```
# Sean MacBride
# Program: smacbrideP6.py
# Description: Simulates the card game of 'Mickslam!' using a couple of different classes.
# Input: If you would like to continue with the game, an input of c (computer) or p (player)
will determine who you believe to win. Will need repeated inputs until either you run out
of cards, or you enter q into the prompt
# Output: The hand dealt for the player and points associated with that hand, the hand
dealt for the computer (if necessary) and points associated with that hand, and repetition
of the hands dealt until you run out of cards or input q to end the game. After the game
concludes, will display the win values of the computer and player.

import random #Importing Random


class Game: #Creating the class Game, which acts as the game manager for your hand and
the computer's hand
   def __init__(self): #Constructor
      self.wincount=0 #Setting initial wincount to 0
      self.points=0 #Setting initial points to 0


   def reset(self): #A reset method used to reset the point values after a round of the game
      self.points=0

   def Calc(self, face): #A calc method to calculate the point values of a card based on the
face of the card
      if face=="King" or face=="Queen" or face=="Jack":
         return 10
      elif face=="Ace":
         return 11
      else:
         return face

   def pointShow(self): #A method to display the amount of points a hand has
      print("\nPoints:",self.points)

   def Won(self): #A method to update the wincount if the computer or player won the
game
      self.wincount+=1


class Deck: #The deck class, which keeps track of all of the cards played
   def __init__(self): #Constructor
      self.deck=list(range(52)) #Using the method used in cards.py, I set all the entries in a
list of 52 items to False
      for card in self.deck:
         self.deck[card]=False
      self.cardnum=0 #The amount of cards that have been played. Is used later as a condition on a for loop

#dealCard uses the ideas put forth in cards.py of a 52 elements representing 52 different cards in a french deck
   def dealCard(self): #A method to deal a card
      card = random.randint(0,51) #Determines a random number to put in the list
      while self.deck[card] == True: #Checks to see if the number was already in the list
         card = random.randint(0, 51) #Re runs the random number integer to see if it was in the list
      self.deck[card]=True #Puts the number in the list, effectively making the card in that place "dealt"
      face,suit=self.cardconvert(card) #Calling cardconvert(card) which returns the face and suit of the card
      self.showCard(face,suit) #calls the showCard function that prints the card
      self.cardnum+=1 #Adds one to the cardnum count
      return face #Returns face

   def cardconvert(self,card): #The cardconvert method which returns the face and suit of the card given the card integer value
      if card < 13:
         value = card+1
         suit = "Hearts"
      elif card < 26:
         value = card-12
         suit = "Diamonds"
      elif card < 39:
         value = card-25
         suit = "Clubs"
      else:
         value = card-38
         suit = "Spades"
      face=self.stringconvert(value)
      return face,suit #returns face and suit

   def stringconvert(self,value): #The stringconvert method that returns the face of the card (ex. 11 corresponds to jack)
      if value==1:
         face="Ace"
      elif value==11:
         face="Jack"
      elif value==12:
         face="Queen"
      elif value==13:
         face="King"
        else:
            face=value
        return face    #returns face

    def showCard(self,face,suit): #The showCard method that prints out the correct format of the card
        print(face,"of",suit) #Printing the card

def main(): #The main function, which runs the game
    cards=Deck() #Creating the deck of cards
    player=Game() #Creating the player's game
    computer=Game() #Creating the computer's game
    prompt=0 #placeholder for prompt, get's updated later
    while cards.cardnum < 50 and prompt != "q": #While loop to test if all the possible cards have been played (if 50 cards can be played, you can't play another full round of the game) and if the prompt asks for you to quit
        print("\nPlayer cards:\n") #Prints a nice header
        for i in range(5): #for loop to deal 5 cards in the player's first hand
            card=cards.dealCard() #dealing a card to player
            player.points+=player.Calc(card) #adding the point value of the card to the player's points
        player.pointShow() #displaying the points
        prompt=input("\nWho do you believe will win? Input p for player, c for computer, and q to quit. ") #Asking who you think will win
        if prompt !="q": #As long as you don't want to quit, will run this loop
            print("\nComputer cards:\n") #Prints a nice header
            for i in range(5): #a for loop to deal the computers hand
                card=cards.dealCard() #dealing a hand to the computer
                computer.points+=computer.Calc(card) #adding to the computer's point total
            computer.pointShow() #displaying the computer's points
            if computer.points==player.points: #Checking if there's a tie
                print("\nTie! Nobody wins!\n")
            elif computer.points>player.points and prompt=="c" or player.points>computer.points and prompt=="p": #Checking if the player has won
                print("\nPlayer wins!\n")
                player.Won() #updating player win count
            else: #Else, the computer has won
                print("\nComputer Wins!\n")
                computer.Won() #updating the computers win count
            player.reset() #Resetting the player's point values
            computer.reset() #resetting the computer's point values
            input("Press enter to continue") #Giving a prompt to break between games
    print("\nGame Over!") #nice header to display a game over
    print("\nPlayer wins:",player.wincount) #player win totals
    print("\nComputer wins:",computer.wincount) #computer win totals
main()
```

#I have abided by the Wheaton Honor code in this work