---GAME---

everything to do with working with game logic

FILES

game.h

type definitions and function prototypes for the exposed interface for working with game logic

game.c

implementation of card and deck functions

TYPES

```
typedef struct {
    player *player;
    int numberOfPlayers;
    player *currentPlayer;
    card *deckOfCards;
}game;
```

keep track of the state of the game
player is a linked list of players
numberOfPlayers is the number of players
currentPlayer is a pointer to the current player
deckOfCards is the pool of cards in the middle of the table

+--+
| FUNCTIONS |
+--+

game.h

```
      void computerMove( game *g);

      void playerTurn(void);

      void deckOfCards(void);

      int checkIfWinner(game *g);
```

+--+
| FUNCTIONS -- IN DETAIL |
+--+

---game.h---

```
void computerMove( game *g, player *p);

      AI for go fish
      the computer makes an automated move

      Parameters:
            g -- a pointer to a game
            p -- a pointer to an automated player

      Returns:
            void

      Pre-conditions:
            g is not NULL

      Post-conditions:
            the computer makes a move, altering the state of the game


void playerTurn(game *g, player *p);

      allow the player to make a move

      Parameters:
```

```
            g -- a pointer to a game
            p -- a pointer to a human player

      Returns:
            void

      Pre-conditions:
            g is not NULL
            p is not NULL

      Post-conditions:
            the human player makes a move, altering the state of the game


card *deckOfCards(game *g);

      take a card from the pool of cards on the table

      Parameters:
            g -- a pointer to a game

      Returns:
            a pointer to the card that was taken from the table

      Pre-conditions:
            g is not NULL
            the table still has cards on it

      Post-conditions:
            a card is removed from the table


void checkIfWinner(game *g, char resultContainer[]);

      check if someone has won the game

      Parameters:
            g -- a pointer to a game
            resultContainer -- a string to hold the winners, if any

      Returns:
            void

      Pre-conditions:
            g is not NULL
            resultContainer has enough elements to hold every player, just in case
                  everyone is tied

      Post-conditions:
            the result of the game is written into resultContainer
            if player 1 wins, resultContainer will be "1"
            if player 1 and 2 are tied, resultContainer will be "12"
```

```
if there are no winners yet, resultContainer will be ""
if everyone is tied, resultContainer will be "12345..."(assuming)
      there will never be more than 7 players
```