ECE 175: Computer Programming for Engineering Applications

Final Project: Modified Go Fish

Due Date: 4 PM Monday December 4, 2023 on zyBooks

All due date: Individual/Group sign up - Tuesday November 21, 2023 by 11.59 pm
Final project - Monday December 4, 2023 at 4.00 pm

Administrative details:

1.1 Group: You can work on this final project **by yourself, or** you can **work in a group of (maximum) 2 students.** If you choose to work as a group, use collaborative tools to facilitate the project development, such as replit.com, google doc, etc.
Sign up your team or if you will work by yourself for the final project
https://forms.gle/n8y6HgY4WYWCcfVS6
The team/group must be signed-up by 11.59 pm on November 21, 2023. No new group can be formed after this date (unless approved by an instructor). You can check that your group signed up properly at
https://docs.google.com/spreadsheets/d/1iKv2K3T6PJbJouMY38UHazWJ3BSbUa66jqSY8wbQ8rA/edit?usp=sharing

1.2 Final Code Submission: The project is due on ***Monday December 4, 2023 at 4.00 pm***.
Both team members must submit the code on zyBooks.

Late submission policy: 20% deduction per day.

The **last day of final project submission is *by 4.00 PM on Wednesday December 6, 2023*.** After this date/time, the final project will NOT be accepted/graded.

1.3 Academic Integrity Policy: Each team is expected to submit its own code. You may ask others for advice and/or discuss the project in general. However, you/your group MUST

WRITE YOUR OWN CODE.

If *any part* of the code submitted by different students or groups of students is identical, ALL involved parties will receive **0** credit on the entire project and one letter reduction in their final grade. **This policy will be very aggressively enforced.** ALL submitted code will be checked with a plagiarism detection tool on zyBooks.

1.4 *Points distribution:* See Rubric (page 6 and 7) for final project grading.

1.5 Suggestions

a) Spend time designing your code and use modular programming. Create all function prototypes and describe what they do before developing your code. Create pseudocode of your program.
b) Determine test cases for your functions and your overall code to ensure proper functionality.
c) Write well-documented code.

Information:

Go Fish (https://en.wikipedia.org/wiki/Go_Fish) is a card game that can be played by two or more players. Each player takes turn asking for a card with a specific rank or face from another player. The goal is to form a book (a set of 4 cards with the same face or rank). The winner is the player who has the most number of books at the end of the game.

For this modified Go Fish card game, only face A and 2-9 will be used.

a) For two players, each player starts **with 6 cards**. The rest of the cards are put in the center so that each player can take a card when needed.
Let's call two players: Wilma (a user) and PC (a computer). After each player gets 6 cards on his/her hand, the players take turn asking for the card.

b) Wilma first asks PC for the card of a particular rank or face value
- If PC has card(s) with that rank,
  - PC has to give all the cards with that rank to Wilma and
  - Wilma gets another turn (to ask for a card).
- If PC has no card with that rank,
  - PC says "Go Fish" and
  - Wilma has to take one card from the center pile and put it in the hand.
  If the face of the card drawn from the center pile is the face that Wilma originally asked for, she gets another turn (to ask for a card). Otherwise, it is PC's turn.

c) When it's PC turn to ask for a card from Wilma, same rules above are applied.

d) The game continues until all 9 books (A, 2-9) are found. The winner is the player with the most numbers of books.

Read more details of the game/implementation in Project requirements below.

Project requirements: For this project, your team will write the card game – ***Modified*** Go Fish, for which you will use only 36 cards of a standard deck of 52 cards. There will be only 2 players, the computer and one user. After each player gets 6 cards on his/her hand, the user will start first.

Not using linked lists in your final project code results in 0 point for this final project.

To simulate the deck of 36 cards, and each of the hands, **your MUST use a dynamic list of cards** with the following struct type:

```
typedef struct card_s {
        char suit;
        int face;
        struct card_s *next;
} card;
```

Each card consists of a suit [**c**lubs (♣), **s**pades (♠), **h**earts (♥), or **d**iamonds (♦))
A face (1(Ace) and 2 – 9

Note: you can modify the above struct (i.e., changing the data type of member(s), add more members,) but you cannot remove the already existing members in the given struct.

1) The 36-card deck must be implemented using the linked list. Each hand (computer or user) must be implemented using the linked list.

2) Your program then shuffles the deck. One algorithm that you may use to shuffle the deck.

for at least 100 times

for j = 1 to 37

generate one random number between 1 – 37

if the random number is not j

swap the content of these two nodes

Notes:

a) The algorithm assumes that you implement the linked list with a dummy head node and 36 cards (a total of 37 nodes in the linked list). When j = 1, it points to the second node (the first actual card) in the list.

b) To swap the content of the two nodes,

- the 1st pointer points to say node j (observe that when j = 1, it points to the first card (second node) in the linked list). Assume that this 1st pointer points to a node containing A of diamonds(A♦),
- when you get one random number say 24, move the 2nd pointer to point to node 24 in the list. Assume that this 2nd pointer points to a node containing 9 of clubs (9♣),
- After the swap, the content of these two nodes, the 1st pointer now points to a node containing 9♣ and the 2nd pointer points to a node containing A♦. These pointers do not move.

Your code must have a ***ShuffleCard( ) function*** and should work with *any size deck of cards*, i.e., shuffling 37 cards, 20 cards or 5 cards. You should seed the random number generator with a call to time() with srand(). [see **sec 2.24 Random numbers in your zyBooks**].

3) After shuffling the deck, deal the cards by giving one card to the user/player, followed by one card to the computer, followed by one card to the user, etc. until each player/hand has 6 cards. The player (computer)'s hand is represented as a dynamic list of cards. The list is populated with the cards drawn by the player (computer).

Note:

a) Card(s) added to each of the player/computer's hand (drawn from the deck) must be added to that player/computer's linked list correctly and MUST be removed from the deck (if removed from the deck, free() function should be used appropriately).

b) Cards removed from each of the player/computer's hand MUST BE deleted from that player/computer's linked list correctly (and free() function should be used appropriately).

4) The game starts with the player. The player asks the computer for the card of a particular rank or face value.

your code should check whether the user enters a valid card (e.g., valid face or rank value (A and 2-9) – if not, ask a user to enter again

If the computer has card(s) with that face/rank, it has to give all cards with that face/rank to the player and the player gets another turn (to ask for a card).

If the computer has no card with that rank, the computer says "Go Fish" and the player has to take one card from the center pile and put it in his/her hand. If the card drawn from the center pile is the card that the player originally asked for, he/she gets another turn (to ask for a card).

5) When the card(s) are put into the hand (whether it is a user or a computer), if the hand has all 4 cards of the same face or rank value (the 4 cards form a "book", i.e., 9 of diamonds (♦), 9 of spade (♠), 9 of club (♣), 9 of heart (♥)),
   - The book of that face is put into that player. The book must be shown face up (to show that the player gets that book).
   - These 4 cards must be removed from the hand. free( ) or delete node function (if you have one) must be used properly.

6) Next is the computer's turn (same rules given in step 4) are applied).
   Computer with no strategy: randomly generate the face value that the computer will ask (from a user).

7) After the computer asks for a card with a specific face/rank value, there should be statements to ask whether a user has card(s) with that face value. Note: you can improve this in your code to make it more user friendly, but the idea is below.

   For example, if a computer asks for a card with 9,

   Do you have card(s) with 9?
   Enter y to give the card(s) with 9 or n to say 'Go Fish': n

   Scenarios:
   If a user enters n, the screen displays "Go Fish" and the computer then takes a card from the center pile.
   If a user enters y,
   - and if a user has the card, the card(s) are then given to the computer.
   - BUT the user's hand has no card asked by the computer, the screen displays "You do NOT have card(s) with 9" and "Go Fish" and the computer then takes a card from the center pile.

   Note: 9 is used in this example but it can be any face that the computer asked for.

8) When one round (both a user and a computer get to play) is completed, the deck should be shuffled (to emulate a "random" pile for the center pile).

9) When one round (both a user and a computer get to play) is completed or at any given time the player (whether it is a computer or a user) has no card left on the hand, it has to take 6 cards (or as many cards the center pile has available at that time) from the center pile to continue playing.

10) The game continues until all 9 books (A, 2-9) are found. The winner is the player with the most numbers of books.

11) At the end of the game, your code should announce the winner.

See sample code executions on pages 8 – 24

- **Game 1: page 8 - 12**
- **Game 2: page 13 - 19**
- **Game 3: page 19 – 24**

Optional Features for extra credit:

***Note:** Each optional feature must be correctly and fully functional in your code in order to get the extra credit for that part.*

No partial credit will be given for "somewhat" functioning bonus feature(s).

Make sure that your program is working correctly before attempting to do these bonus features.

1) Graphics: Add graphics to your game.
   (+2 pt) when displaying a hand, you can make each card display (at least) as shown

```
----------      ----------
|9♥    |      |A♣    |
|       |      |       |
|       |      |       |
|   9♥ |      |   A♣|
----------      ---------
```

2) (+4 pts) Computer Strategy
   Instead of randomly generating a face value to ask for a card (from a user) (Step 6 in Project requirements), the computer will have a strategy to ask for a card. Below is one suggestion (you/your team is encouraged to come up with a smarter strategy):
   a) The computer asks for a card with one of the face values that is on the computer's hand.
   For example, if the computer's hand is 9♣, 8♥, A♠, the computer should ask for 9 or 8 or A.

   b) The computer should not ask for the card(s) that are already a book.
   For example, if the user's book(s) are 6, 2, 7 and the computer's book(s) are 9.
   The computer should not ask for card 6, 2, 7 or 9.

3) (+8 pts) Write a program for 3-player Go Fish. There will be two players and one computer. **The linked list MUST be used for the deck and all player's and computer hands -> NO extra credit will be given for this part if you/your team implements the 3-player Go Fish code without using the linked list.**

No partial credit will be given for "somewhat" functioning bonus feature(s).

ECE175--Modified Go Fish
Grading Rubric for Final Project

As stated in the final project handout, linked list must be used for this project, otherwise 0 point for this final project

	Criteria	Maximum Points	Exemplary (100%)	Proficient (75%)	Marginal (20%)	Unacceptable (0%)
1	Create Deck	5	The deck of 36 cards is created correctly using the linked list		The deck is missing card(s)	The deck is not created
2	Deck Shuffling	4	Deck is properly shuffled to a random deck	Deck is partially shuffled - the deck does not appear sufficiently random	Deck shuffling breaks upon execution	The shuffling is not done corrrectly or No Shuffling
3	Card Dealing	6	The 6 cards are dealt to each player to start the game. Cards are dealt in alternate order to each player.	The right number of cards is dealt, but cards are not dealt in alternate order	Cards are not dealt correctly	Card dealing is not implemented
4	Shuffling during the game	2	Every round (after both user and computer get to play), the deck is shuffled (with proper number of cards in the deck)		Deck shuffling breaks upon execution	No shuffle done
5	Printing during the game	6	During the game, the player's ***hand*** and ***book(s)*** are printed correctly and the computer's ***book(s)*** are printed correctly		There was an attempt to do this but not correctly function	No implementation
6	When it is a computer turn, it can ask for a "valid" card	5	The computer correctly asks for a "valid" card (2-9, A)	75% working (missing 1-2 face values)	There was an attempt to do this but not correctly function	No implementation
7	Let a user pick a card	3	The program only allows a user to enter 2 - 9, and A. If a user does not, continue asking until the user enters a valid card.	Mainly working but not 100% correct	There was an attempt to do this but not correctly function	No implementation
9	The "hand" does not get a card that it asks for	6	If the "hand" does not get a card that it asks for, 1) it says "go fish", 2) (3 pts) it takes a card from the center pile correctly.	75% working (no "go fish" but cannot correctly take a card from the pile)	say Go Fish but NOT taking a card correctly	No implementation
10	The turn when the "hand" does not get a card that it asks for	2	If the "hand" does not get a card that it asks for, the turn is changed to the other hand (user to computer turn or vice versa)		There was an attempt to do this but not correctly function	No turn change or not correctly implemented
11	Get a card that the "hand" (user or computer) asks for	7	If the "hand" gets the card that it asks for, the card can be removed from the other hand correctly and the card is added to the "hand" correctly	If the "hand" gets the card that it asks for, the card can be removed from the other hand correctly BUT the card is NOT added to the "hand" correctly	this operation breaks execution (code stops working)	No implementaion
12	The turn when the "hand" gets the card that it asks for or when the card drawn from the center pile has the same face value that it originally asked for	4	The "hand" continues to get the turn as long as 1) it gets the card that it asks for or 2) it takes a card from the center pile and the drawn card (from the center pile) has the same face value that the "hand" originally asks for	Correctly implement this but miss one scenario described in the 'exemplary' box on the left	There was an attempt to do this but not correctly function or not working	No implementaion
13	Card Matching (find a book)	3	the book (set of 4 cards with the same face value) can be found		There was an attempt to do this but not correctly function or not working	No implementaion
14	Remove card from the hands and the deck	6	The card on the computer and player's hand can be removed correctly when the book is found.	there is a major attempt on this part and it is 75% working. If free() is not used or used but make the code not working correctly (-3 pts)	There was an attempt to do this but not correctly function or not working	No implementaion
15	Check the user's answer	2	After the computer asks for a card with a specific face/rank value, there is a code to check when a user say y even though there is no such card on the hand		There was an attempt to do this but not correctly function or not working	No implementaion
16	Deal with no card left on the hand but the game is not end yet	8	If the hand has no card left but the game is still going, the cards should be drawn from the deck apropriately (6 cards if the deck has enough cards or as many cards that the deck has at that time). The hand is correctly created (again) and the cards are removed correctly from the deck	there is a major attempt on this part and it is 75% working. If free() is not used or used but make the code not working correctly (-3 pts)	There was an attempt to do this but not correctly function or not working	No implementaion

17	No card left on the deck	3	When the deck has no card left, the program has a way to prevent card being drawn from the deck	there is a major attempt on this part and it is 75% working	There was an attempt to do this but not correctly function or not working	No implementaion
18	deallocate memory	4	free( ) or delete node function is used properly when the node is no longer used		There was an attempt to do this but the code breaks	No implementaion
19	Winner annouce and books displayes	4	At the end of the game, the winner is decided correctly and announced (2 pts). The books of each player is displayed correctly (2 pts).	there is a major attempt on this part and it is 75% working.	There was an attempt to do this but not correctly function or not working	No implementaion
20	Game Interface	5	(3 pts) The interface is intuitive. The user is able to play the game with minimal intructions. Transitions from computer's turn and player's turn (vice versa) are clear.	The interface is intuitive for the most part. Some difficulty in understanding the game navigation.	The interface is counter-intuitive. Navigation options are not clearly stated. Interface limitations prevent proper game functionality.	The interface is very basic and does not allow transitions between turn.
	Program Design					
21	Code modularity	5	The code is logically divided to several functions that implement **important** functionality	The code is modular but further simplification could have been attempted	The code only has a few functions (ones from the Zybook) - there is no atempt in creating their own user-defined function	Code is not modular (all statements are written in main)
22	Code documentation	3	The code is properly documented. The input/output and goal of every function is adequately described. Comments are provided for various parts of the code	The code is partially documented	The code is scarcely documented	No documentation is provided
23	Compilation	7	Code succesfully compiles without errors or warnings. The code does not hang while in execution	The code succesfully compiles, but some conditions may make it hang (-4 pts)		Code does not compile
Total		**100**				
	Extra Credit		**Exemplary (100%)**			
	Graphics	2	See Bonus points section of the final project handout for details			
	Computer Strategy	4	See Bonus points section of the final project handout for details			
	a program for 3-player Go Fish	8	**Linked list must be used for this 3-player part. If the code is implemented using array (no bonus points for this part)** See Bonus points section of the final project handout for details			
Total		**14**				

Sample Code Execution 1: Red entered by a user

Statements in purple in the example below are to explain how the game is played (NOT part of the output when your code is run)

You can always generate this test case if you create the deck as follows A♥ A♦ A♣ A♠ 2♥ 2♦ 2♣ 2♠ … (an order of 4 suits can be different) and you do not shuffle the deck.

You can use this test case to check several scenarios:

1) A player can continue when asking for a card that is available on the computer hand.
2) A book (4 cards with one face and all suits) can be found.
3) A record of book(s) can be kept for each hand.
4) Cards can be correctly drawn from the pile when the hand has no card left.
5) The winner can be decided and the game ends successfully.

Note: You are encouraged to make your output look more appealing than what is given below. However, your code should display very similar information in order for a user to be able to play the game.

Enter your name: **Wilma**

|@@@@ @@@@ @@@@ @@@@ @@@@ @@@@ @@@@|
Wilma, Let's play Go Fish
|@@@@ @@@@ @@@@ @@@@ @@@@ @@@@ @@@@|

Wilma's hand has 6 cards
3♣ 3♥ 2♣ 2♥ A♣ A♥

Computer's hand has 6 cards **<- for Debugging purpose (no display in actual game)**
3♠ 3♦ 2♠ 2♦ A♠ A♦

Deck after cards are distributed. **<- for Debugging purpose (no display in actual game)**
4♥ 4♦ 4♣ 4♠ 5♥ 5♦ 5♣ 5♠ 6♥ 6♦ 6♣ 6♠
7♥ 7♦ 7♣ 7♠ 8♥ 8♦ 8♣ 8♠ 9♥ 9♦ 9♣ 9♠

*** Type C/c to Continue ***: **c**

Wilma, your turn

Wilma's hand has 6 cards
3♣ 3♥ 2♣ 2♥ A♣ A♥

Which card (A, 2 - 9) do you want to ask for? **J**
Enter only A, 2 - 9 for the card
Which card (A, 2 - 9) do you want to ask for? **1**
Enter only A, 2 - 9 for the card
Which card (A, 2 - 9) do you want to ask for? **0**
Enter only A, 2 - 9 for the card
Which card (A, 2 - 9) do you want to ask for? **A**

Not a valid card, ask for another card

Taking card(s) from Computer's hand

Wilma's hand has 8 cards
A♦ A♠ 3♣ 3♥ 2♣ 2♥ A♣ A♥

Book of A is found on the hand **<- Found 4 cards of A on the hand**

*** Type C/c to Continue ***: **C**

Wilma, your turn **<- still Wilma's turn since Wilma got the cards from the computer**

Wilma's hand has 4 cards
3♣ 3♥ 2♣ 2♥

Wilma's book(s) are A

Which card (A, 2 - 9) do you want to ask for? **2**

Taking card(s) from Computer's hand

Wilma's hand has 6 cards
2♦ 2♠ 3♣ 3♥ 2♣ 2♥

Book of 2 is found on the hand **<- Found 4 cards of 2 on the hand**

*** Type C/c to Continue ***: **c**

Wilma, your turn **<- still Wilma's turn since Wilma got the cards from the computer**

Wilma's hand has 2 cards
3♣ 3♥

Wilma's book(s) are A 2 **<- Wilma now has 2 books of A and 2**

Which card (A, 2 - 9) do you want to ask for? **3**

Taking card(s) from Computer's hand

Wilma's hand has 4 cards
3♦ 3♠ 3♣ 3♥

Book of 3 is found on the hand **<- Wilma now has 2 books of A, 2, 3**

*** Type C/c to Continue ***: **c**

P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P

Wilma, You have no card left. **<- situation where there is no card left on Wilma's hand**

Pile has 24 cards

Draw 6 cards from the pile.

Check whether Wilma's newly drawn cards has a book

At this point, the pile has
4♥ 4♦ 4♣ 4♠ 5♥ 5♦ 5♣ 5♠ 6♥ 6♦ 6♣ 6♠
7♥ 7♦ 7♣ 7♠ 8♥ 8♦ 8♣ 8♠ 9♥ 9♦ 9♣ 9♠
The ones highlighted in blue are given to Wilma.

Book of 4 is found on the hand **<- situation where the cards that are just drawn has a book**
You now have 2 cards. **<-Wilma's hand is** 4♥ 4♦ 4♣ 4♠ 5♥ 5♦

Wilma's hand:
5♦ 5♥

Wilma's book(s) are A 2 3 4

P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P

*** Type C/c to Continue ***: **c**

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

Computer has no card left. **<- situation where there is no card left on the computer's hand**

Pile has 18 cards

Draw 6 cards from the pile.

At this point, the pile has
5♣ 5♠ 6♥ 6♦ 6♣ 6♠ 7♥ 7♦ 7♣ 7♠ 8♥ 8♦
8♣ 8♠ 9♥ 9♦ 9♣ 9♠
The ones highlighted in blue are given to Computer.

Check whether Computer's newly drawn cards has a book

Book of 6 is found on the hand **<- situation where the cards that are just drawn has a book**
Computer now has 2 cards. **<-Computer's hand is** 5♣ 5♠ 6♥ 6♦ 6♣ 6♠

Computer's hand: **<- for Debugging purpose (no display in actual game)**
5♠ 5♣

Computer's book(s) are 6

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

*** Type C/c to Continue ***: **c**

Wilma, your turn **<- still Wilma's turn since Wilma got the cards from the computer**

Wilma's hand has 2 cards
5♦ 5♥

Wilma's book(s) are A 2 3 4

Which card (A, 2 - 9) do you want to ask for? **5**

Taking card(s) from Computer's hand

Wilma's hand has 4 cards
5♣ 5♠ 5♦ 5♥

Book of 5 is found on the hand

*** Type C/c to Continue ***: **c**

P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P

Wilma, You have no card left.

Pile has 12 cards

At this point, the pile has
7♥ 7♦ 7♣ 7♠ 8♥ 8♦ 8♣ 8♠ 9♥ 9♦ 9♣ 9♠
The ones highlighted in blue are given to Wilma.

Draw 6 cards from the pile.

Check whether Wilma's newly drawn cards has a book

Book of 7 is found on the hand
You now have 2 cards.

Wilma's hand:
8♦ 8♥

Wilma's book(s) are A 2 3 4 5 7

P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P

*** Type C/c to Continue ***: **c**

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

Computer has no card left.

Pile has 6 cards

At this point, the pile has
8♣ 8♠ 9♥ 9♦ 9♣ 9♠
The ones highlighted in blue are given to Computer.

Draw 6 cards from the pile.

Check whether Computer's newly drawn cards has a book

Book of 9 is found on the hand
Computer now has 2 cards.

Computer's hand: **<- for Debugging purpose (no display in actual game)**
8♠ 8♣

Computer's book(s) are 6 9

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

*** Type C/c to Continue ***: **c**

Wilma, your turn

Wilma's hand has 2 cards
8♦ 8♥

Wilma's book(s) are A 2 3 4 5 7

Which card (A, 2 - 9) do you want to ask for? **8**

Taking card(s) from Computer's hand

Wilma's hand has 4 cards
8♣ 8♠ 8♦ 8♥

Book of 8 is found on the hand

Wilma's book(s) are A 2 3 4 5 7 8

Computer's book(s) are 6 9

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Congratulations, YOU won.

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Sample Code Execution 2: Red entered by a user

Statements in purple in the example below are to explain how the game is played (NOT part of the output when your code is run)

You can always generate this test case if you create the deck as follows A♥ A♦ A♣ A♠ 2♥ 2♦ 2♣ 2♠ … (an order of 4 suits can be different) and you **do not shuffle** the deck.

You can use this test case to check several scenarios:

1) Go fish scenario occurs when the requested card is not on the hand
2) A computer can continue when asking for a card that is available on the player hand.
3) A book (4 cards with one face and all suits) can be found.
4) A record of book(s) can be kept for each hand.
5) Cards can be correctly drawn from the pile when the hand has no card left.
6) The winner can be decided and the game ends successfully.

Note: You are encouraged to make your output look more appealing than what is given below. However, your code should display very similar information in order for a user to be able to play the game.

Enter your name: **KK**

|@@@@ @@@@ @@@@ @@@@ @@@@ @@@@ @@@@|
KK, Let's play Go Fish
|@@@@ @@@@ @@@@ @@@@ @@@@ @@@@ @@@@|

KK's hand has 6 cards
3♣ 3♥ 2♣ 2♥ A♣ A♥

Computer's hand has 6 cards **<- for Debugging purpose (no display in actual game)**
3♠ 3♦ 2♠ 2♦ A♠ A♦

Deck after cards are distributed. **<- for Debugging purpose (no display in actual game)**
4♥ 4♦ 4♣ 4♠ 5♥ 5♦ 5♣ 5♠ 6♥ 6♦ 6♣ 6♠
7♥ 7♦ 7♣ 7♠ 8♥ 8♦ 8♣ 8♠ 9♥ 9♦ 9♣ 9♠

*** Type C/c to Continue ***: **c**

KK, your turn

KK's hand has 6 cards
3♣ 3♥ 2♣ 2♥ A♣ A♥

Which card (A, 2 - 9) do you want to ask for? **8** **<- ask for a card that is NOT on the computer hand**

><(((('> ><(((('> ><(((('>
~~~ ~~~ Go Fish ~~~ ~~~ **<-you can just display "Go Fish" in your code**
><(((('> ><(((('> ><(((('>

KK, you take 4♥ from the center pile

KK's hand has 7 cards
~~~

4♥ 3♣ 3♥ 2♣ 2♥ A♣ A♥

*** Type C/c to Continue ***: **c**

Computer turn

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
3♠ 3♦ 2♠ 2♦ A♠ A♦

KK's hand has 7 cards
4♥ 3♣ 3♥ 2♣ 2♥ A♣ A♥

Do you have card(s) with 3? **<- the computer asks for 3 which are on the player hand**

Enter y to give the card(s) with 3 or n to say 'Go Fish': y

Remove card(s) with 3 from your hand **<- book of 3 is removed from the computer hand**

Book of 3 is found on the hand

*** Type C/c to Continue ***: **c**

Computer turn **<- still the computer's turn**

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
2♠ 2♦ A♠ A♦

Computer's book(s) are 3 **<- the computer has a book of 3**

KK's hand has 5 cards
4♥ 2♣ 2♥ A♣ A♥

Do you have card(s) with 2?

Enter y to give the card(s) with 2 or n to say 'Go Fish': **y**

Remove card(s) with 2 from your hand

Book of 2 is found on the hand

*** Type C/c to Continue ***: **c**

Computer turn

Computer (debugging purpose):
A♠ A♦

Computer's book(s) are 3 2 **<- the computer has a book of 3 and 2**

KK's hand has 3 cards
4♥ A♣ A♥

Do you have card(s) with A?

Enter y to give the card(s) with A or n to say 'Go Fish': **y**

Remove card(s) with A from your hand

Book of A is found on the hand **<- the computer has a book of 3, 2, A**

*** Type C/c to Continue ***: **c**

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

Computer has no card left. **<- need to draw more cards before the game continues**

Pile has 23 cards

Draw 6 cards from the pile.

Check whether Computer's newly drawn cards has a book
Computer now has 6 cards.

Computer's hand:
5♣ 5♦ 5♥ 4♠ 4♣ 4♦

Computer's book(s) are 3 2 A

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

*** Type C/c to Continue ***: **c**

Computer turn **<- still the computer's turn**

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
5♣ 5♦ 5♥ 4♠ 4♣ 4♦

Computer's book(s) are 3 2 A

KK's hand has 1 cards
4♥

Do you have card(s) with 5?

Enter y to give the card(s) with 5 or n to say 'Go Fish': **y**

You do NOT have card(s) with 5. **<- the player said y even though there is no 5 card on the hand**

><(((('> ><(((('> ><(((('>
~~~ ~~~ Go Fish ~~~ ~~~
><(((('> ><(((('> ><(((('>
~~~

Computer takes one card from the center pile

The card has the same face that Computer asked for, Computer will get another turn

Book of 5 is found on the hand **<- the cards just taken from the file forms a book of 5**

*** Type C/c to Continue ***: **c**

Computer turn

Computer (debugging purpose):
4♠ 4♣ 4♦

Computer's book(s) are 3 2 A 5

KK's hand has 1 cards
4♥

Do you have card(s) with 4?

Enter y to give the card(s) with 4 or n to say 'Go Fish': **y**

Remove card(s) with 4 from your hand

Book of 4 is found on the hand **<- the computer has a book of 3, 2, A, 5 and 4**

*** Type C/c to Continue ***: **c**

P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P

KK, You have no card left. **<- this to draw more cards before the game continues**

Pile has 16 cards

Draw 6 cards from the pile.

Check whether KK's newly drawn cards has a book

Book of 6 is found on the hand
You now have 2 cards.

KK's hand:
7♦ 7♥

KK's book(s) are 6

P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P

*** Type C/c to Continue ***: **c**

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

Computer has no card left. **<- this to draw more cards before the game continues**

Pile has 10 cards

Draw 6 cards from the pile.

Check whether Computer's newly drawn cards has a book

Book of 8 is found on the hand
Computer now has 2 cards.

Computer's hand:
7♠ 7♣

Computer's book(s) are 3 2 A 5 4 8

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

*** Type C/c to Continue ***: **c**

Computer turn

Computer (debugging purpose):
7♠ 7♣

Computer's book(s) are 3 2 A 5 4 8

KK's hand has 2 cards
7♦ 7♥

Do you have card(s) with 7?

Enter y to give the card(s) with 7 or n to say 'Go Fish': **n** **<- the user cheats by saying no**

><(((('> ><(((('> ><(((('>
~~~ ~~~ Go Fish ~~~ ~~~
><(((('> ><(((('> ><(((('>

*** Type C/c to Continue ***: **c**

KK, your turn

KK's hand has 2 cards
7♦ 7♥

KK's book(s) are 6

Which card (A, 2 - 9) do you want to ask for? **8**
~~~

><(((('> ><(((('> ><(((('> **<- the computer has no card 8, the computer says "go fish"**
~~~ ~~~ Go Fish ~~~ ~~~
><(((('> ><(((('> ><(((('>

KK, you take 9♥ from the center pile

KK's hand has 3 cards
9♥ 7♦ 7♥

*** Type C/c to Continue ***: **c**

Computer turn

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
7♠ 7♣

Computer's book(s) are 3 2 A 5 4 8

KK's hand has 3 cards
9♥ 7♦ 7♥

Do you have card(s) with 7?

Enter y to give the card(s) with 7 or n to say 'Go Fish': **y**

Remove card(s) with 7 from your hand

Book of 7 is found on the hand

*** Type C/c to Continue ***: **c**

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C

Computer has no card left.

Pile has 3 cards

Draw 3 cards from the pile.

Check whether Computer's newly drawn cards has a book
Computer now has 3 cards.

Computer's hand:
9♠ 9♣ 9♦

Computer's book(s) are 3 2 A 5 4 8 7

C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C-C
~~~

*** Type C/c to Continue ***: **c**
Computer turn

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
9♠ 9♣ 9♦

Computer's book(s) are 3 2 A 5 4 8 7

KK's hand has 1 cards
9♥

Do you have card(s) with 9?

Enter y to give the card(s) with 9 or n to say 'Go Fish': **y**

Remove card(s) with 9 from your hand

Book of 9 is found on the hand

KK's book(s) are 6

Computer's book(s) are 3 2 A 5 4 8 7 9

:(You lose :(

Sample Code Execution 3: Red entered by a user

Statements in purple in the example below are to explain how the game is played (NOT part of the output when your code is run)

Note: You are encouraged to make your output look more appealing than what is given below. However, your code should display very similar information in order for a user to be able to play the game.
It is VERY LIKELY that you will NOT get the exact same code execution shown below. This is used to show you how the game should be played.

Shuffle card **<- for Debugging purpose (no display in actual game)**
6♥ 7♠ 4♥ 5♣ 6♠ 9♥ 7♦ 5♥ 4♣ 3♥ 2♠ 5♦
2♦ 7♣ 2♣ 2♥ 9♦ 9♠ 3♦ A♣ 8♦ 3♣ 8♥ 8♣
A♥ A♠ 4♠ 8♠ 6♦ A♦ 3♠ 6♣ 4♦ 9♣ 5♠ 7♥

Enter your name: **Kay**

|@@@@ @@@@ @@@@ @@@@ @@@@ @@@@ @@@@|
Kay, Let's play Go Fish
|@@@@ @@@@ @@@@ @@@@ @@@@ @@@@ @@@@|

Kay's hand has 6 cards
2♠ 4♣ 7♦ 6♠ 4♥ 6♥

Computer's hand has 6 cards **<- for Debugging purpose (no display in actual game)**
5♦ 3♥ 5♥ 9♥ 5♣ 7♠

Deck after cards are distributed. **<- for Debugging purpose (no display in actual game)**
2♦ 7♣ 2♣ 2♥ 9♦ 9♠ 3♦ A♣ 8♦ 3♣ 8♥ 8♣
A♥ A♠ 4♠ 8♠ 6♦ A♦ 3♠ 6♣ 4♦ 9♣ 5♠ 7♥

*** Type C/c to Continue ***: **C**

Kay, your turn

Kay's hand has 6 cards
2♠ 4♣ 7♦ 6♠ 4♥ 6♥

Which card (A, 2 - 9) do you want to ask for? **6** **<- the computer has no 6 card, computer says go fish**

><(((('> ><(((('> ><(((('> **<- you can just display "Go Fish" in your code**
~~~ ~~~ Go Fish ~~~ ~~~
><(((('> ><(((('> ><(((('>

Kay, you take 2♦ from the center pile <- my code took the first card from the pile

Kay's hand has 7 cards
2♦ 2♠ 4♣ 7♦ 6♠ 4♥ 6♥ **<- 2♦ is added to player's hand**
~~~

*** Type C/c to Continue ***: **c**

Computer turn **<- the player loses a turn since he/she asks for a card that is not on the computer hand**

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
5♦ 3♥ 5♥ 9♥ 5♣ 7♠

Kay's hand has 7 cards
2♦ 2♠ 4♣ 7♦ 6♠ 4♥ 6♥

Do you have card(s) with 4? **<- Computer asks for a card (my code does not have "smart" computer)**

Enter y to give the card(s) with 4 or n to say 'Go Fish': **n** **<- the player can cheat by saying no**

><(((('> ><(((('> ><(((('>
~~~ ~~~ Go Fish ~~~ ~~~
><(((('> ><(((('> ><(((('>

*** Type C/c to Continue ***: **c**

Kay, your turn **<- the computer loses a turn, it is now back to the player's turn**

Kay's hand has 7 cards
2♦ 2♠ 4♣ 7♦ 6♠ 4♥ 6♥

Which card (A, 2 - 9) do you want to ask for? **7**

Taking card(s) from Computer's hand

Kay's hand has 8 cards **<- get 7♠ from the computer's hand**
7♠ 2♦ 2♠ 4♣ 7♦ 6♠ 4♥ 6♥

*** Type C/c to Continue ***: **c**

Kay, your turn

Kay's hand has 8 cards
7♠ 2♦ 2♠ 4♣ 7♦ 6♠ 4♥ 6♥

Which card (A, 2 - 9) do you want to ask for? **8**

><(((('> ><(((('> ><(((('>
~~~ ~~~ Go Fish ~~~ ~~~
><(((('> ><(((('> ><(((('>

Kay, you take A♣ from the center pile **<- my code shuffles the deck, it still take the first actual card form the deck**

Kay's hand has 9 cards
A♣ 7♠ 2♦ 2♠ 4♣ 7♦ 6♠ 4♥ 6♥

*** Type C/c to Continue ***: **c**

Computer turn

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
5♦ 3♥ 5♥ 9♥ 5♣

Kay's hand has 9 cards
A♣ 7♠ 2♦ 2♠ 4♣ 7♦ 6♠ 4♥ 6♥

Do you have card(s) with 4?

Enter y to give the card(s) with 4 or n to say 'Go Fish': **y**

Remove card(s) with 4 from your hand

*** Type C/c to Continue ***: **c**

Computer turn **<- still the computer turn since the computer got a card from the player's hand**

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
4♥ 4♣ 5♦ 3♥ 5♥ 9♥ 5♣ **<- 4♥ from the player's hand is added to the computer hand**

Kay's hand has 7 cards
A♣ 7♠ 2♦ 2♠ 7♦ 6♠ 6♥

Do you have card(s) with 2?

Enter y to give the card(s) with 2 or n to say 'Go Fish': **y**

Remove card(s) with 2 from your hand

*** Type C/c to Continue ***: **C**

Computer turn

Computer (debugging purpose): **<- for Debugging purpose (no display in actual game)**
2♠ 2♦ 4♥ 4♣ 5♦ 3♥ 5♥ 9♥ 5♣

Kay's hand has 5 cards
A♣ 7♠ 7♦ 6♠ 6♥

Do you have card(s) with 2?

Enter y to give the card(s) with 2 or n to say 'Go Fish': **y**

You do NOT have card(s) with 2. **<- the player said yes even though he/she has no card**
<- your code should check and prevent this.

><(((('> ><(((('> ><(((('>
~~~ ~~~ Go Fish ~~~ ~~~
><(((('> ><(((('> ><(((('>

Computer takes one card from the center pile

*** Type C/c to Continue ***: **c**

Kay, your turn

Kay's hand has 5 cards
A♣ 7♠ 7♦ 6♠ 6♥

Which card (A, 2 - 9) do you want to ask for? **7**

><(((('> ><(((('> ><(((('>
~~~ ~~~ Go Fish ~~~ ~~~
><(((('> ><(((('> ><(((('>

Kay, you take 7♣ from the center pile

The card has the same face that you asked for, you will get another turn.

Kay's hand has 6 cards
7♣ A♣ 7♠ 7♦ 6♠ 6♥

*** Type C/c to Continue ***: **C**

Kay, your turn

Kay's hand has 6 cards
7♣ A♣ 7♠ 7♦ 6♠ 6♥

Which card (A, 2 - 9) do you want to ask for?

... <- assume that the game continues and below is a scenario

P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P

Kay, You have no card left.

Pile has 1 cards **<- pile has only 1 card left,**

Draw 1 cards from the pile. **<- the hand can draw only 1 card**

Check whether Kay's newly drawn cards has a book
You now have 1 cards.

Kay's hand:
7♥

Kay's book(s) are 3 6 9 8 4 A

P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P-P

*** Type C/c to Continue ***: **c** **<- at this point, the pile has no card left**

Kay, your turn

Kay's hand has 1 cards
7♥

Kay's book(s) are 3 6 9 8 4 A

Which card (A, 2 - 9) do you want to ask for? 7

Taking card(s) from Computer's hand

Kay's hand has 4 cards
7♣ 7♠ 7♦ 7♥

Book of 7 is found on the hand

Kay's book(s) are 3 6 9 8 4 A 7

Computer's book(s) are 2 5

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@

Congratulations, YOU won.

@@@@@@@@@@@@@@@@@@@@@@@@@@@@@