Programming Project 3 FAQ

Q: Can you clarify the circularity and what it has to do with foundation piles?
A: Foundation piles ALWAYS build up and in the same suit. i.e., a 4 comes on top of a 3, etc. What comes on top of K? The answer depends whether the pile is circular or not. If the pile is not circular, nothing can be played on K. If the pile is circular, 1 can be played on K.

Q: What is that base card in foundation piles? Does it have something to do with circularity of piles?
A: If the foundation pile is non-circular, the base card MUST be 1. Otherwise, you wouldn't be able to place all cards on the foundation pile. Thus, static base card should be 1 in case of non-circular piles and it would be some other card (whichever is first played onto any foundation pile) in case the foundation piles are circular.

Q: Why the baseCard in foundation piles is static?
A: You are free to use other implementations. However, I used static baseCard because in each game there are four foundation piles and each of them MUST start with the same baseCard. Of course, you wouldn't want the card itself to be static; it is only the rank of the card which is static. For example, if one foundation pile starts with 6H, all other foundation piles must start with a 6 (of different suits, of course!).

Q: What is getNextCard() supposed to return? Just the top most card in the Deck?
A: Top most card in the deck! You may realize that returning a pointer will be beneficial because you don't want to create copies of cards *ever*.

Q: Can I use random_shuffle() from STL algorithm? It gives the same sequence of cards.
A: Use of random_shuffle() is encouraged compared to writing your own shuffle algorithm. The reason you always get same shuffled sequence from random_shuffle() is because the seed to the random number generator is the same. If that does not make sense to you, simply call srand(time(0)) before calling random_shuffle() and you will be fine. Make sure you #include <ctime> in your implementation. For Unix implementations, even this may not work in which case you will need to use srand48(time(NULL)).

Q: Should we have bools for everything in the Pile class: circular, fanned, canReceiveTopCard(), canRemoveTopCard()?
A: Of course, you can have bools to deal with it. However, another, more interesting, way of dealing with it is to make effective use of inheritance. e.g., consider classes D1 and D2 derive from class B. In class B, you can choose not to use bools and then provide Circular behavior in class D1 built-in, etc. Similarly, canRemoveTopCard(dstPile) can be a method that different Piles implement their own way. More interestingly, canAddTopCard(srcPile) is a message that is sent to the destination pile which can check whether the destination is circular or not and it is able to accept the top card from the source pile. Fanned behavior can also be the made the property of individual classes in how they print. Note that fanned is only important when you are printing the file.
Therefore, a fanned pile will print normally while an un-fanned pile will override the print behavior by printing only the top card.

Q: How are the moves controlled on the console? What are those src# dst#  
A: We assign pile numbers to each pile and those numbers are simply the pile numbers. Please run the program available on the class web pages.

Q: I don’t understand why you keep talking about Pile in Foundation Pile description in the project handout?  
A: Here we are using the classic inheritance terminology. Remember that FoundationPile *IS A* Pile. Under the description of FoundationPile, when we refer to Pile, we really mean this specific type of Pile which just happens to be the FoundationPile.

Q: StockPile doesn’t receive any cards, so where do we initialize its contents? If this is done in the constructor then to what should we initialize them to?  
A: When you are dealing cards (remember each game has its own dealcards() method), distribute cards in the piles. Also note that during the dealing, you may not stick to the "build" rules for piles (e.g., alternating in color, building down, etc.)

Q: What should really go in the Pile constructor?  
A: All I did in the constructor of base class Pile is to initialize the topCard to NULL and set the maximum number of cards this pile may contain. Of course, you might want to do something in addition to this in derived classes.

Q: When we initially distribute the cards in the tableau piles, are they distributed randomly or does the rank and color order also matter initially?  
A: In the initial deal, the cards are distributed in the piles randomly (without following the rules of the piles), as in a regular deal of Solitaire.

Q: In the initial deal of Sixteens, will SpecialTableauPiles contain 2 cards each?  
A: SpecialTableauPiles in Sixteens each will start with 2 cards each, yes! But later on during the game, they can carry up to a maximum of 3 cards.