#include <stdio.h>

#include <time.h>

#include <stdlib.h>

#define MAX\_PLAYERS 4

#define SUIT 4

#define DECK 13

typedef struct {

int id;

}Player;

typedef struct {

char name[4];

int value;

}Card;

//declare functions

void setPlayerInfo(int numPlayer, Player playerList[MAX\_PLAYERS]);

void deal(int numPlayers, Card allCards[MAX\_PLAYERS][DECK]);

void selectCard(int numPlayers, Card allPlayersCards[MAX\_PLAYERS][DECK], Card selectedCards[MAX\_PLAYERS]);

void getScores(int numPlayers, Card selectedCards[MAX\_PLAYERS], int scores[MAX\_PLAYERS]);

void showWinner(int numPlayers, int totalScores[MAX\_PLAYERS]);

void startNewGame();

//Global Variables

int roundTie = 0;

int previousRoundTieChecker = 0;

int currentRound = 0;

void main()

{

srand(time(NULL));

Player allPlayers[MAX\_PLAYERS];

Card allPlayersCards[MAX\_PLAYERS][DECK];

Card selectedCards[MAX\_PLAYERS];

int totalScores[MAX\_PLAYERS] = { 0,0,0,0 };

int choiceForNewGame;

char saveCheck;

char safeCheck;

int option = 0;

//Welcome screen

printf("Hong Wei Phang G00388774\n\n");

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

printf(" Welcome to WAR\n");

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

//ask user to start a new game or load a previously saved game

printf("Do you wish to start a new game or load a previously saved game?\n\n\nPress 1 to start a new game\nPress 0 to load a previously saved game\n");

scanf("%d", &choiceForNewGame);

if (choiceForNewGame > 1 || choiceForNewGame < 0)

{

while (choiceForNewGame > 1 || choiceForNewGame < 0)

{

printf("Sorry please enter 1 or 2 to play: ");

scanf("%d", &choiceForNewGame);

}

}

system("cls");

//if user start a new game

if (choiceForNewGame == 1)

{

startNewGame();

}

}

//Functions

//==================================================================================

void createNewDeck(Card newDeck[4][DECK])

{

Card cards[SUIT][DECK] = {

{

{ "2", 2 },{ "3", 3 },{ "4", 4 },{ "5", 5 },{ "6", 6 },{ "7", 7 },{ "8", 8 },{ "9", 9 },

{ "10", 10 },{ "J", 11 },{ "Q", 12 },{ "K", 13 },{ "A", 14 }

},

{

{ "2", 2 },{ "3", 3 },{ "4", 4 },{ "5", 5 },{ "6", 6 },{ "7", 7 },{ "8", 8 },{ "9", 9 },

{ "10", 10 },{ "J", 11 },{ "Q", 12 },{ "K", 13 },{ "A", 14 }

},

{

{ "2", 2 },{ "3", 3 },{ "4", 4 },{ "5", 5 },{ "6", 6 },{ "7", 7 },{ "8", 8 },{ "9", 9 },

{ "10", 10 },{ "J", 11 },{ "Q", 12 },{ "K", 13 },{ "A", 14 }

},

{

{ "2", 2 },{ "3", 3 },{ "4", 4 },{ "5", 5 },{ "6", 6 },{ "7", 7 },{ "8", 8 },{ "9", 9 },

{ "10", 10 },{ "J", 11 },{ "Q", 12 },{ "K", 13 },{ "A", 14 }

}

};

//copy to new array

for (int i = 0; i < SUIT; i++)

{

for (int j = 0; j < DECK; j++)

{

newDeck[i][j] = cards[i][j];

}

}

}

void dealRandomCards(Card playersHand[DECK])

{

//Create a new deck for player

Card newDeck[4][DECK];

createNewDeck(newDeck);

//Distribute random cards into the players deck

for (int i = 0; i < DECK; i++)

{

int ran = rand();

Card random = newDeck[ran % 4][ran % DECK];

if (random.value > 0)

{

playersHand[i] = random;

newDeck[ran % 4][ran % DECK].value = 0;

}

else

{

i--;

}

}

}

void setPlayerInfo(int numPlayer, Player playerList[MAX\_PLAYERS])

{

for (int i = 0; i < numPlayer; i++)

{

Player p1;

p1.id = i;

playerList[i] = p1;

}

}

void deal(int numPlayers, Card allCards[MAX\_PLAYERS][DECK])

{

for (int i = 0; i < numPlayers; i++)

{

Card playersHand[DECK];

dealRandomCards(playersHand);

for (int j = 0; j < DECK; j++)

{

//allCards[i][j].name[3] = '\0';

allCards[i][j] = playersHand[j];

}

}

}

void selectCard(int numPlayers, Card allCards[MAX\_PLAYERS][DECK], Card selectedCards[MAX\_PLAYERS])

{

//Local Variables

int selected;

char safeCheck;

int option;

for (int i = 0; i < numPlayers; i++)

{

printf("Player %d press Enter to continue\n", i + 1);

scanf\_s("%c", &safeCheck);

//Clear console

system("cls");

//show Cards

for (int j = 0; j < DECK; j++)

{

printf("%2s ", allCards[i][j].name);

}

//prompt user to select a card

printf("\nPlayer %d select a card by its position (e.g 1st, 2nd etc.): \n", i + 1);

scanf\_s("%d", &selected);

selected--;

while (selected > 12 || selected < 0)

{

printf("There are only 13 cards in your hand.\nTry Again: ");

scanf\_s("%d", &selected);

selected--;

}

selectedCards[i] = allCards[i][selected];

//Check

printf("\nYou've selected card %s\n\n", allCards[i][selected].name);

printf("Player %d press Enter to continue\n", i + 1);

scanf\_s("%c", &safeCheck);

system("cls");

}

}

void getScores(int numPlayers, Card selectedCards[MAX\_PLAYERS], int scores[MAX\_PLAYERS])

{

//Local Variables

int safeCheck;

int sameHolder = 0;

int same;

int highest = 0;

int highPlayer = 999;

int winPoints = 0;

int checker[MAX\_PLAYERS] = { 0,0,0,0 };

//Check for duplicates

for (same = 2; same < 15; same++)

{

for (int i = 0; i < numPlayers; i++)

{

if (selectedCards[i].value == same)

{

//Counts number of duplicates

sameHolder++;

}

}

if (sameHolder > 1)

{

for (int i = 0; i < numPlayers; i++)

{

if (selectedCards[i].value == same)

{

//Players position in this array identifies if they have a duplicate

checker[i] = sameHolder;

}

}

}

//Reset

sameHolder = 0;

}//Duplicate

//Check for highest value

for (int i = 0; i < numPlayers; i++)

{

//Getting total point from round

winPoints += selectedCards[i].value;

if (checker[i] <= 1) //No duplicates

{

if (selectedCards[i].value > highest)

{

highest = selectedCards[i].value;

highPlayer = i;

}

}

}//highest

//Check if previous round had a tie

if (highest == 0)

{

roundTie = winPoints;

previousRoundTieChecker = 1;

}

else

{

if (previousRoundTieChecker = 1)

{

winPoints += roundTie;

previousRoundTieChecker = 0;

}

//Declare Winner of round

scores[highPlayer] += winPoints;

}

currentRound++;

//Header for scoreboard

printf("Players \tCards\tPoints\tTotal Score\n");

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

for (int i = 0; i < numPlayers; i++)

{

printf("Player %d:\t%s \t%d \t%d\n", i + 1, selectedCards[i].name, selectedCards[i].value, scores[i]);

}

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

printf("Round %d\n", currentRound);

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

printf("Press Enter to continue to next round...\n");

scanf\_s("%c", &safeCheck);

}

void showWinner(int numPlayers, int totalScores[MAX\_PLAYERS])

{

//Local Variables

int highest = 0;

int highPlayer = 999;

//Header for scoreboard

printf("Placement \tPlayer\t\tPoints\n");

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

for (int i = 0; i < numPlayers; i++)

{

for (int j = 0; j < numPlayers; j++)

{

if (totalScores[j] > highest)

{

highest = totalScores[j];

highPlayer = j;

}

}

printf("%d.\t\tPlayer %d\t%d Points\n", i + 1, highPlayer + 1, highest);

totalScores[highPlayer] = -1;

highest = -1;

highPlayer = 999;

}

}

void startNewGame()

{

char safeCheck;

int numPlayers = 0;

Player allPlayers[MAX\_PLAYERS];

Card allPlayersCards[MAX\_PLAYERS][DECK];

Card selectedCards[MAX\_PLAYERS];

int totalScores[MAX\_PLAYERS] = { 0,0,0,0 };

printf("Please enter number of players (2 - 4): ");

scanf\_s("%d", &numPlayers);

//ensure between MAX and MIN

while (numPlayers < 2 || numPlayers > 4)

{

printf("\nMinimum is 2 and maximum is 4.\nEnter number of players: ");

scanf\_s("%d", &numPlayers);

}

setPlayerInfo(numPlayers, allPlayers);

printf("Player 1 press Enter to continue...\n");

scanf\_s("%c", &safeCheck);

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

{

deal(numPlayers, allPlayersCards);

//To clear the console

system("cls");

selectCard(numPlayers, allPlayersCards, selectedCards,totalScores);

system("cls");

getScores(numPlayers, selectedCards, totalScores);

}

system("cls");

showWinner(numPlayers, totalScores);

}