

# POKER APP

---

Julie Giles

30 September 2020

# FEATURES

- Receives player's name and keeps track up until 5 losses occur
- Deck of 52 shuffled cards which will be dealt in the order they are shuffled – no repeats in a game
- Detects Poker combinations based on the player's hand and the 5 cards on the table
- Visual representation of cards dealt (including card dealt face down)

# GAME LOGIC

```
cards = [
    [{2d: 1}, {2c: 2}, {2h: 3}, {2s: 4}], ,
    [{3d: 5}, {3c: 6}, {3h: 7}, {3s: 8}],
    [{4d: 9}, {4c: 10}, {4h: 11}, {4s: 12}],
    [{5d: 13}, {5c: 14}, {5h: 15}, {5s: 16}],
    [{6d: 17}, {6c: 18}, {6h: 19}, {6s: 20}],
    [{7d: 21}, {7c: 22}, {7h: 23}, {7s: 24}],
    [{8d: 25}, {8c: 26}, {8h: 27}, {8s: 28}],
    [{9d: 29}, {9c: 30}, {9h: 31}, {9s: 32}],
    [{10d: 33}, {10c: 34}, {10h: 35}, {10s: 36}],
    [{Jd: 37}, {Jc: 38}, {Jh: 39}, {Js: 40}],
    [{Qd: 41}, {Qc: 42}, {Qh: 43}, {Qs: 44}],
    [{Kd: 45}, {Kc: 46}, {Kh: 47}, {Ks: 48}],
    [{Ad: 49}, {Ac: 50}, {Ah: 51}, {As: 52}]
]
```

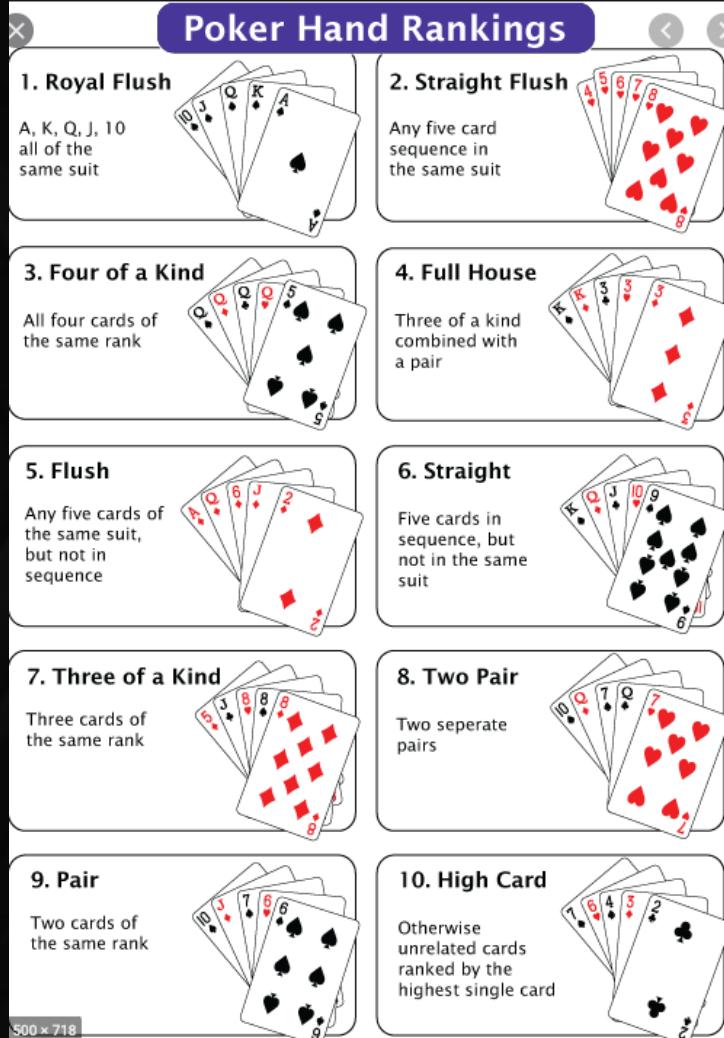


- Card object
- Deck object (array of 52 cards)

```
order = [
    ['', ''],
    [2, :♦], [2, :♣], [2, :♥], [2, :♠],
    [3, :♦], [3, :♣], [3, :♥], [3, :♠],
    [4, :♦], [4, :♣], [4, :♥], [4, :♠],
    [5, :♦], [5, :♣], [5, :♥], [5, :♠],
    [6, :♦], [6, :♣], [6, :♥], [6, :♠],
    [7, :♦], [7, :♣], [7, :♥], [7, :♠],
    [8, :♦], [8, :♣], [8, :♥], [8, :♠],
    [9, :♦], [9, :♣], [9, :♥], [9, :♠],
    [10, :♦], [10, :♣], [10, :♥], [10, :♠],
    ['J', :♦], ['J', :♣], ['J', :♥], ['J', :♠],
    ['Q', :♦], ['Q', :♣], ['Q', :♥], ['Q', :♠],
    ['K', :♦], ['K', :♣], ['K', :♥], ['K', :♠],
    ['A', :♦], ['A', :♣], ['A', :♥], ['A', :♠]
]
p order.index([player_hand[0].value, player_hand[0].suit])
```

```
if player_scores.combination(5).find{|a,b,c,d,e| a+b+c+d+e == 205 ||
a+b+c+d+e == 210 || a+b+c+d+e == 215 || a+b+c+d+e == 220}
```

# GAME LOGIC



## 1. Royal flush:

```
if player_scores.combination(5).find{|a,b,c,d,e| a+b+c+d+e == 205 ||  
a+b+c+d+e == 210 || a+b+c+d+e == 215 || a+b+c+d+e == 220}
```

## 2. Straight flush:

```
elsif (player_card_number.each_cons(5).find{|a| a[1] - a[0] == 4})  
&& (player_card_suits.combination(5).find{|a,b,c,d,e| (a==b) && (b==c) && (c==d) && (d==e)})
```

## 3. Four of a kind:

```
player_card_number.combination(4).find{|a,b,c,d| a+b+c+d == a* 4}
```

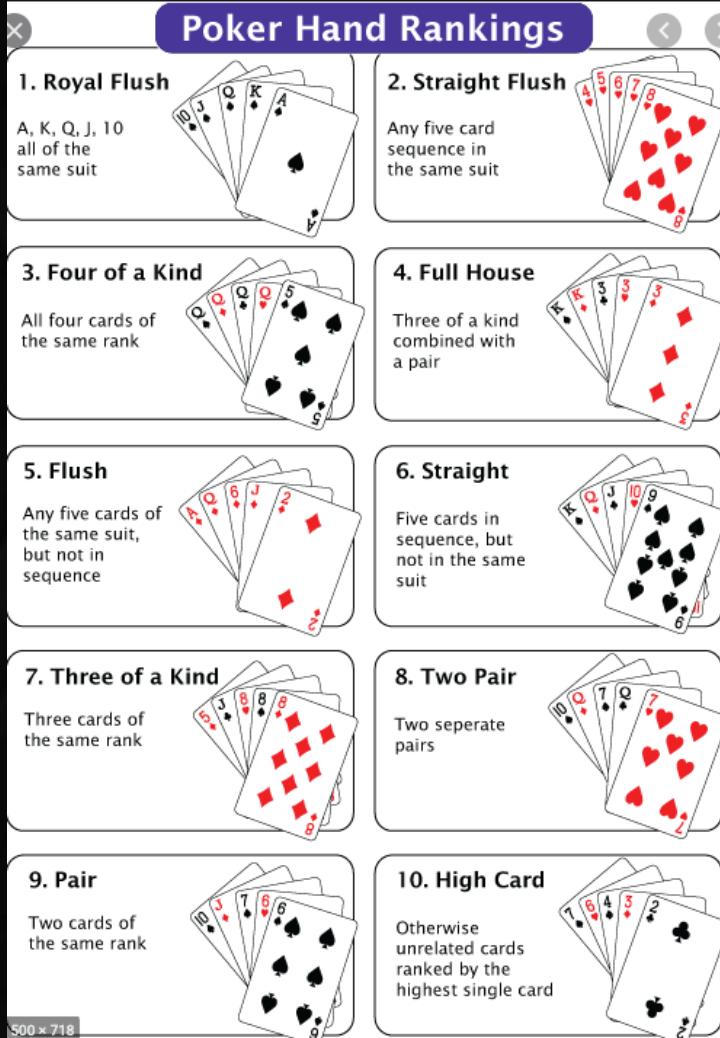
## 4. Full House:

```
((player_card_number.combination(3).find{|a,b,c, d, e| (a+b+c ==a*3) && (d+e == d*2)} ))
```

## 5. Flush:

```
player_card_suits.combination(5).find{|a,b,c,d,e| (a==b) && (b==c) && (c==d) && (d==e)}
```

# GAME LOGIC



## 6. Straight

```
player_card_number.each_cons(5).find { |a| ((a[1] - a[0] == 1) && (a[2] - a[1] == 1) && (a[3] - a[2] == 1) && (a[4] - a[3] == 1)) }
```

## 7. Three of a kind

```
player_card_number.combination(3).find { |a,b,c| a+b+c == a* 3 }
```

## 8. Two pair

```
player_card_number.combination(4).find { |a,b,c,d| (a+b == a* 2) && (c+d == c*2) }
```

## 9. Pair

```
player_card_number.combination(2).find { |a,b| a+b == a* 2 }
```

# REVIEW

## Challenges:

- MVC
- Visually displaying cards in command line
- Animation sequence by iterating through arrays containing ASCII art

## Favourite:

- Game logic