```python
"""
This module contains the functions that will generate and randomize 52 unique cards.
"""
import random


def create_deck():
    """
    Creates a set of the 52 unique playing cards found within a deck.
    :return: Returns a deck of 52 playing cards.
    """
    deck = []
    rank_list = [str(2), str(3), str(4), str(5), str(6), str(7),
                 str(8), str(9), str(10), 'Jack', 'Queen', 'King', 'Ace']

    for rank in rank_list:
        deck.append(('Diamond', rank))
    for rank in rank_list:
        deck.append(('Heart', rank))
    for rank in rank_list:
        deck.append(('Spade', rank))
    for rank in rank_list:
        deck.append(('Club', rank))

    return deck


def random_deck(deck):
    """
    Randomizes the order of the deck of cards.
    :param: deck: The deck of cards, denoted as 'base' because it is still ordered.
    :return: Returns the deck, now with its card-order completely shuffled.
    """
    random.shuffle(deck)
    return deck
```

```python
"""
This module contains a single function to run. This function, using functions created in the two modules it
imports from, does the following: creates a deck of 52 unique playing cards, shuffles them, and draws 5
random cards from the deck. After drawing 5 cards 100,000 times (the type of hand drawn is recorded for each
draw), the number of each type of hand and the percentage of times it was drawn is tabulated on a table.
"""

'''
I affirm that I have carried out the attached academic endeavors with full academic honesty, in
accordance with the Union College Honor Code and the course syllabus.
'''

from Categorization import *
from Cards import *


def play_rounds():
    """
    This function, using functions created in the two modules this file imports from, does the following:
    creates a deck of 52 unique playing cards, shuffles them, and draws 5 random cards from the deck.
    After drawing 5 cards 100,000 times (the type of hand drawn is recorded for each draw), the number
    of each type of hand and the percentage of times it was drawn is tabulated on a table.
    """

    flush = 0
    pair = 0
    two_pair = 0
    high_card = 0

    print('# of hands     pairs       %      2 pairs       %      flushes       %    high card       %')
    no_of_draws = 0
    deck = create_deck()
    ran_deck = random_deck(deck)

    while no_of_draws <= 100000:
        if ((no_of_draws % 10000) == 0) and (no_of_draws != 0):
            flush_percent = round(((flush / no_of_draws) * 100), 2)
            two_pair_percent = round(((two_pair / no_of_draws) * 100), 2)
            pair_percent = round(((pair / no_of_draws) * 100), 2)
            high_card_percent = round(((high_card / no_of_draws) * 100), 2)
            print(f'{no_of_draws:10,d}{pair:10,d}{pair_percent:10}{two_pair:10,d}   {two_pair_percent:05.6}{flush:10,d}    {flush_percent:05.6}{high_card:10,d}{high_card_percent:10}')

        hand = draw_hand(ran_deck)

        result = determine_hand(hand)
        if result == 1:
            flush += 1
        elif result == 2:
            two_pair += 1
        elif result == 3:
            pair += 1
        else:
            high_card += 1

        if len(ran_deck) == 2:
            deck = create_deck()
            ran_deck = random_deck(deck)

        no_of_draws += 1
```

```python


play_rounds()

```

```python
"""
This module will contain the functions related to drawing cards and categorizing them.
"""


def rank_sort(index_one):
    """
    A function that exists to reverse the sorting order of hand of cards.
    When applied, this function will sort the hand by rank, rather than by suit.
    :param: index_one: Parameter generally is named after how the function will be sorting
    entries. This function exists to sort by the first index of a tuple, so its
    parameter is named index_one.
    :return: Returns the first index of an undetermined object.
    """
    return index_one[1]


def draw_hand(deck):
    """
    Draws a hand of 5 cards from the shuffled deck.
    :param: deck: the deck of 52 cards. At this point it is now randomized.
    :return: Returns a hand of 5 random cards from the deck.
    """

    hand = []
    for card in range(5):
        card = deck[0]
        hand.append(card)
        deck.pop(0)
    return hand


def determine_flush(hand):
    """
    A function that categorizes the card hand as a flush.
    :param: hand: A hand of 5 random cards.
    :return: Returns True if the hand is a flush, False otherwise.
    """

    hand.sort()
    suit = hand[0][0]

    for card in hand:
        if card[0] != suit:
            return False
    return True


def determine_pair(hand):
    """
    A function that categorizes the card hand as a pair.
    :param: hand: A hand of 5 random cards.
    :return: Returns True if the hand is a pair, False otherwise.
    """

    hand.sort(key=rank_sort)
    # Sorts in order of rank

    for i in range(0, 4):
        for j in range((i + 1), 5):
            if hand[i][1] == hand[j][1]:
                return True
    return False


def determine_two_pair(hand):
    """
    A function that categorizes the card hand as a two_pair.
    :param: hand: A hand of 5 random cards.
    :return: Returns True if the hand is a two_pair, False otherwise.
    """

    hand.sort(key=rank_sort)
    # Sorts in order of rank

    matching_card_1 = 0
    matching_card_2 = 0
    x = False

    for i in range(0, 4):
        for j in range((i + 1), 5):
            if hand[i][1] == hand[j][1]:
                matching_card_1 = hand[i]
                matching_card_2 = hand[j]
                x = True

    if x:
        hand.remove(matching_card_1)
        hand.remove(matching_card_2)
    else:
        return False

    for i in range(0, 2):
        for j in range((i + 1), 3):
            if hand[i][1] == hand[j][1]:
                hand.append(matching_card_1)
                hand.append(matching_card_2)
                return True

    hand.append(matching_card_1)
    hand.append(matching_card_2)
    return False


def determine_high_card():
    """
    A function that categorizes the card hand as a high_card. This function requires no parameters
    because it will always return True if run.
```

```python
    :return: Returns True if the hand is a high_card. The return will always be True, because
    every hand that is not a flush,pair, or two_pair must be a high_card.
    """

    return True


def determine_hand(hand):
    """
    This function will determine what category the hand of cards falls into.
    The categories include flushes, pairs, two_pairs, and high_cards.
    :param: hand: A hand of 5 random cards.
    :return: Returns a variable representing a number, the identity of which (1 through 4)
    is important for tabulating data relating to what hands are pulled and how often.
    """

    flush = 0
    pair = 0
    two_pair = 0
    high_card = 0

    if determine_flush(hand):
        flush += 1
        return flush

    if determine_two_pair(hand):
        two_pair += 2
        return two_pair

    if determine_pair(hand):
        pair += 3
        return pair

    if determine_high_card():
        high_card += 4
        return high_card

```