

# WAR CARD GAME DOCUMENTATION

1. Project Description
2. How is the game played?
  - 2.1.1 Setting up Visual Studio
  - 2.1.2 Creating classes
  - 2.1.3 Creating Enum
3. Screenshots of the Application
4. Improvements that can be made
5. Steps to run the project

## Getting Started

The main motive of this project is to implement the War game. It is one of the interesting card games which requires at least 2 players to start with. A standard deck of 52 cards is used to play this game. The deck of **52** cards involves:

- Clubs (9 cards from 2 to 10 and 4 cards of J, Q, K, A) - **13**
- Diamonds (9 cards from 2 to 10 and 4 cards of J, Q, K, A) - **13**
- Hearts (9 cards from 2 to 10 and 4 cards of J, Q, K, A) - **13**
- Spades (9 cards from 2 to 10 and 4 cards of J, Q, K, A) – **13**

## How is the game played?

The 52-card deck is distributed among the two players equally and each player draws a single card on every turn. If the value of the card drawn by the player 1 is greater than the value of the card drawn by the player 2, player 1 gets to keep both the cards and vice versa. The game is continued until one of the players runs out of the cards (If 2 players are playing) and that the player 2 get all the cards in the deck, he is declared the winner. If two cards drawn by both the players are equal, another card is drawn and again compared which player has the highest value of the card. This process repeats until a winner is declared.

## Setting up visual studio

I have implemented this game in C# using visual studio Console App (.NET Core). The version of visual studio used is visual studio 2017.

For this project, I have created different classes. The starting point of the project is from the main method in **Program.cs** file.

## Creating Classes

### **Program.cs**

This file asks the user to either enter “s” to start the game or “q” to quit the game.

### **NotEmptyDeck.cs**

This file has the implementation of what happens if the below conditions occur. It first checks if both the player's deck has cards.

- If the player 1 card is greater than the player 2 card
- If the player 2 card is greater than the player 1 card
- If the player 1 card is equal to the player 2 card
- If the player 1 card is greater than the player 2 card

**If the player 1 card is greater than the player 2 card** – Both player1 and player2 cards are put back into the deck collection of player 1 using the method "back\_into\_deck"

**If the player 2 card is greater than the player 1 card** – Both player1 and player2 cards are put back into the deck collection of player 2 using the method "back\_into\_deck"

**If the player 1 card is equal to the player 2 card** – Another round of game is played to check the value of the pulled card again. The process repeats until one person has the card value greater than the other person.

### **EmptyDeck.cs**

It first checks if both the player's deck has cards. If the player 1 do not have cards left, player 2 is declared a winner. If the player 2 do not have cards left, player 1 is declared a winner.

### **PlayGame.cs**

This file has the code implementation for loading up the complete deck and as well as the deck's for player1 and player2 using the method "shapenumbercombo". After the decks are generated, they are shuffled using "shuffle" method. The player1 and player 2 decks are filled in alternate matter.

### **CardProperties.cs**

This file sets the value for shape and number for each card

### **Functionalities.cs**

This file has all the methods which implements the below functionalities

- Generating cards in a deck with shape and value for each card
- Shuffling the deck after it is generated using Random() function
- A method for players to draw a single card each from their deck of cards
- A method for players to draw three cards each from their deck of cards

- A method to place the cards back into the player's deck based on whoever has the higher value card.

### Creating Enum

#### **ShapeEnum.cs**


This file has the enum values for all the different shapes in a deck i.e. Hearts, Diamonds, Spades, and Clubs.

#### **NumberEnum .cs**

This file has the enum values for all the different number in a deck i.e. from 2 to 10 and J, Q, K, A


### Screenshots of the implemented project

1. The above screenshot asks if the user wants to start the game or quit

 C:\Program Files\dotnet\dotnet.exe


```
Choose one of the below options for the game
Press s and enter to start the game
Press q and enter to end the game
```

2. Quits the game if "q" is pressed

 C:\Program Files\dotnet\dotnet.exe

```
Choose one of the below options for the game
Press s and enter to start the game
Press q and enter to end the game
q
****
BYE
****
```

### 3. Starts the game if “s” is pressed

 C:\Program Files\dotnet\dotnet.exe

```
Choose one of the below options for the game
Press s and enter to start the game
Press q and enter to end the game
s
---- Press 'Enter' to enter into the WAR game and for both the players to draw the first card ----

Player 1 turn - Six Diamond.

Player 2 turn - Q Diamond.

Since Q is greater than Six, Player 2 wins this round and gets the card
*****
```

```
Player 1 turn - K spade.

Player 2 turn - Three spade.

Since K is greater than Three, Player 1 wins this round and gets the card
*****
```

```
Player 1 turn - A Diamond.

Player 2 turn - A Diamond.

Since A is equal to A, another card is drawn
*****
```

### Improvements that can be made

I would further add additional functionalities to cover the case where both the players has the same value of card. Also, I would also consider the weightage for the shapes of the cards (Hearts, Clubs, Diamonds, Spades) as well instead of only numbers. That is providing some value to the shapes as well in the ascending order in some sequence respectively and consider the value of the shape as well along with the value of the card.

### How to Run the project

- Download the zip folder of the project (WarCardGame)
- Open the project or import the project folder in visual studio
- Open the Program.cs file and run the project.