

Computer Project #10

Edit 11/20/19: updated Test 4 output

Assignment Overview

This assignment focuses on the design, implementation and testing of a Python program which uses an instructor-supplied module to play a card game, as described below.

It is worth 55 points (5.5% of course grade) and must be completed no later than 11:59 PM on Monday, **November 25, 2019**.

Assignment Deliverables

The deliverable for this assignment is the following file:

`proj10.py` – the source code for your Python program

Be sure to use the specified file name and to submit it for grading via the **Mimir system** before the project deadline.

Assignment Background

Aces Up is a popular solitaire card game which is played by one person with a standard 52-card deck of cards. The rules and a tutorial video are available at:

<http://worldofsolitaire.com/>

Click on “Choose Game” at the top of the screen and click on “Aces Up”.

Your program will allow the user to play a simplified version of Aces Up, with the program managing the game. The game rules are given below.

Game Rules

1. Start: The game is played with one standard deck of 52 cards. The deck is shuffled and becomes the initial *stock* (pile of cards). The cards in the stock are placed face down.

Four cards are then dealt from the stock, left to right, one to each column of a four-column *tableau*. During play, additional cards will be added to the tableau, but it starts with just one card in each column. All tableau cards are visible.

The game also has a *foundation*, which is a single pile of cards. The foundation starts out empty and is filled during play.

2. Goal: The game is won when the stock is empty and the only cards left in the tableau are the four aces.

3. Moves: A player can move only one card at a time and the moved card must be the bottom card of a tableau column.

The card at the bottom of a column can be moved to the foundation if a higher rank card of the same suit is at the bottom of another column. Note that aces are high in this game—that is, the rank of an ace is greater than the rank of any other card.

If a column of the tableau becomes empty, any card at the bottom of another column can be moved to the empty column.

No other moves are permitted.

4. Deal: A player can choose to deal instead of moving a card. In this case, the top four cards are dealt from the stock, left to right, with one card placed at the bottom of each column of the tableau. (Because four cards are always dealt at the same time the stock will never have fewer than four cards in this game.)

Assignment Specifications

You will develop a program that allows the user to play Aces Up according to the rules given above. The program will use the instructor-supplied `cards.py` module to model the cards and deck of cards. To help clarify the specifications, we provide a sample interaction with a program satisfying the specifications at the end of this document.

1. The program will recognize the following commands (upper or lower case):

D	Deal four cards from the stock to the tableau
F x	Move card from Tableau column x to the Foundation.
T x y	Move card from Tableau column x to empty Tableau column y .
R	Restart the game (after shuffling)
H	Display the menu of choices
Q	Quit

where **x** and **y** denote column numbers, and the columns are numbered from 1 to 4.

The program will repeatedly display the current state of the game and prompt the user to enter a command until the user wins the game or enters “**Q**” (or “**q**”), whichever comes first.

The program will detect, report and recover from invalid commands. None of the data structures representing the stock, tableau, or foundation will be altered by an invalid command.

2. The program will use the following function to initialize a game:

```
init_game() → (stock, tableau, foundation)
```

That function has no parameters. It creates and initializes the `stock`, `tableau`, and `foundation`, and then returns them as a tuple, in that order. This corresponds to rule 1 in the game rules:

- `stock` is a deck class object,
- `foundation` is an empty list, and
- `tableau` is a list of lists, each containing one of the dealt cards. The first element of `tableau` is the leftmost column when displayed

The deck is shuffled and becomes the initial `stock` (pile of cards). Four cards are then dealt from the `stock`, left to right, one to each column of a four-column `tableau`.

3. The program will use the following function to deal cards to the tableau:

```
deal_to_tableau( stock, tableau ): → None
```

That function has two parameters: the data structure representing the `stock` and the data structure representing the `tableau`. It will deal a card from the `stock` to each column of the `tableau`. It will always deal four cards (why?).

4. The program will use the following function to display the current state of the game:

```
display( stock, tableau, foundation ) → None
```

Provided.

5. The program will use the following function to prompt the user to enter an option and return a representation of the option designed to facilitate subsequent processing.

```
get_option() → list
```

That function takes no parameters. It prompts the user for an option and checks that the input supplied by the user is of the form requested in the menu. Valid inputs for options are described in item (bullet) 1 of the specifications. If the input is not of the required form, the function prints an error message and returns `None`.

Note that input with the incorrect number of arguments, e.g. `D 2`, or incorrect types, e.g. `F D`, will return `None`.

Also, note that in this function you do not check that 'y' of 'T x y' is empty (you do not have `tableau` as a parameter to check that).

The function returns a list as follows:

- **None**, if the input is not of the required form
- **['D']**, for deal
- **['F', x]**, where **x** is an **int**, for moving a card to the foundation
- **['T', x, y]**, where **x** and **y** are **int**'s, for moving a card within the tableau from column **x** to column **y**
- **['R']**, for restart
- **['H']**, for displaying the menu
- **['Q']**, for quit

6. The program will use the following function to determine if a requested move to the foundation is valid:

```
validate_move_to_foundation( tableau, from_col ) → bool
```

That function has two parameters: the data structure representing the tableau and an **int** indicating the column whose bottom card should be moved. The function will return **True**, if the move is valid; and **False**, otherwise. In the latter case, it will also print an appropriate error message. There are two errors that can be caught: `from_col` is empty or move is invalid. Remember: a card can be moved to the foundation only if a higher ranked card of the same suit is at the bottom of a Tableau column.

Hint: consider tableau index and remember that Ace is high. Also, `from_col` must be a valid value ($1 \leq \text{from_col} \leq 4$).

7. The program will use the following function to move a card from the tableau to the foundation:

```
move_to_foundation( tableau, foundation, from_col ) → None
```

That function has three parameters: the data structure representing the tableau, the data structure representing the foundation, and an **int** indicating the column whose bottom card should be moved. If the move is valid, the function will update the tableau and foundation; otherwise, it will do nothing to them. Hint: the list `pop()` method is useful, if you desire a terse function. `from_col` must be a valid value ($1 \leq \text{from_col} \leq 4$).

8. The program will use the following function to determine if a requested move to within the tableau is valid:

```
validate_move_within_tableau( tableau, from_col, to_col ) → bool
```

That function has three parameters: the data structure representing the tableau, an **int** indicating the column whose bottom card should be moved, and an **int** indicating the column the card should be moved to. The function will return **True**, if the move is valid; and **False**, otherwise.

In the latter case, it will also print an appropriate error message. Remember: you can only move a card to an empty tableau slot. Also, `from_col` and `to_col` must be valid column values ($1 \leq \text{from_col} \leq 4$ and $1 \leq \text{to_col} \leq 4$).

9. The program will use the following function to move a card from the tableau to the foundation:

```
move_within_tableau( tableau, from_col, to_col ) → None
```

That function has three parameters: the data structure representing the tableau, an `int` indicating the column whose bottom card should be moved, and an `int` indicating the column the card should be moved to. If the move is valid, the function will update the tableau; otherwise, it will do nothing to it. Hint: the list `pop()` method is useful, if you desire a terse function. `from_col` and `to_col` must be valid column values: ($1 \leq \text{from_col} \leq 4$ and $1 \leq \text{to_col} \leq 4$).

10. The program will use the following function to check if the game has been won:

```
check_for_win( stock, tableau ) → bool
```

That function has two parameters: the data structure representing the stock and the data structure representing the tableau. It returns `True`, if the stock is empty and the tableau contains only the four aces; and `False`, otherwise.

11. Once you write all your function, it is time to write your main function:

- a) Your program should start by initializing the board (the stock, the tableau and the foundation).
- b) Display the menu and the board.
- c) Ask to input an option and check the validity of the input.
- d) If `'D'`, deal four cards from the stock to the tableau
- e) If `'F x'`, move card from Tableau column `x` to the Foundation
- f) If `'T x y'`, move card from Tableau column `x` to empty Tableau column `y`.
- g) If `'R'`, restart the game by initializing the board (after shuffling). Display the rules and the menu.
- h) If `'H'`, display the menu of choices
- i) If `'Q'`, quit the game
- j) If none of these options, the program should display an error message.
- k) The program should repeat until the user won or quit the game. Display a goodbye message.

Assignment Notes

1. Before you begin to write any code, play with the provided demo program and look over the sample interaction supplied on the project website to be sure you understand the rules of the game and how you will simulate the demo program. The demo program is at <http://worldofsolitaire.com/>: Click on “Choose Game” at the top of the screen and click on “Aces Up”.

2. We provide a module called `cards.py` that contains a `Card` class and a `Deck` class. Your program must use this module (`import cards`). *Do not modify this file!* This is a generic module for any card game and happens to be implemented with “aces low” (having a rank of 1). Your game requires aces to have a rank higher than any other card. Your program must implement “aces high” *without modifying the `cards` module*.

3. Laboratory Exercise #11 demonstrates how to use the `cards` module. Understanding those programs should give you a good idea how you can use the module in your game.

4. We have provided a framework named `proj10.py` to get you started. *Using this framework is mandatory.* Begin by downloading `proj10.py`. Check that it compiles. Gradually replace the “stub” code (marked with comments) with your own code. (Delete the stub code.)

5. The coding standard for CSE 231 is posted on the course website:

<http://www.cse.msu.edu/~cse231/General/coding.standard.html>

Items 1-9 of the Coding Standard will be enforced for this project.

6. Your program may not use any global variables inside of functions. That is, all variables used in a function body must belong to the function’s local name space. The only global references will be to functions and constants.

7. Your program may not use any nested functions. That is, you may not nest the definition of a function inside another function definition.

8. Your program must contain the functions listed above; you may develop additional functions, as appropriate.

TEST 1

Input options:

D: Deal to the Tableau (one card on each column).
F x: Move card from Tableau column x to the Foundation.
T x y: Move card from Tableau column x to empty Tableau column y.
R: Restart the game (after shuffling)
H: Display the menu of choices
Q: Quit the game

```
stock      tableau      foundation
XX        A♠ K♠ Q♠ A♥
```

Input an option (DFTRHQ): H

Input options:

D: Deal to the Tableau (one card on each column).
F x: Move card from Tableau column x to the Foundation.
T x y: Move card from Tableau column x to empty Tableau column y.
R: Restart the game (after shuffling)
H: Display the menu of choices
Q: Quit the game

```
stock      tableau      foundation
XX        A♠ K♠ Q♠ A♥
```

Input an option (DFTRHQ): D

```
stock      tableau      foundation
XX        A♠ K♠ Q♠ A♥
           K♥ Q♥ A♦ K♦
```

Input an option (DFTRHQ): R

===== Restarting: new game =====

Aces High Card Game:

Tableau columns are numbered 1,2,3,4.
Only the card at the bottom of a Tableau column can be moved.
A card can be moved to the Foundation only if a higher ranked card
of the same suit is at the bottom of another Tableau column.
To win, all cards except aces must be in the Foundation.

Input options:

D: Deal to the Tableau (one card on each column).
F x: Move card from Tableau column x to the Foundation.
T x y: Move card from Tableau column x to empty Tableau column y.
R: Restart the game (after shuffling)
H: Display the menu of choices
Q: Quit the game

```
stock      tableau      foundation
XX        A♠ K♠ Q♠ A♥
```

Input an option (DFTRHQ): Q

You have chosen to quit.

TEST 2 : stock is only queens, kings, and aces

Input options:

D: Deal to the Tableau (one card on each column).

F x: Move card from Tableau column x to the Foundation.

T x y: Move card from Tableau column x to empty Tableau column y.

R: Restart the game (after shuffling)

H: Display the menu of choices

Q: Quit the game

```
stock      tableau      foundation
XX        A♠ K♠ Q♠ A♥
```

Input an option (DFTRHQ): F 1

Error, cannot move A♠.

```
stock      tableau      foundation
XX        A♠ K♠ Q♠ A♥
```

Input an option (DFTRHQ): F 2

```
stock      tableau      foundation
XX        A♠      Q♠ A♥      K♠
```

Input an option (DFTRHQ): F 3

```
stock      tableau      foundation
XX        A♠      A♥      Q♠
```

Input an option (DFTRHQ): D

```
stock      tableau      foundation
XX        A♠ Q♥ A♦ A♥      Q♠
          K♥      K♦
```

Input an option (DFTRHQ): f 4

```
stock      tableau      foundation
XX        A♠ Q♥ A♦ A♥      K♦
          K♥
```

Input an option (DFTRHQ): F 2

```
stock      tableau      foundation
XX        A♠      A♦ A♥      Q♥
          K♥
```

Input an option (DFTRHQ): T 1 2

```
stock      tableau      foundation
XX        A♠ K♥ A♦ A♥      Q♥
```

Input an option (DFTRHQ): F 2

stock	tableau	foundation
XX	A♠ A♦ A♥	K♥

Input an option (DFTRHQ): D

stock	tableau	foundation
	A♠ A♣ A♦ A♥	K♥
	Q♦ K♣ Q♣	

Input an option (DFTRHQ): F 3

stock	tableau	foundation
	A♠ A♣ A♦ A♥	K♣
	Q♦ Q♣	

Input an option (DFTRHQ): F 4

stock	tableau	foundation
	A♠ A♣ A♦ A♥	Q♣
	Q♦	

Input an option (DFTRHQ): F 1
You won!

TEST 3: stock is only queens, kings, and aces

Input options:

- D: Deal to the Tableau (one card on each column).
- F x: Move card from Tableau column x to the Foundation.
- T x y: Move card from Tableau column x to empty Tableau column y.
- R: Restart the game (after shuffling)
- H: Display the menu of choices
- Q: Quit the game

stock tableau foundation

XX Q♠ Q♥ Q♦ Q♣

Input an option (DFTRHQ): D

stock tableau foundation

XX Q♠ Q♥ Q♦ Q♣
 A♠ K♠ A♥ K♥

Input an option (DFTRHQ): F a

Error in option: F a

Invalid option.

stock tableau foundation

XX Q♠ Q♥ Q♦ Q♣
 A♠ K♠ A♥ K♥

Input an option (DFTRHQ): T 2

Error in option: T 2

Invalid option.

stock tableau foundation

XX Q♠ Q♥ Q♦ Q♣
 A♠ K♠ A♥ K♥

Input an option (DFTRHQ): D 2

Error in option: D 2

Invalid option.

stock tableau foundation

XX Q♠ Q♥ Q♦ Q♣
 A♠ K♠ A♥ K♥

Input an option (DFTRHQ): F 2

stock	tableau	foundation
XX	Q♠ Q♥ Q♦ Q♣	K♠
	A♠ A♥ K♥	

Input an option (DFTRHQ): F 4

stock	tableau	foundation
XX	Q♠ Q♥ Q♦ Q♣	K♥
	A♠ A♥	

Input an option (DFTRHQ): F 2

stock	tableau	foundation
XX	Q♠ Q♦ Q♣	Q♥
	A♠ A♥	

Input an option (DFTRHQ): T 1 2

stock	tableau	foundation
XX	Q♠ A♠ Q♦ Q♣	Q♥
	A♥	

Input an option (DFTRHQ): F 1

stock	tableau	foundation
XX	A♠ Q♦ Q♣	Q♠
	A♥	

Input an option (DFTRHQ): T 3 1

stock	tableau	foundation
XX	A♥ A♠ Q♦ Q♣	Q♠

Input an option (DFTRHQ): d

stock	tableau	foundation
	A♥ A♠ Q♦ Q♣	Q♠
	A♦ K♦ A♣ K♣	

Input an option (DFTRHQ): F 4

stock	tableau	foundation
	A♥ A♠ Q♦ Q♣	K♣
	A♦ K♦ A♣	

Input an option (DFTRHQ): F 4

stock	tableau	foundation
A♥ A♠ Q♦		Q♣
A♦ K♦ A♣		

Input an option (DFTRHQ): T 3 4

stock	tableau	foundation
A♥ A♠ Q♦ A♣		Q♣
A♦ K♦		

Input an option (DFTRHQ): 3

Error in option: 3

Invalid option.

stock	tableau	foundation
A♥ A♠ Q♦ A♣		Q♣
A♦ K♦		

Input an option (DFTRHQ): F 3

stock	tableau	foundation
A♥ A♠ A♣		Q♦
A♦ K♦		

Input an option (DFTRHQ): T 1 3

stock	tableau	foundation
A♥ A♠ A♦ A♣		Q♦
K♦		

Input an option (DFTRHQ): F 3

Error, cannot move A♦.

stock	tableau	foundation
A♥ A♠ A♦ A♣		Q♦
K♦		

Input an option (DFTRHQ): F 2

You won!

TEST 4

Input options:

- D: Deal to the Tableau (one card on each column).
- F x: Move card from Tableau column x to the Foundation.
- T x y: Move card from Tableau column x to empty Tableau column y.
- R: Restart the game (after shuffling)
- H: Display the menu of choices
- Q: Quit the game

```
stock      tableau      foundation
XX        K♠  Q♠  J♠  10♠
```

Input an option (DFTRHQ): f 4

```
stock      tableau      foundation
XX        K♠  Q♠  J♠                10♠
```

Input an option (DFTRHQ): T 1 4

```
stock      tableau      foundation
XX                Q♠  J♠  K♠        10♠
```

Input an option (DFTRHQ): d

```
stock      tableau      foundation
XX        9♠  Q♠  J♠  K♠        10♠
           8♠  7♠  6♠
```

Input an option (DFTRHQ): f 4

```
stock      tableau      foundation
XX        9♠  Q♠  J♠  K♠        6♠
           8♠  7♠
```

Input an option (DFTRHQ): f 4
Error, cannot move K♠.

```
stock      tableau      foundation
XX        9♠  Q♠  J♠  K♠        6♠
           8♠  7♠
```

Input an option (DFTRHQ): f 2

```
stock      tableau      foundation
XX        9♠  Q♠  J♠  K♠        8♠
           7♠
```

Input an option (DFTRHQ): f 3

stock tableau foundation
XX 9♠ Q♠ J♠ K♠ 7♠

Input an option (DFTRHQ): d

stock tableau foundation
XX 9♠ Q♠ J♠ K♠ 7♠
 5♠ 4♠ 3♠ 2♠

Input an option (DFTRHQ): f 3

stock tableau foundation
XX 9♠ Q♠ J♠ K♠ 3♠
 5♠ 4♠ 2♠

Input an option (DFTRHQ): f 2

stock tableau foundation
XX 9♠ Q♠ J♠ K♠ 4♠
 5♠ 2♠

Input an option (DFTRHQ): f 1

stock tableau foundation
XX 9♠ Q♠ J♠ K♠ 5♠
 2♠

Input an option (DFTRHQ): f 1

stock tableau foundation
XX Q♠ J♠ K♠ 9♠
 2♠

Input an option (DFTRHQ): t 4 1

stock tableau foundation
XX 2♠ Q♠ J♠ K♠ 9♠

Input an option (DFTRHQ): d

stock tableau foundation
XX 2♠ Q♠ J♠ K♠ 9♠
 A♠ K♥ Q♥ J♥

Input an option (DFTRHQ): f 3

stock tableau foundation
XX 2♠ Q♠ J♠ K♠ Q♥
 A♠ K♥ J♥

Input an option (DFTRHQ): f 4

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	J♥
	A♠ K♥	

Input an option (DFTRHQ): f 2

Error, cannot move K♥.

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	J♥
	A♠ K♥	

Input an option (DFTRHQ): f 1

Error, cannot move A♠.

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	J♥
	A♠ K♥	

Input an option (DFTRHQ): d

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	J♥
	A♠ K♥ 8♥ 7♥	
	10♥ 9♥	

Input an option (DFTRHQ): d

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	J♥
	A♠ K♥ 8♥ 7♥	
	10♥ 9♥ 4♥ 3♥	
	6♥ 5♥	

Input an option (DFTRHQ): f 1

Error, cannot move 6♥.

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	J♥
	A♠ K♥ 8♥ 7♥	
	10♥ 9♥ 4♥ 3♥	
	6♥ 5♥	

Input an option (DFTRHQ): f 1

Error, cannot move 6♥.

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	J♥
	A♠ K♥ 8♥ 7♥	
	10♥ 9♥ 4♥ 3♥	
	6♥ 5♥	

Input an option (DFTRHQ): f 2

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	5♥
	A♠	K♥	8♥	7♥	
	10♥	9♥	4♥	3♥	
	6♥				

Input an option (DFTRHQ): f 3

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	4♥
	A♠	K♥	8♥	7♥	
	10♥	9♥		3♥	
	6♥				

Input an option (DFTRHQ): f 3

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	8♥
	A♠	K♥		7♥	
	10♥	9♥		3♥	
	6♥				

Input an option (DFTRHQ): d

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	8♥
	A♠	K♥	K♦	7♥	
	10♥	9♥		3♥	
	6♥	A♥		Q♦	
	2♥				

Input an option (DFTRHQ): f 2

Error, cannot move A♥.

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	8♥
	A♠	K♥	K♦	7♥	
	10♥	9♥		3♥	
	6♥	A♥		Q♦	
	2♥				

Input an option (DFTRHQ): f 1

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	2♥
	A♠	K♥	K♦	7♥	
	10♥	9♥		3♥	
	6♥	A♥		Q♦	

Input an option (DFTRHQ): f 4

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	Q♦
	A♠	K♥	K♦	7♥	
	10♥	9♥		3♥	
	6♥	A♥			

Input an option (DFTRHQ): f 4

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	3♥
	A♠	K♥	K♦	7♥	
	10♥	9♥			
	6♥	A♥			

Input an option (DFTRHQ): f 4

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	7♥
	A♠	K♥	K♦		
	10♥	9♥			
	6♥	A♥			

Input an option (DFTRHQ): f 4
 Error, cannot move K♠.

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	7♥
	A♠	K♥	K♦		
	10♥	9♥			
	6♥	A♥			

Input an option (DFTRHQ): t 2 4
 Invalid move

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	7♥
	A♠	K♥	K♦		
	10♥	9♥			
	6♥	A♥			

Input an option (DFTRHQ): f 2
 Error, cannot move A♥.

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	7♥
	A♠	K♥	K♦		
	10♥	9♥			
	6♥	A♥			

Input an option (DFTRHQ): t 3 2
Invalid move

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	7♥
	A♠ K♥ K♦	
	10♥ 9♥	
	6♥ A♥	

Input an option (DFTRHQ): d

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	7♥
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦	
	6♥ A♥	
	J♦ 10♦	

Input an option (DFTRHQ): d

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	7♥
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦ 4♦	
	6♥ A♥ 5♦	
	J♦ 10♦	
	7♦ 6♦	

Input an option (DFTRHQ): f 2

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	6♦
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦ 4♦	
	6♥ A♥ 5♦	
	J♦ 10♦	
	7♦	

Input an option (DFTRHQ): f 4

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	4♦
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦	
	6♥ A♥ 5♦	
	J♦ 10♦	
	7♦	

Input an option (DFTRHQ): d

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	4♦

```

A♠ K♥ K♦ 8♦
10♥ 9♥ 9♦ K♣
6♥ A♥ 5♦
J♦ 10♦ A♦
7♦ 2♦
3♦

```

Input an option (DFTRHQ): f 4
 Error, cannot move K♣.

```

stock      tableau      foundation
XX         2♠ Q♠ J♠ K♠      4♦
           A♠ K♥ K♦ 8♦
           10♥ 9♥ 9♦ K♣
           6♥ A♥ 5♦
           J♦ 10♦ A♦
           7♦ 2♦
           3♦

```

Input an option (DFTRHQ): f 3
 Error, cannot move A♦.

```

stock      tableau      foundation
XX         2♠ Q♠ J♠ K♠      4♦
           A♠ K♥ K♦ 8♦
           10♥ 9♥ 9♦ K♣
           6♥ A♥ 5♦
           J♦ 10♦ A♦
           7♦ 2♦
           3♦

```

Input an option (DFTRHQ): f 3
 Error, cannot move A♦.

```

stock      tableau      foundation
XX         2♠ Q♠ J♠ K♠      4♦
           A♠ K♥ K♦ 8♦
           10♥ 9♥ 9♦ K♣
           6♥ A♥ 5♦
           J♦ 10♦ A♦
           7♦ 2♦
           3♦

```

Input an option (DFTRHQ): f 1

```

stock      tableau      foundation
XX         2♠ Q♠ J♠ K♠      3♦
           A♠ K♥ K♦ 8♦
           10♥ 9♥ 9♦ K♣
           6♥ A♥ 5♦
           J♦ 10♦ A♦
           7♦ 2♦

```

Input an option (DFTRHQ): f 3
Error, cannot move A♦.

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	3♦
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
	7♦	2♦			

Input an option (DFTRHQ): f 3
Error, cannot move A♦.

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	3♦
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
	7♦	2♦			

Input an option (DFTRHQ): f 3
Error, cannot move A♦.

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	3♦
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
	7♦	2♦			

Input an option (DFTRHQ): t 4 3
Invalid move

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	3♦
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
	7♦	2♦			

Input an option (DFTRHQ): f 3
Error, cannot move A♦.

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	3♦
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	

6♥ A♥ 5♦
 J♦ 10♦ A♦
 7♦ 2♦

Input an option (DFTRHQ): t 1 3
 Invalid move

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	3♦
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦ K♣	
	6♥ A♥ 5♦	
	J♦ 10♦ A♦	
	7♦ 2♦	

Input an option (DFTRHQ): f 1

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	7♦
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦ K♣	
	6♥ A♥ 5♦	
	J♦ 10♦ A♦	
	2♦	

Input an option (DFTRHQ): f 2

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	2♦
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦ K♣	
	6♥ A♥ 5♦	
	J♦ 10♦ A♦	

Input an option (DFTRHQ): d

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	2♦
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦ K♣	
	6♥ A♥ 5♦ 9♣	
	J♦ 10♦ A♦	
	Q♣ J♣ 10♣	

Input an option (DFTRHQ): f 2

stock	tableau	foundation
XX	2♠ Q♠ J♠ K♠	J♣
	A♠ K♥ K♦ 8♦	
	10♥ 9♥ 9♦ K♣	
	6♥ A♥ 5♦ 9♣	
	J♦ 10♦ A♦	

Q♣ 10♣

Input an option (DFTRHQ): f 2
Error, cannot move 10♦.

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	J♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦	9♣	
	J♦	10♦	A♦		
	Q♣		10♣		

Input an option (DFTRHQ): f 4

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	9♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
	Q♣		10♣		

Input an option (DFTRHQ): f 3

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	10♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
	Q♣				

Input an option (DFTRHQ): f 1

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	Q♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		

Input an option (DFTRHQ): d

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	Q♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦	5♣	
	J♦	10♦	A♦		
	8♣	7♣	6♣		

Input an option (DFTRHQ): f 4

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	5♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
	8♣	7♣	6♣		

Input an option (DFTRHQ): f 1

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	8♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
		7♣	6♣		

Input an option (DFTRHQ): f 2

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	7♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		
			6♣		

Input an option (DFTRHQ): f 3

stock	tableau				foundation
XX	2♠	Q♠	J♠	K♠	6♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦		
	J♦	10♦	A♦		

Input an option (DFTRHQ): d

stock	tableau				foundation
	2♠	Q♠	J♠	K♠	6♣
	A♠	K♥	K♦	8♦	
	10♥	9♥	9♦	K♣	
	6♥	A♥	5♦	A♣	
	J♦	10♦	A♦		
	4♣	3♣	2♣		

Input an option (DFTRHQ): q
You have chosen to quit.

Grading Rubric

Computer Project #10
Scoring Summary

General Requirements:

(5 pts) Coding Standard 1-9
(descriptive comments, mnemonic identifiers, format, etc...)

Implementations:

(3 pts) `init_game`
(6 pts) `validate_move_to_foundation`
(4 pts) `move_to_foundation`
(6 pts) `validate_move_within_tableau`
(4 pts) `move_within_tableau`
(4 pts) `check_for_win`
(2 pts) Test 1
(7 pts) Test 2
(7 pts) Test 3
(7 pts) Test 4