Project 2

Title
Go Fish

course
CIS-17A

Author
Paola Alcala

Game introduction

52 cards
4 suits: Hearts, Spades, Diamonds, Clubs.
13 ranks: Ace, One, Two, Three, Four, Five, Six, Seven, Eight, Nine, Ten, Jack, Queen, King.

The cards are divided in suits and ranks. The goal is to collect as many complete sets of 2 suit matching cards. Collect cards needed from players or go fishing from the remaining cards.

How to play Go Fish

1. Deal 7 cards to each player
2. Player 1 chooses a player and asks for a card.
3. Either one of each players gives up the card or says "Go Fish".
4. If player is told to "go fish" they grab a card from the center pile and add it to their hand.
5. If player gets a set of 2 matching cards remove from hand and add to matching cards pile.
6. The player's turn will end when they went "fishing" for a card.
7. If player secures a card from another player the get another turn.
8. Continue until one player runs out of cards.
9. The game is won by the player who has collected the most matching set of 4 cards.

Development Summary:

Lines of code: 505
Comment Lines: 180
Total Lines of code: 685

Version 1:

InitInitial set up of the Card, Deck, and Player classes with their respective .cpp files and .h files. The card class handles individual cards with with its attributes suit, face, and value. The player class has attributes for name and hand. The main focus is ensuring the deck prints correctly.

Version 2:

Focused on developing the Player class and its corresponding .cpp file. Implemented a feature that allows the user to choose between playing against another player or the computer. Redesigned the Deck structure and its .cpp file, ensuring all deck-related functions are organized and handled within the deck implementation.

Version 3:

Manipulating the deck by implementing a shuffle function to randomize the cards. Added functionality to distribute the cards from the deck into player's hands. These changes will create a relationship between the deck and player classes.

Version 4:

Introduced a Dealer structure with its corresponding .cpp and .h files. Established connections between the Player, Hand, and Dealer files to enable the dealer to distribute cards to each player. This will create a relationship between the player and dealer classes.

Versions 5:

Implemented functionality to compare cards in each player's hand to identify matches. Added a function in the Hand class to search for and track matching cards. Created a variable in the Player class to hold cards with matches and updated the Player class to include a score tracker.

Version 6:

Added a feature where players can request cards from each other. This lead to added the overall game logic. If a match is found, the card is removed from both players' decks and their updated decks are displayed.
If no match is found, the player is prompted to "Go Fish," retrieving a card from the center pile. The player's deck is then updated and displayed with the new card. Added functionality to announce whether the requested card is present in the other player's deck

Version 7:

Automated the computer player by introducing abstract classes and leveraging inheritance. Created a base class to define common player behaviors and implemented a derived class for the computer player with automated decision-making logic.

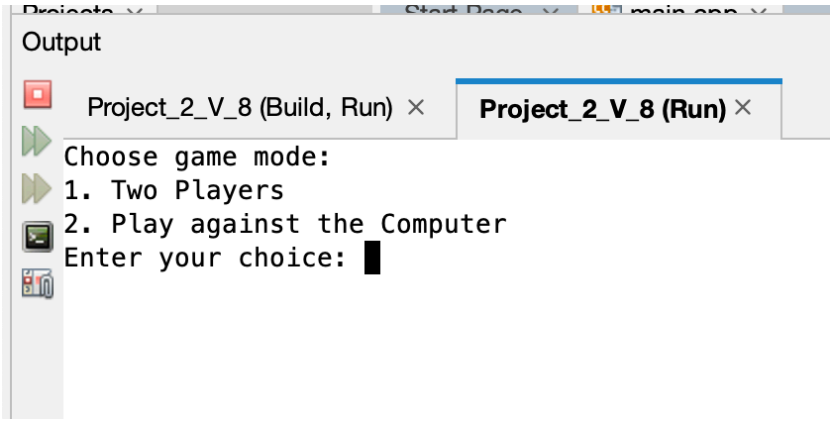
Version 8:

Added an array of objects to compare player scores and determine the winner. Included the functionality to announce the result. Implemented friend relationship for the Dealer and Deck classes to improve collaboration between these components. Added a copy constructor and operator overloading to the Card class in Card.h to support effective copying and comparison of the card objects.

Version 9:

Attempted to implement additional advanced class concepts into the game; however, this version does not function as intended.
Please refer to Version 8 for the last fully operational version of the game.

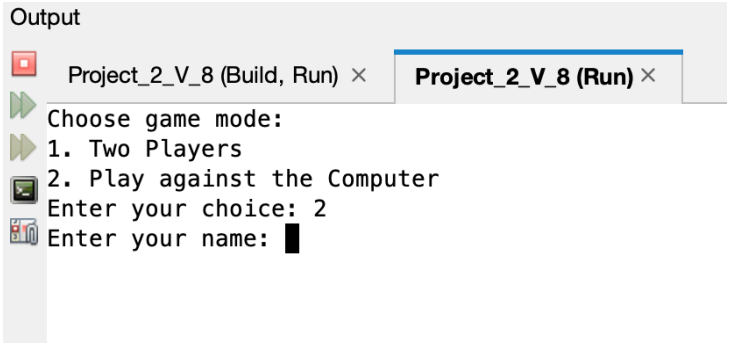
Example Inputs and Outputs version 8



example output:
"Choose game mode:"

example input: 2

example output:
"Enter your name:"



example input: Paola

example output:

```
Dealing cards to players...
paola's Hand:
Hand contains:
Face: A of Suit: S Value: 1
Face: 3 of Suit: S Value: 3
Face: 5 of Suit: S Value: 5
Face: 7 of Suit: S Value: 7
Face: 9 of Suit: S Value: 9
Computer's Hand:
Hand contains:
Face: 2 of Suit: S Value: 2
Face: 4 of Suit: S Value: 4
Face: 6 of Suit: S Value: 6
Face: 8 of Suit: S Value: 8
Face: T of Suit: S Value: 10

Checking for initial matches...
paola's Matches Deck:
Hand contains:
Computer's Matches Deck:
Hand contains:
```

example output:
Shows you each hand. Searches for matches in initial deck.
Takes a score of initial matches

```
paola's turn.
Your hand:
Hand contains:
Face: A of Suit: S Value: 1
Face: 3 of Suit: S Value: 3
Face: 5 of Suit: S Value: 5
Face: 7 of Suit: S Value: 7
Face: 9 of Suit: S Value: 9
Ask for a card value (1-10):
```

example output:
Shows your updated deck.
"Ask for a card value (1-10):"

example input: 3

example output:
Example of when a player does not have the card asked for.

example input:
"Computer does not have any threes. Go Fish!"

```
Ask for a card value (1-10): 3
Computer does not have the card. Go Fish!
paola drew a card: J of S
paola's Matches Deck: 0 sets
Computer's Matches Deck: 0 sets

Computer's turn.
Your hand:
Hand contains:
Face: 2 of Suit: S Value: 2
Face: 4 of Suit: S Value: 4
Face: 6 of Suit: S Value: 6
Face: 8 of Suit: S Value: 8
Face: T of Suit: S Value: 10
Computer asks for card value: 2
paola does not have the card. Go Fish!
Computer drew a card: Q of S
paola's Matches Deck: 0 sets
Computer's Matches Deck: 0 sets

paola's turn.
Your hand:
Hand contains:
Face: A of Suit: S Value: 1
Face: 3 of Suit: S Value: 3
Face: 5 of Suit: S Value: 5
Face: 7 of Suit: S Value: 7
Face: 9 of Suit: S Value: 9
Face: J of Suit: S Value: 10
```

```
paola's turn.
Your hand:
Hand contains:
Face: A of Suit: S Value: 1
Face: 3 of Suit: S Value: 3
Face: 5 of Suit: S Value: 5
Face: 7 of Suit: S Value: 7
Face: 9 of Suit: S Value: 9
Face: J of Suit: S Value: 10
Ask for a card value (1-10): 10
Computer has the card! Transferring...

Matches found this turn:
Hand contains:
Face: J of Suit: S Value: 10
Face: T of Suit: S Value: 10
paola's Matches Deck: 1 sets
Computer's Matches Deck: 0 sets

paola's turn.
Your hand:
Hand contains:
Face: A of Suit: S Value: 1
Face: 3 of Suit: S Value: 3
Face: 5 of Suit: S Value: 5
Face: 7 of Suit: S Value: 7
Face: 9 of Suit: S Value: 9
Ask for a card value (1-10):
```

example output:
Example of when a player does have the card asked for.

example input:
"Computer has the card! Transferring...'

example output:
Example of when a player has an empty hand and the game ends.
The program compares scores and announces a winner.

```
----------------------------------------------------
Paola's hand after being asked: Paola's hand: four of Diamonds
-----------------------------------
Paola's turn.
Current hand: Paola's hand: four of Diamonds
Enter a rank to ask for (e.g., Ace, Two, Three): four
Paola asks Sunny for cards of rank four.

****************************************************
Sunny does not have any fours. Paola goes fishing!
Paola drew a four of Clubs from the center pile.
New match found in Paola's hand: four of Diamonds and four of Clubs
----------------------------------------------------
Paola's hand after asking: Paola's hand:
Paola's updated score: Paola's Score: 8

----------------------------------------------------
Sunny's hand after being asked: Sunny's hand: queen of Hearts, jack of Spades, seven of Clubs
Game over! Final scores:
Paola's Score: 8

Sunny's Score: 15

The winner is Sunny with 15 matches!

RUN FINISHED; exit value 0; real time: 21m 59s; user: 0ms; system: 0ms
```