] = p4FaceValue;

}

else if (player == 4) {

playerMethod(p5SelectedCards, playerFive);

p5FaceValue = pFaceValue;

perRoundSelectedCards[4] = p5FaceValue;

}

else if (player == 5) {

playerMethod(p6SelectedCards, playerSix);

p6FaceValue = pFaceValue;

perRoundSelectedCards[5] = p6FaceValue;

}

else if (player == 6) {

playerMethod(p7SelectedCards, playerSeven);

p7FaceValue = pFaceValue;

perRoundSelectedCards[6] = p7FaceValue;

}

else if (player == 7) {

playerMethod(p8SelectedCards, playerEight);

p8FaceValue = pFaceValue;

perRoundSelectedCards[7] = p8FaceValue;

}

else if (player == 8) {

playerMethod(p9SelectedCards, playerNine);

p9FaceValue = pFaceValue;

perRoundSelectedCards[8] = p9FaceValue;

}

else if (player == 9) {

playerMethod(p10SelectedCards, playerTen);

p10FaceValue = pFaceValue;

perRoundSelectedCards[9] = p10FaceValue;

}

}

//displaying the results of the round

printf("\n\n\n\t\tRound %d Results\n", card + 1);

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

printf(" Face Values of cards selected by the Players |\n");

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

printf("\tPlayer Number\t\tCard Value\t |\n");

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

//value of the cards selected by each player

//

for (int h = 0; h < 10; h++) {

if (perRoundSelectedCards[h] > 0) {

printf("\tPlayer %d\t|\t%d\t\t |\n", h + 1, perRoundSelectedCards[h]);

}

}

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

//setting up the frequency array to -1

for (int v = 0; v < 10; v++) {

frequency[v] = -1;

}

//checking the higest unque card

for (i = 0; i < 10; i++)

{

count = 1;

for (j = i + 1; j < 10; j++)

{

//if cards is repeating it self then count increases by one

//and frequency set to 0

if (perRoundSelectedCards[i] == perRoundSelectedCards[j])

{

count++;

frequency[j] = 0;

}

}

//if frequency is not equal to 0

//then set the it to the count

if (frequency[i] != 0)

{

frequency[i] = count;

}

}

largestNumber = 0;

for (int e = 0; e < 10; e++)

{

if (frequency[e] == 1)

{

//getting the heigest unique number from the array

if (perRoundSelectedCards[e] > largestNumber) {

largestNumber = perRoundSelectedCards[e];

}

}

}

printf("\t\t\t\t\t\t |\n");

printf("\tLargest Unique Number is : %d\t\t |\n", largestNumber);

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

for (int f = 0; f < 10; f++) {

if (largestNumber == perRoundSelectedCards[f] && largestNumber != 0) {

printf("\tPlayer %d has won the round\t\t |\n", f + 1);

//adding the scores to the player

//depends upon who won the round

//whoever had the heigest uniwue card will get all the points

if (f == 0) {

for (b = 0; b < 10; b++) {

p1Scores += perRoundSelectedCards[b];

}

p1Scores += specialRoundScore;

}

else if (f == 1) {

for (b = 0; b < 10; b++) {

p2Scores += perRoundSelectedCards[b];

}

p2Scores += specialRoundScore;

}

else if (f == 2) {

for (b = 0; b < 10; b++) {

p3Scores += perRoundSelectedCards[b];

}

p3Scores += specialRoundScore;

}

else if (f == 3) {

for (b = 0; b < 10; b++) {

p4Scores += perRoundSelectedCards[b];

}

p4Scores += specialRoundScore;

}

else if (f == 4) {

for (b = 0; b < 10; b++) {

p5Scores += perRoundSelectedCards[b];

}

p5Scores += specialRoundScore;

}

else if (f == 5) {

for (b = 0; b < 10; b++) {

p6Scores += perRoundSelectedCards[b];

}

p6Scores += specialRoundScore;

}

else if (f == 6) {

for (b = 0; b < 10; b++) {

p7Scores += perRoundSelectedCards[b];

}

p7Scores += specialRoundScore;

}

else if (f == 7) {

for (b = 0; b < 10; b++) {

p8Scores += perRoundSelectedCards[b];

}

p8Scores += specialRoundScore;

}

else if (f == 8) {

for (b = 0; b < 10; b++) {

p9Scores += perRoundSelectedCards[b];

}

p9Scores += specialRoundScore;

}

else if (f == 9) {

for (b = 0; b < 10; b++) {

p10Scores += perRoundSelectedCards[b];

}

p10Scores += specialRoundScore;

}

specialRoundScore = 0;

break;

}

//if there is no heigest unique card in the round

//then the points are saved as specialRoundScores

//and applid to the player

//who will will the next round

else if (largestNumber == 0) {

for (int r = 0; r < 10; r++) {

specialRoundScore += perRoundSelectedCards[r];

}

printf("\t\t\t\t\t\t |\n");

printf("\t\t TIE ROUND\t\t\t |\n");

printf("\t\t\t\t\t\t |\n");

printf("\tThis round scores will be added\t\t |\n\t To next round winer's scores\t\t |\n");

printf("\t\t\t\t\t\t |\n");

printf("\t\t\t\t\t\t |\n");

break;

}

}

//displaying the results and

//scores of all the players

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

printf("\t\t\t\t\t\t |\n");

printf("\t\t\t\t\t\t |\n");

printf("\t\t Scores \t\t\t |\n");

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

printf("\tPlayer Number\t\tTotal Score\t |\n");

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

printf("\tPlayer 1\t|\t%d\t\t |\n", p1Scores);

printf("\tPlayer 2\t|\t%d\t\t |\n", p2Scores);

printf("\tPlayer 3\t|\t%d\t\t |\n", p3Scores);

printf("\tPlayer 4\t|\t%d\t\t |\n", p4Scores);

printf("\tPlayer 5\t|\t%d\t\t |\n", p5Scores);

printf("\tPlayer 6\t|\t%d\t\t |\n", p6Scores);

printf("\tPlayer 7\t|\t%d\t\t |\n", p7Scores);

printf("\tPlayer 8\t|\t%d\t\t |\n", p8Scores);

printf("\tPlayer 9\t|\t%d\t\t |\n", p9Scores);

printf("\tPlayer 10\t|\t%d\t\t |\n", p10Scores);

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

printf("\n\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*Round %d Completed\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n", card + 1);

//prompting the user

//whether they want to save the game or continue thw game

for (int xx = 0; xx < 1; xx++) {

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

printf("\t Enter (S) to save the Game\t\t |\n\t\t\t\t\t\t |\n\t\t\tOR\t\t\t |\n\t\t\t\t\t\t |\n\t Enter (C) to continue\t\t |\n");

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

scanf(" %c", &save);

switch (save)

{

case 'c':

case 'C':

break;

case 's':

case 'S':

//if s is enterd

//it will save all the cards

//and the the cards selected in each round

//will be saved to the file called war.txt

filePointer = fopen("war.txt", "w");

if (filePointer == NULL) {

printf("File could not be opend\n");

}

else {

for (int g = 0; g < 13; g++) {

fprintf(filePointer, "%d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d\n", playerOne[g], playerTwo[g], playerThree[g],

playerFour[g], playerFive[g], playerSix[g], playerSeven[g], playerEight[g], playerNine[g]

, playerTen[g], p1SelectedCards[g], p2SelectedCards[g], p3SelectedCards[g], p4SelectedCards[g], p5SelectedCards[g]

, p6SelectedCards[g], p7SelectedCards[g], p8SelectedCards[g], p9SelectedCards[g], p10SelectedCards[g]);

}

fprintf(filePointer, "\n%d %d %d %d %d %d %d %d %d %d %d %d %d", totalPlayers, card + 1, specialRoundScore, p1Scores, p2Scores, p3Scores, p4Scores

, p5Scores, p6Scores, p7Scores, p8Scores, p9Scores, p10Scores);

}

printf("\n\n\t Game has been Saved\n\n\n");

exit(0);

break;

default:

xx--;

break;

}

}

}

//checking who is the overall winner

//whoever has the heigest score in all the 13 rounds

//going to winn the game

for (int t = 0; t < 10; t++) {

if (t == 0) {

finalScores[0] = p1Scores;

}

else if (t == 1) {

finalScores[1] = p2Scores;

}

else if (t == 2) {

finalScores[2] = p3Scores;

}

else if (t == 3) {

finalScores[3] = p4Scores;

}

else if (t == 4) {

finalScores[4] = p5Scores;

}

else if (t == 5) {

finalScores[5] = p6Scores;

}

else if (t == 6) {

finalScores[6] = p7Scores;

}

else if (t == 7) {

finalScores[7] = p8Scores;

}

else if (t == 8) {

finalScores[8] = p9Scores;

}

else if (t == 9) {

finalScores[9] = p10Scores;

}

}

winnerScore = 0;

for (int s = 0; s < 10; s++) {

if (finalScores[s] > winnerScore) {

winnerScore = finalScores[s];

}

}

printf("\n\n\n\n\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ALL ROUNDS COMPLETED\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n\n\n\n");

//displaying the winner of teh game

for (int w = 0; w < 10; w++) {

if (winnerScore == finalScores[w]) {

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

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

printf("\t\t\t\t\t\t |\n");

printf("\t\t\t\t\t\t |\n");

printf("\t PLAYER %d IS THE WINNER\t\t |\n", w + 1);

printf("\t\t\t\t\t\t |\n");

printf("\t\t\t\t\t\t |\n");

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

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

}

}

//after finnish up the new game

//prompt to user

//if they want to start a new game again

for (int v = 0; v < 1; v++) {

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

printf("\t Enter (N) to Start New Game\t\t |\n\t\t\t\t\t\t |\n\t\t\tOR\t\t\t |\n\t\t\t\t\t\t |\n\t Enter (C) to Close\t\t |\n");

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

scanf(" %c", &startGame);

switch (startGame)

{

case 'n':

case 'N':

system("@cls||clear");

newGame();

break;

case 'c':

case 'C':

exit(0);

break;

default:

v--;

break;

}

}

}

else {

printf("\nNumber of players has to be in between 2 and 10\n");

newGame();

}

}

//method for playing the game

//displaying the card of each player

//and give the user option

//to select the card

int playerMethod(int playerSelectedCards[13], int playerCards[13]) {

int selectedCard;

if (l < 13) {

printf("\n\n");

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

printf("\t\tPlayer %d Cards\t\t\t |\n", player + 1);

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

printf("Card Value\tCard Number\t\tCard\t |\n");

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

for (y = 0; y < 13; y++) {

if (playerCards[y] + 1 != playerSelectedCards[y]) {

cards(playerCards[y]);

}

}

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

}

printf("\nPlayer %d Please enter the card number to select the card : ", player + 1);

scanf("%d", &selectedCard);

printf("\n");

for (l = 0; l < 13; l++) {

if ((selectedCard == playerCards[l] + 1) && (selectedCard != playerSelectedCards[l])) {

printf("\n\nPlayer %d Selected cards is : \n", player + 1);

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

cards(selectedCard - 1);

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

printf("\n");

//system("@cls||clear");

playerSelectedCards[l] = playerCards[l] + 1;

pFaceValue = faceValue;

break;

}

else {

if (l == 12) {

player--;

}

}

}

return pFaceValue;

}