PROGRAM 1
PROJECT REPORT

Agent Alex Pruitt

Agent Obie Carnathan

BRIEF ASSIGNMENT OVERVIEW

- Using a deck of cards numbered 1 - 52, shuffle the cards fifteen times using several different methods to produce a permutation of the original deck. Through four different runs we will analyze the correlation coefficient by plotting the result on a graph and finding out when/if the cards return to their original order.

FUNCTIONALITY

- The program is set up to take user input to select of one of the four runs to be executed. Below is the screenshot of the prompt that is shown to the user upon running the program.
- After selecting one of the four runs, the program will execute the run, showing the deck and the value of 'r' at each iteration. It will also tell where the deck is most random and show a graph of the values of 'r'.

```
322  print("First Run: 1")
323  print("Second Run: 2")
324  print("Third Run: 3")
325  print("Fourth Run: 4\n")
326
327  option = input("Enter which run: ")
328
329  if (option == '1'):
330      print("\n", First_Run(15),"\n")
331  elif (option == '2'):
332      print("\n", Second_Run(15),"\n")
333  elif (option == '3'):
334      print("\n", Third_Run(15), "\n")
335  elif (option == '4'):
336      print("\n", Fourth_Run(15), "\n")
337  else:
338      print("Not a valid input!")
```

```
First Run: 1
Second Run: 2
Third Run: 3
Fourth Run: 4

Enter which run:
```

NOTE--→Since the only differences among the First, Second, Third, and Fourth run are the size and order of the deck we will cover the general functionality of a run function on the slides to come instead of discussing each individual run function. We will begin by showing a full screenshot of a run function on the next slide.

FULL SCREENSHOT OF FIRST_RUN

```
def First_Run(cycles):
    # hold the values of 'r' to find the minimum and maximum
    all_r = []

    # initialize cards
    cards = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27,
             28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52]

    # Finds the constants for the equation to calculate 'r'
    sumsq = 0
    sumi = 0
    for i in range(1, len(cards)):
        sumsq += ((cards[i])**2)
        sumi += (cards[i])

    sqsum = sumi*sumi

    # Calculates 'r', given the deck, with the constants calculated above
    def calc_r(new_deck):

        n = (len(new_deck))

        Enum = 0
        for i in range(1, n+1):
            Enum += (i * (new_deck[i-1]))

        r = (((n * Enum) - sqsum) / ((n * sumsq) - sqsum))
        r = (round(r, 3))
        all_r.append(r)
        return r
```

```
    # shuffles the deck in a perfect riffle
    # given: deck to be shuffled, 0
    def shuffle(deck, iterations, half, count):
        # Stops shuffling after a certain iterations
        if (iterations == 0):
            return deck

        cards = []

        # shuffles the deck in a perfect riffle
        for i in range(0, half):
            cards.append(deck[i])
            cards.append(deck[half+i])

        count = count+1
        print(count, ". New deck: ", cards)
        print("r-iterations: ", calc_r(cards), "\n")

        return shuffle(cards, (iterations-1), half, count)


    new_deck = shuffle(cards, cycles, (len(cards)//2), 0)

    min_val = all_r[0]
    min_index = 0
    for i in range(0, len(all_r)):
        if (min_val > all_r[i]):
            min_val = all_r[i]
            min_index = (i+1)

    print("\nThe list is at the most random after", min_index, "iterations")
    print("The 'r' value of it is", min_val)

    plt.title("First Run")
    plt.xlabel("Iteration")
    plt.ylabel("r-value")
    plt.axis([1,15, -1 ,1])
    xVal = list(range(1,16))
    plt.plot(xVal, all_r)
    plt.show()

    return new_deck
```

RUN FUNCTIONS

- Each run function (First_Run, Second_Run, Third_Run, Fourth_Run) begins by initializing an empty array to hold the values of ‘r’ and another array of numbers to represent either a single deck or two decks of cards. From there the constants ‘sumsq’,‘sumi’, and ‘sqsum’ are calculated by looping through the card array. Example below…

```
10  def First_Run(cycles):
11      # hold the values of 'r' to find the minimum and maximum
12      all_r = []
13
14      # initialize cards
15      cards = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27,
16               28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52]
17
18      # Finds the constants for the equation to calculate 'r'
19      sumsq = 0
20      sumi = 0
21      for i in range(1, len(cards)):
22          sumsq += ((cards[i])**2)
23          sumi += (cards[i])
24
25      sqsum = sumi*sumi
```

CALC_R FUNCTION

- Every run function has a calc_r function. The purpose of calc_r is to calculate the correlation coefficient ‘r’ for a ‘new_deck’.
- The function begins by calculating the length of ‘new_deck’ and assigning it to the variable ‘n’. It then initializes a variable ‘Enum’ to 0, and uses a for loop to iterate over each value of ‘I’ from 1 to ‘n’. On each iteration of the loop, the product of ‘I’ and the ‘(i-1)th’ element of ‘new_deck’ is added to ‘Enum’.
- The correlation coefficient ‘r’ is then calculated using the formula ‘r = (((n * Enum) - sqsum) / ((n * sumsq) - sqsum))’. The value of r is rounded to 3 decimal places using the round() function, and then appended to a list called ‘all_r’.
- Finally, the function returns the value of r.

CALC_R FUNCTION

```
27      # Calculates 'r', given the deck, with the constants calculated above
28      def calc_r(new_deck):
29
30          n = (len(new_deck))
31
32          Enum = 0
33          for i in range(1, n+1):
34              Enum += (i * (new_deck[i-1]))
35
36          r = (((n * Enum) - sqsum) / ((n * sumsq) - sqsum))
37          r = (round(r, 3))
38          all_r.append(r)
39          return r
```

SHUFFLE FUNCTION

- Every Run function has a shuffle function. The function begins by checking whether the desired number of iterations has been reached (i.e., iterations == 0). If so, the function simply returns the original deck without shuffling it any further.

- If the desired number of iterations has not been reached, the function creates an empty list called ‘cards’. It then performs a perfect riffle shuffle on ‘deck’ by iterating over the first half of the deck (range(0, half)) and appending alternating cards to cards (cards.append(deck[i]) and cards.append(deck[half+i])).

- The function then increments the count variable and prints out the new deck (print(count, ". New deck: ", cards)) and the correlation coefficient for the new deck (print("r-iterations: ", calc_r(cards), "\n")).

- Finally, the function recursively calls itself with the shuffled deck (shuffle(cards, (iterations-1), half, count)) and decrements iterations by 1. This process continues until the desired number of iterations is reached, at which point the function returns the final shuffled deck.

SHUFFLE FUNCTION

```
42      # shuffles the deck in a perfect riffle
43      # given: deck to be shuffled, 0
44      def shuffle(deck, iterations, half, count):
45          # Stops shuffling after a certain iterations
46          if (iterations == 0):
47              return deck
48
49          cards = []
50
51          # shuffles the deck in a perfect riffle
52          for i in range(0, half):
53              cards.append(deck[i])
54              cards.append(deck[half+i])
55
56          count = count+1
57          print(count, ". New deck: ", cards)
58          print("r-iterations: ", calc_r(cards), "\n")
59
60          return shuffle(cards, (iterations-1), half, count)
```

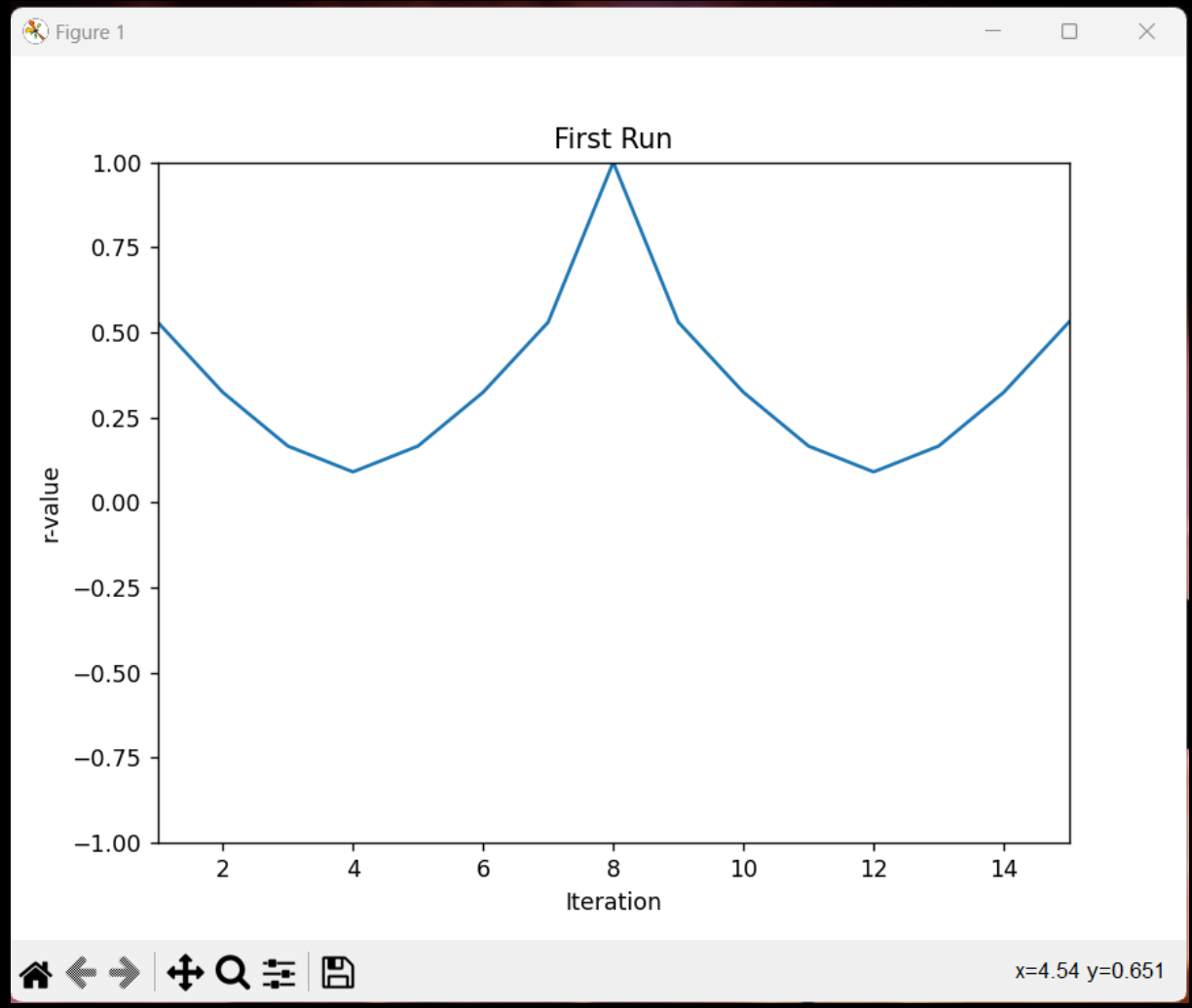
RUN FUNCTIONS

- After defining both helper functions we are now able to find out when the deck is most random among the shuffles. The code first shuffles the deck of cards by calling the shuffle function. The code then finds the minimum "r-value" in the ‘all_r’ list and the index of that minimum value. The minimum "r-value" indicates the most random permutation of the deck, and the index of that value indicates the iteration number when that permutation was generated.

- Finally, the code plots the "r-value" of each shuffle against the iteration number using the Matplotlib library. The resulting plot shows how the "r-value" changes as the deck is shuffled multiple times, and it highlights the iteration number when the most random permutation was generated.

OUTPUT OF FIRST_RUN

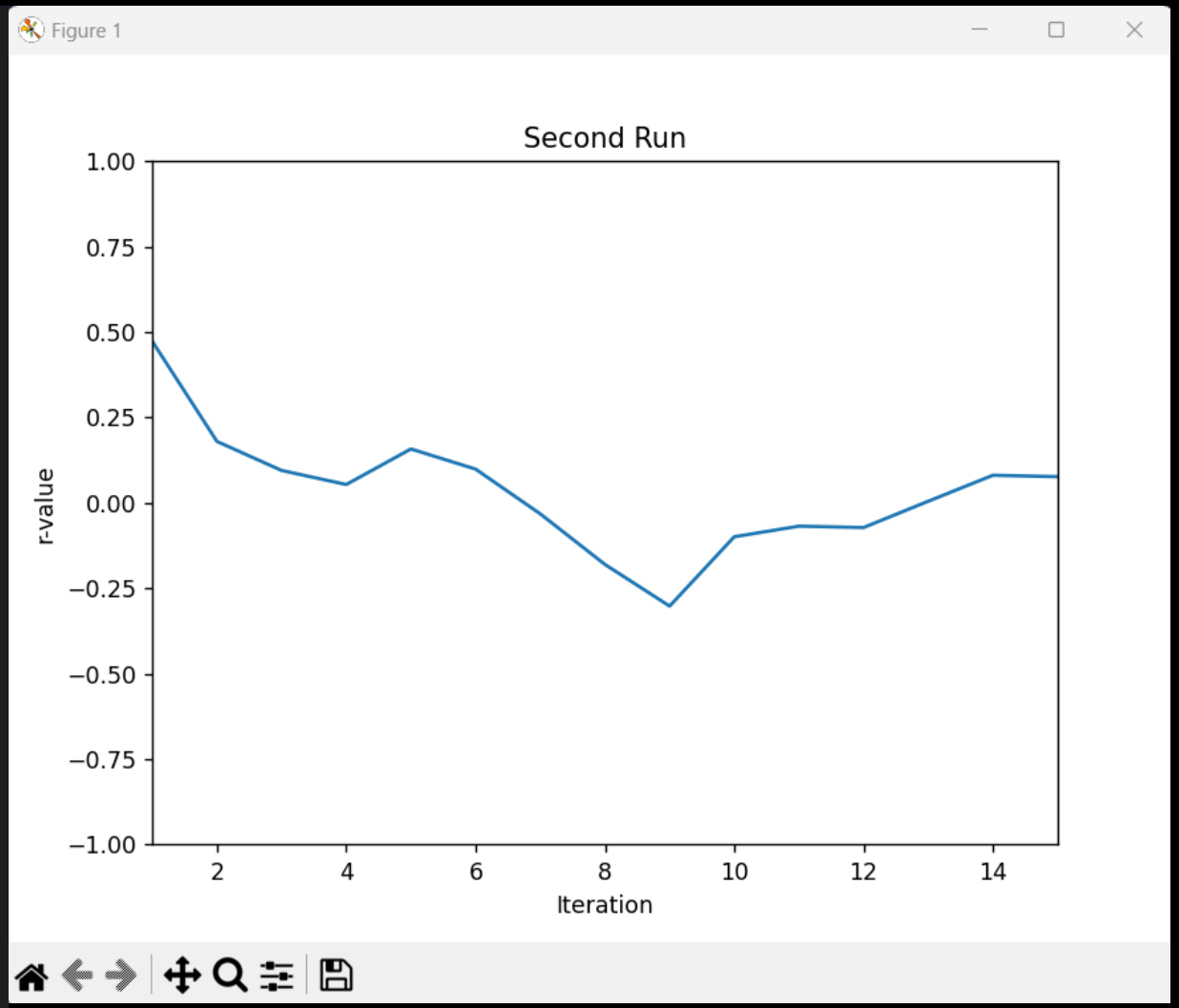
```
Enter which run: 1
1 . New deck:  [1, 27, 2, 28, 3, 29, 4, 30, 5, 31, 6, 32, 7, 33, 8, 34, 9, 35, 10, 36, 11, 37, 12, 38, 13, 39, 14, 40, 15, 41, 16, 42,
17, 43, 18, 44, 19, 45, 20, 46, 21, 47, 22, 48, 23, 49, 24, 50, 25, 51, 26, 52]
r-iterations:  0.53

2 . New deck:  [1, 14, 27, 40, 2, 15, 28, 41, 3, 16, 29, 42, 4, 17, 30, 43, 5, 18, 31, 44, 6, 19, 32, 45, 7, 20, 33, 46, 8, 21, 34, 47,
 9, 22, 35, 48, 10, 23, 36, 49, 11, 24, 37, 50, 12, 25, 38, 51, 13, 26, 39, 52]
r-iterations:  0.324

3 . New deck:  [1, 33, 14, 46, 27, 8, 40, 21, 2, 34, 15, 47, 28, 9, 41, 22, 3, 35, 16, 48, 29, 10, 42, 23, 4, 36, 17, 49, 30, 11, 43, 2
4, 5, 37, 18, 50, 31, 12, 44, 25, 6, 38, 19, 51, 32, 13, 45, 26, 7, 39, 20, 52]
r-iterations:  0.166

4 . New deck:  [1, 17, 33, 49, 14, 30, 46, 11, 27, 43, 8, 24, 40, 5, 21, 37, 2, 18, 34, 50, 15, 31, 47, 12, 28, 44, 9, 25, 41, 6, 22, 3
8, 3, 19, 35, 51, 16, 32, 48, 13, 29, 45, 10, 26, 42, 7, 23, 39, 4, 20, 36, 52]
r-iterations:  0.09

5 . New deck:  [1, 9, 17, 25, 33, 41, 49, 6, 14, 22, 30, 38, 46, 3, 11, 19, 27, 35, 43, 51, 8, 16, 24, 32, 40, 48, 5, 13, 21, 29, 37, 4
5, 2, 10, 18, 26, 34, 42, 50, 7, 15, 23, 31, 39, 47, 4, 12, 20, 28, 36, 44, 52]
r-iterations:  0.166

6 . New deck:  [1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 3, 7, 11, 15, 19, 23
, 27, 31, 35, 39, 43, 47, 51, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52]
r-iterations:  0.324

7 . New deck:  [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 2, 4, 6, 8, 10, 12,
14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52]
r-iterations:  0.53

8 . New deck:  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32,
33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52]
r-iterations:  1.0

9 . New deck:  [1, 27, 2, 28, 3, 29, 4, 30, 5, 31, 6, 32, 7, 33, 8, 34, 9, 35, 10, 36, 11, 37, 12, 38, 13, 39, 14, 40, 15, 41, 16, 42,
17, 43, 18, 44, 19, 45, 20, 46, 21, 47, 22, 48, 23, 49, 24, 50, 25, 51, 26, 52]
r-iterations:  0.53

10 . New deck:  [1, 14, 27, 40, 2, 15, 28, 41, 3, 16, 29, 42, 4, 17, 30, 43, 5, 18, 31, 44, 6, 19, 32, 45, 7, 20, 33, 46, 8, 21, 34, 47
, 9, 22, 35, 48, 10, 23, 36, 49, 11, 24, 37, 50, 12, 25, 38, 51, 13, 26, 39, 52]
r-iterations:  0.324

11 . New deck:  [1, 33, 14, 46, 27, 8, 40, 21, 2, 34, 15, 47, 28, 9, 41, 22, 3, 35, 16, 48, 29, 10, 42, 23, 4, 36, 17, 49, 30, 11, 43,
24, 5, 37, 18, 50, 31, 12, 44, 25, 6, 38, 19, 51, 32, 13, 45, 26, 7, 39, 20, 52]
r-iterations:  0.166

12 . New deck:  [1, 17, 33, 49, 14, 30, 46, 11, 27, 43, 8, 24, 40, 5, 21, 37, 2, 18, 34, 50, 15, 31, 47, 12, 28, 44, 9, 25, 41, 6, 22,
38, 3, 19, 35, 51, 16, 32, 48, 13, 29, 45, 10, 26, 42, 7, 23, 39, 4, 20, 36, 52]
r-iterations:  0.09

13 . New deck:  [1, 9, 17, 25, 33, 41, 49, 6, 14, 22, 30, 38, 46, 3, 11, 19, 27, 35, 43, 51, 8, 16, 24, 32, 40, 48, 5, 13, 21, 29, 37,
45, 2, 10, 18, 26, 34, 42, 50, 7, 15, 23, 31, 39, 47, 4, 12, 20, 28, 36, 44, 52]
r-iterations:  0.166

14 . New deck:  [1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46, 50, 3, 7, 11, 15, 19, 2
3, 27, 31, 35, 39, 43, 47, 51, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52]
r-iterations:  0.324

15 . New deck:  [1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 2, 4, 6, 8, 10, 12,
 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52]
r-iterations:  0.53


The list is at the most random after 4 iterations
The 'r' value of it is 0.09
```



At the 8th iteration the cards are back in their original order. Therefore 15 runs is enough to return the original order.

OUTPUT OF SECOND_RUN

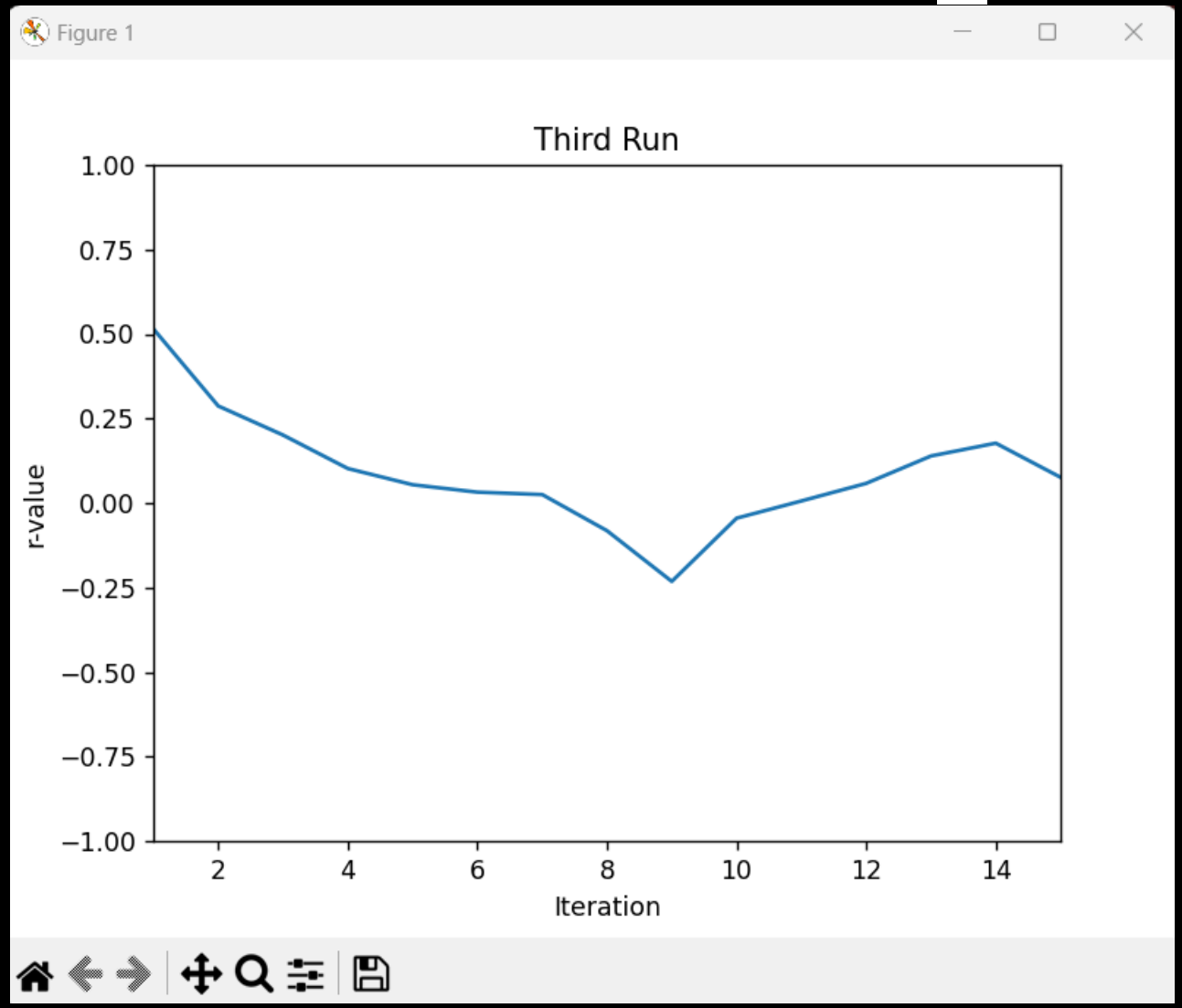
```
Enter which run: 2
1 . New deck:  [27, 1, 28, 2, 29, 3, 30, 4, 31, 5, 32, 6, 33, 7, 34, 8, 35, 9, 36, 10, 37, 11, 38, 12, 39, 13, 40, 14, 41, 15, 42, 16,
43, 17, 44, 18, 45, 19, 46, 20, 47, 21, 48, 22, 49, 23, 50, 24, 51, 25, 52, 26]
r-iterations:  0.473

2 . New deck:  [40, 27, 14, 1, 41, 28, 15, 2, 42, 29, 16, 3, 43, 30, 17, 4, 44, 31, 18, 5, 45, 32, 19, 6, 46, 33, 20, 7, 47, 34, 21, 8,
 48, 35, 22, 9, 49, 36, 23, 10, 50, 37, 24, 11, 51, 38, 25, 12, 52, 39, 26, 13]
r-iterations:  0.18

3 . New deck:  [20, 40, 7, 27, 47, 14, 34, 1, 21, 41, 8, 28, 48, 15, 35, 2, 22, 42, 9, 29, 49, 16, 36, 3, 23, 43, 10, 30, 50, 17, 37, 4
, 24, 44, 11, 31, 51, 18, 38, 5, 25, 45, 12, 32, 52, 19, 39, 6, 26, 46, 13, 33]
r-iterations:  0.095

4 . New deck:  [10, 20, 30, 40, 50, 7, 17, 27, 37, 47, 4, 14, 24, 34, 44, 1, 11, 21, 31, 41, 51, 8, 18, 28, 38, 48, 5, 15, 25, 35, 45,
2, 12, 22, 32, 42, 52, 9, 19, 29, 39, 49, 6, 16, 26, 36, 46, 3, 13, 23, 33, 43]
r-iterations:  0.054

5 . New deck:  [5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 2, 7, 12, 17, 22, 27, 32, 37, 42, 47, 52, 4, 9, 14, 19, 24, 29, 34, 39, 44, 49,
1, 6, 11, 16, 21, 26, 31, 36, 41, 46, 51, 3, 8, 13, 18, 23, 28, 33, 38, 43, 48]
r-iterations:  0.158

6 . New deck:  [29, 5, 34, 10, 39, 15, 44, 20, 49, 25, 1, 30, 6, 35, 11, 40, 16, 45, 21, 50, 26, 2, 31, 7, 36, 12, 41, 17, 46, 22, 51,
27, 3, 32, 8, 37, 13, 42, 18, 47, 23, 52, 28, 4, 33, 9, 38, 14, 43, 19, 48, 24]
r-iterations:  0.099

7 . New deck:  [41, 29, 17, 5, 46, 34, 22, 10, 51, 39, 27, 15, 3, 44, 32, 20, 8, 49, 37, 25, 13, 1, 42, 30, 18, 6, 47, 35, 23, 11, 52,
40, 28, 16, 4, 45, 33, 21, 9, 50, 38, 26, 14, 2, 43, 31, 19, 7, 48, 36, 24, 12]
r-iterations:  -0.032

8 . New deck:  [47, 41, 35, 29, 23, 17, 11, 5, 52, 46, 40, 34, 28, 22, 16, 10, 4, 51, 45, 39, 33, 27, 21, 15, 9, 3, 50, 44, 38, 32, 26,
 20, 14, 8, 2, 49, 43, 37, 31, 25, 19, 13, 7, 1, 48, 42, 36, 30, 24, 18, 12, 6]
r-iterations:  -0.18

9 . New deck:  [50, 47, 44, 41, 38, 35, 32, 29, 26, 23, 20, 17, 14, 11, 8, 5, 2, 52, 49, 46, 43, 40, 37, 34, 31, 28, 25, 22, 19, 16, 13
, 10, 7, 4, 1, 51, 48, 45, 42, 39, 36, 33, 30, 27, 24, 21, 18, 15, 12, 9, 6, 3]
r-iterations:  -0.302

10 . New deck:  [25, 50, 22, 47, 19, 44, 16, 41, 13, 38, 10, 35, 7, 32, 4, 29, 1, 26, 51, 23, 48, 20, 45, 17, 42, 14, 39, 11, 36, 8, 33
, 5, 30, 2, 27, 52, 24, 49, 21, 46, 18, 43, 15, 40, 12, 37, 9, 34, 6, 31, 3, 28]
r-iterations:  -0.099

11 . New deck:  [39, 25, 11, 50, 36, 22, 8, 47, 33, 19, 5, 44, 30, 16, 2, 41, 27, 13, 52, 38, 24, 10, 49, 35, 21, 7, 46, 32, 18, 4, 43,
 29, 15, 1, 40, 26, 12, 51, 37, 23, 9, 48, 34, 20, 6, 45, 31, 17, 3, 42, 28, 14]
r-iterations:  -0.068

12 . New deck:  [46, 39, 32, 25, 18, 11, 4, 50, 43, 36, 29, 22, 15, 8, 1, 47, 40, 33, 26, 19, 12, 5, 51, 44, 37, 30, 23, 16, 9, 2, 48,
41, 34, 27, 20, 13, 6, 52, 45, 38, 31, 24, 17, 10, 3, 49, 42, 35, 28, 21, 14, 7]
r-iterations:  -0.072

13 . New deck:  [23, 46, 16, 39, 9, 32, 2, 25, 48, 18, 41, 11, 34, 4, 27, 50, 20, 43, 13, 36, 6, 29, 52, 22, 45, 15, 38, 8, 31, 1, 24,
47, 17, 40, 10, 33, 3, 26, 49, 19, 42, 12, 35, 5, 28, 51, 21, 44, 14, 37, 7, 30]
r-iterations:  0.005

14 . New deck:  [38, 23, 8, 46, 31, 16, 1, 39, 24, 9, 47, 32, 17, 2, 40, 25, 10, 48, 33, 18, 3, 41, 26, 11, 49, 34, 19, 4, 42, 27, 12,
50, 35, 20, 5, 43, 28, 13, 51, 36, 21, 6, 44, 29, 14, 52, 37, 22, 7, 45, 30, 15]
r-iterations:  0.081

15 . New deck:  [19, 38, 4, 23, 42, 8, 27, 46, 12, 31, 50, 16, 35, 1, 20, 39, 5, 24, 43, 9, 28, 47, 13, 32, 51, 17, 36, 2, 21, 40, 6, 2
5, 44, 10, 29, 48, 14, 33, 52, 18, 37, 3, 22, 41, 7, 26, 45, 11, 30, 49, 15, 34]
r-iterations:  0.077


The list is at the most random after 9 iterations
The 'r' value of it is -0.302
```



The cards do not return to their original order after 15 iterations. It takes 52 iterations to reach the original order.

OUTPUT OF THIRD_RUN

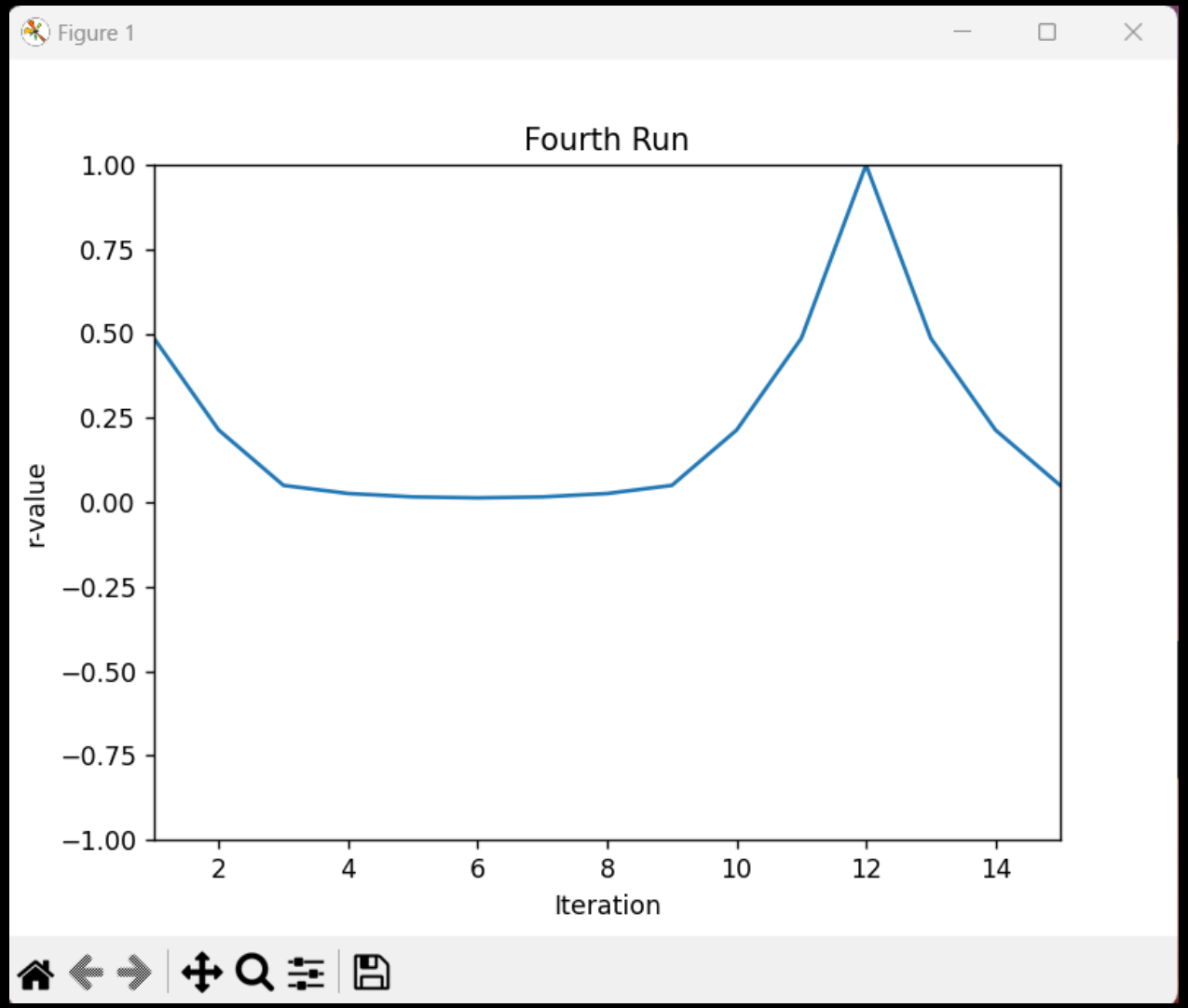
```
Enter which run: 3
1 . New deck:  [1, 53, 2, 54, 3, 55, 4, 56, 5, 57, 6, 58, 7, 59, 8, 60, 9, 61, 10, 62, 11, 63, 12, 64, 13, 65, 14, 66, 15, 67, 16, 68,
 17, 69, 18, 70, 19, 71, 20, 72, 21, 73, 22, 74, 23, 75, 24, 76, 25, 77, 26, 78, 27, 79, 28, 80, 29, 81, 30, 82, 31, 83, 32, 84, 33, 85,
 34, 86, 35, 87, 36, 88, 37, 89, 38, 90, 39, 91, 40, 92, 41, 93, 42, 94, 43, 95, 44, 96, 45, 97, 46, 98, 47, 99, 48, 100, 49, 101, 50,
102, 51, 103, 52, 104]
r-iterations:  0.515

2 . New deck:  [1, 27, 53, 79, 2, 28, 54, 80, 3, 29, 55, 81, 4, 30, 56, 82, 5, 31, 57, 83, 6, 32, 58, 84, 7, 33, 59, 85, 8, 34, 60, 86,
 9, 35, 61, 87, 10, 36, 62, 88, 11, 37, 63, 89, 12, 38, 64, 90, 13, 39, 65, 91, 14, 40, 66, 92, 15, 41, 67, 93, 16, 42, 68, 94, 17, 43,
 69, 95, 18, 44, 70, 96, 19, 45, 71, 97, 20, 46, 72, 98, 21, 47, 73, 99, 22, 48, 74, 100, 23, 49, 75, 101, 24, 50, 76, 102, 25, 51, 77,
 103, 26, 52, 78, 104]
r-iterations:  0.287

3 . New deck:  [1, 14, 27, 40, 53, 66, 79, 92, 2, 15, 28, 41, 54, 67, 80, 93, 3, 16, 29, 42, 55, 68, 81, 94, 4, 17, 30, 43, 56, 69, 82,
 95, 5, 18, 31, 44, 57, 70, 83, 96, 6, 19, 32, 45, 58, 71, 84, 97, 7, 20, 33, 46, 59, 72, 85, 98, 8, 21, 34, 47, 60, 73, 86, 99, 9, 22,
 35, 48, 61, 74, 87, 100, 10, 23, 36, 49, 62, 75, 88, 101, 11, 24, 37, 50, 63, 76, 89, 102, 12, 25, 38, 51, 64, 77, 90, 103, 13, 26, 39
, 52, 65, 78, 91, 104]
r-iterations:  0.201

4 . New deck:  [1, 59, 14, 72, 27, 85, 40, 98, 53, 8, 66, 21, 79, 34, 92, 47, 2, 60, 15, 73, 28, 86, 41, 99, 54, 9, 67, 22, 80, 35, 93,
 48, 3, 61, 16, 74, 29, 87, 42, 100, 55, 10, 68, 23, 81, 36, 94, 49, 4, 62, 17, 75, 30, 88, 43, 101, 56, 11, 69, 24, 82, 37, 95, 50, 5,
 63, 18, 76, 31, 89, 44, 102, 57, 12, 70, 25, 83, 38, 96, 51, 6, 64, 19, 77, 32, 90, 45, 103, 58, 13, 71, 26, 84, 39, 97, 52, 7, 65, 20
, 78, 33, 91, 46, 104]
r-iterations:  0.102

5 . New deck:  [1, 30, 59, 88, 14, 43, 72, 101, 27, 56, 85, 11, 40, 69, 98, 24, 53, 82, 8, 37, 66, 95, 21, 50, 79, 5, 34, 63, 92, 18, 4
7, 76, 2, 31, 60, 89, 15, 44, 73, 102, 28, 57, 86, 12, 41, 70, 99, 25, 54, 83, 9, 38, 67, 96, 22, 51, 80, 6, 35, 64, 93, 19, 48, 77, 3,
 32, 61, 90, 16, 45, 74, 103, 29, 58, 87, 13, 42, 71, 100, 26, 55, 84, 10, 39, 68, 97, 23, 52, 81, 7, 36, 65, 94, 20, 49, 78, 4, 33, 62
, 91, 17, 46, 75, 104]
r-iterations:  0.054

6 . New deck:  [1, 67, 30, 96, 59, 22, 88, 51, 14, 80, 43, 6, 72, 35, 101, 64, 27, 93, 56, 19, 85, 48, 11, 77, 40, 3, 69, 32, 98, 61, 2
4, 90, 53, 16, 82, 45, 8, 74, 37, 103, 66, 29, 95, 58, 21, 87, 50, 13, 79, 42, 5, 71, 34, 100, 63, 26, 92, 55, 18, 84, 47, 10, 76, 39,
 2, 68, 31, 97, 60, 23, 89, 52, 15, 81, 44, 7, 73, 36, 102, 65, 28, 94, 57, 20, 86, 49, 12, 78, 41, 4, 70, 33, 99, 62, 25, 91, 54, 17, 8
3, 46, 9, 75, 38, 104]
r-iterations:  0.032

7 . New deck:  [1, 34, 67, 100, 30, 63, 96, 26, 59, 92, 22, 55, 88, 18, 51, 84, 14, 47, 80, 10, 43, 76, 6, 39, 72, 2, 35, 68, 101, 31,
 64, 97, 27, 60, 93, 23, 56, 89, 19, 52, 85, 15, 48, 81, 11, 44, 77, 7, 40, 73, 3, 36, 69, 102, 32, 65, 98, 28, 61, 94, 24, 57, 90, 20,
 53, 86, 16, 49, 82, 12, 45, 78, 8, 41, 74, 4, 37, 70, 103, 33, 66, 99, 29, 62, 95, 25, 58, 91, 21, 54, 87, 17, 50, 83, 13, 46, 79, 9, 4
2, 75, 5, 38, 71, 104]
r-iterations:  0.025

8 . New deck:  [1, 69, 34, 102, 67, 32, 100, 65, 30, 98, 63, 28, 96, 61, 26, 94, 59, 24, 92, 57, 22, 90, 55, 20, 88, 53, 18, 86, 51, 16
, 84, 49, 14, 82, 47, 12, 80, 45, 10, 78, 43, 8, 76, 41, 6, 74, 39, 4, 72, 37, 2, 70, 35, 103, 68, 33, 101, 66, 31, 99, 64, 29, 97, 62,
 27, 95, 60, 25, 93, 58, 23, 91, 56, 21, 89, 54, 19, 87, 52, 17, 85, 50, 15, 83, 48, 13, 81, 46, 11, 79, 44, 9, 77, 42, 7, 75, 40, 5, 7
3, 38, 3, 71, 36, 104]
r-iterations:  -0.082

9 . New deck:  [1, 35, 69, 103, 34, 68, 102, 33, 67, 101, 32, 66, 100, 31, 65, 99, 30, 64, 98, 29, 63, 97, 28, 62, 96, 27, 61, 95, 26,
 60, 94, 25, 59, 93, 24, 58, 92, 23, 57, 91, 22, 56, 90, 21, 55, 89, 20, 54, 88, 19, 53, 87, 18, 52, 86, 17, 51, 85, 16, 50, 84, 15, 49,
 83, 14, 48, 82, 13, 47, 81, 12, 46, 80, 11, 45, 79, 10, 44, 78, 9, 43, 77, 8, 42, 76, 7, 41, 75, 6, 40, 74, 5, 39, 73, 4, 38, 72, 3, 3
7, 71, 2, 36, 70, 104]
r-iterations:  -0.232

10 . New deck:  [1, 18, 35, 52, 69, 86, 103, 17, 34, 51, 68, 85, 102, 16, 33, 50, 67, 84, 101, 15, 32, 49, 66, 83, 100, 14, 31, 48, 65,
 82, 99, 13, 30, 47, 64, 81, 98, 12, 29, 46, 63, 80, 97, 11, 28, 45, 62, 79, 96, 10, 27, 44, 61, 78, 95, 9, 26, 43, 60, 77, 94, 8, 25,
42, 59, 76, 93, 7, 24, 41, 58, 75, 92, 6, 23, 40, 57, 74, 91, 5, 22, 39, 56, 73, 90, 4, 21, 38, 55, 72, 89, 3, 20, 37, 54, 71, 88, 2, 1
9, 36, 53, 70, 87, 104]
r-iterations:  -0.045

11 . New deck:  [1, 61, 18, 78, 35, 95, 52, 9, 69, 26, 86, 43, 103, 60, 17, 77, 34, 94, 51, 8, 68, 25, 85, 42, 102, 59, 16, 76, 33, 93,
 50, 7, 67, 24, 84, 41, 101, 58, 15, 75, 32, 92, 49, 6, 66, 23, 83, 40, 100, 57, 14, 74, 31, 91, 48, 5, 65, 22, 82, 39, 99, 56, 13, 73,
 30, 90, 47, 4, 64, 21, 81, 38, 98, 55, 12, 72, 29, 89, 46, 3, 63, 20, 80, 37, 97, 54, 11, 71, 28, 88, 45, 2, 62, 19, 79, 36, 96, 53, 1
0, 70, 27, 87, 44, 104]
r-iterations:  0.006

12 . New deck:  [1, 31, 61, 91, 18, 48, 78, 5, 35, 65, 95, 22, 52, 82, 9, 39, 69, 99, 26, 56, 86, 13, 43, 73, 103, 30, 60, 90, 17, 47,
77, 4, 34, 64, 94, 21, 51, 81, 8, 38, 68, 98, 25, 55, 85, 12, 42, 72, 102, 29, 59, 89, 16, 46, 76, 3, 33, 63, 93, 20, 50, 80, 7, 37, 67
, 97, 24, 54, 84, 11, 41, 71, 101, 28, 58, 88, 15, 45, 75, 2, 32, 62, 92, 19, 49, 79, 6, 36, 66, 96, 23, 53, 83, 10, 40, 70, 100, 27, 5
7, 87, 14, 44, 74, 104]
r-iterations:  0.058

13 . New deck:  [1, 16, 31, 46, 61, 76, 91, 3, 18, 33, 48, 63, 78, 93, 5, 20, 35, 50, 65, 80, 95, 7, 22, 37, 52, 67, 82, 97, 9, 24, 39,
 54, 69, 84, 99, 11, 26, 41, 56, 71, 86, 101, 13, 28, 43, 58, 73, 88, 103, 15, 30, 45, 60, 75, 90, 2, 17, 32, 47, 62, 77, 92, 4, 19, 34
, 49, 64, 79, 94, 6, 21, 36, 51, 66, 81, 96, 8, 23, 38, 53, 68, 83, 98, 10, 25, 40, 55, 70, 85, 100, 12, 27, 42, 57, 72, 87, 102, 14, 2
9, 44, 59, 74, 89, 104]
r-iterations:  0.139

14 . New deck:  [1, 60, 16, 75, 31, 90, 46, 2, 61, 17, 76, 32, 91, 47, 3, 62, 18, 77, 33, 92, 48, 4, 63, 19, 78, 34, 93, 49, 5, 64, 20,
 79, 35, 94, 50, 6, 65, 21, 80, 36, 95, 51, 7, 66, 22, 81, 37, 96, 52, 8, 67, 23, 82, 38, 97, 53, 9, 68, 24, 83, 39, 98, 54, 10, 69, 25
, 84, 40, 99, 55, 11, 70, 26, 85, 41, 100, 56, 12, 71, 27, 86, 42, 101, 57, 13, 72, 28, 87, 43, 102, 58, 14, 73, 29, 88, 44, 103, 59, 1
5, 74, 30, 89, 45, 104]
r-iterations:  0.177

15 . New deck:  [1, 82, 60, 38, 16, 97, 75, 53, 31, 9, 90, 68, 46, 24, 2, 83, 61, 39, 17, 98, 76, 54, 32, 10, 91, 69, 47, 25, 3, 84, 62
, 40, 18, 99, 77, 55, 33, 11, 92, 70, 48, 26, 4, 85, 63, 41, 19, 100, 78, 56, 34, 12, 93, 71, 49, 27, 5, 86, 64, 42, 20, 101, 79, 57, 3
5, 13, 94, 72, 50, 28, 6, 87, 65, 43, 21, 102, 80, 58, 36, 14, 95, 73, 51, 29, 7, 88, 66, 44, 22, 103, 81, 59, 37, 15, 96, 74, 52, 30,
 8, 89, 67, 45, 23, 104]
r-iterations:  0.076


The list is at the most random after 9 iterations
The 'r' value of it is -0.232
```



The cards do not return to their original order after 15 iterations. It takes 51 iterations to reach the original order.

OUTPUT OF FOURTH_RUN

```
Enter which run: 4
1 . New deck:  [53, 1, 54, 2, 55, 3, 56, 4, 57, 5, 58, 6, 59, 7, 60, 8, 61, 9, 62, 10, 63, 11, 64, 12, 65, 13, 66, 14, 67, 15, 68, 16,
 69, 17, 70, 18, 71, 19, 72, 20, 73, 21, 74, 22, 75, 23, 76, 24, 77, 25, 78, 26, 79, 27, 80, 28, 81, 29, 82, 30, 83, 31, 84, 32, 85, 33,
 86, 34, 87, 35, 88, 36, 89, 37, 90, 38, 91, 39, 92, 40, 93, 41, 94, 42, 95, 43, 96, 44, 97, 45, 98, 46, 99, 47, 100, 48, 101, 49, 102,
 50, 103, 51, 104, 52]
r-iterations:  0.486

2 . New deck:  [79, 53, 27, 1, 80, 54, 28, 2, 81, 55, 29, 3, 82, 56, 30, 4, 83, 57, 31, 5, 84, 58, 32, 6, 85, 59, 33, 7, 86, 60, 34, 8,
 87, 61, 35, 9, 88, 62, 36, 10, 89, 63, 37, 11, 90, 64, 38, 12, 91, 65, 39, 13, 92, 66, 40, 14, 93, 67, 41, 15, 94, 68, 42, 16, 95, 69,
 43, 17, 96, 70, 44, 18, 97, 71, 45, 19, 98, 72, 46, 20, 99, 73, 47, 21, 100, 74, 48, 22, 101, 75, 49, 23, 102, 76, 50, 24, 103, 77, 51
, 25, 104, 78, 52, 26]
r-iterations:  0.214

3 . New deck:  [92, 79, 66, 53, 40, 27, 14, 1, 93, 80, 67, 54, 41, 28, 15, 2, 94, 81, 68, 55, 42, 29, 16, 3, 95, 82, 69, 56, 43, 30, 17
, 4, 96, 83, 70, 57, 44, 31, 18, 5, 97, 84, 71, 58, 45, 32, 19, 6, 98, 85, 72, 59, 46, 33, 20, 7, 99, 86, 73, 60, 47, 34, 21, 8, 100, 8
7, 74, 61, 48, 35, 22, 9, 101, 88, 75, 62, 49, 36, 23, 10, 102, 89, 76, 63, 50, 37, 24, 11, 103, 90, 77, 64, 51, 38, 25, 12, 104, 91, 7
8, 65, 52, 39, 26, 13]
r-iterations:  0.05

4 . New deck:  [46, 92, 33, 79, 20, 66, 7, 53, 99, 40, 86, 27, 73, 14, 60, 1, 47, 93, 34, 80, 21, 67, 8, 54, 100, 41, 87, 28, 74, 15, 6
1, 2, 48, 94, 35, 81, 22, 68, 9, 55, 101, 42, 88, 29, 75, 16, 62, 3, 49, 95, 36, 82, 23, 69, 10, 56, 102, 43, 89, 30, 76, 17, 63, 4, 50
, 96, 37, 83, 24, 70, 11, 57, 103, 44, 90, 31, 77, 18, 64, 5, 51, 97, 38, 84, 25, 71, 12, 58, 104, 45, 91, 32, 78, 19, 65, 6, 52, 98, 3
9, 85, 26, 72, 13, 59]
r-iterations:  0.026

5 . New deck:  [23, 46, 69, 92, 10, 33, 56, 79, 102, 20, 43, 66, 89, 7, 30, 53, 76, 99, 17, 40, 63, 86, 4, 27, 50, 73, 96, 14, 37, 60,
83, 1, 24, 47, 70, 93, 11, 34, 57, 80, 103, 21, 44, 67, 90, 8, 31, 54, 77, 100, 18, 41, 64, 87, 5, 28, 51, 74, 97, 15, 38, 61, 84, 2, 2
5, 48, 71, 94, 12, 35, 58, 81, 104, 22, 45, 68, 91, 9, 32, 55, 78, 101, 19, 42, 65, 88, 6, 29, 52, 75, 98, 16, 39, 62, 85, 3, 26, 49, 7
2, 95, 13, 36, 59, 82]
r-iterations:  0.016

6 . New deck:  [64, 23, 87, 46, 5, 69, 28, 92, 51, 10, 74, 33, 97, 56, 15, 79, 38, 102, 61, 20, 84, 43, 2, 66, 25, 89, 48, 7, 71, 30, 9
4, 53, 12, 76, 35, 99, 58, 17, 81, 40, 104, 63, 22, 86, 45, 4, 68, 27, 91, 50, 9, 73, 32, 96, 55, 14, 78, 37, 101, 60, 19, 83, 42, 1, 6
5, 24, 88, 47, 6, 70, 29, 93, 52, 11, 75, 34, 98, 57, 16, 80, 39, 103, 62, 21, 85, 44, 3, 67, 26, 90, 49, 8, 72, 31, 95, 54, 13, 77, 36
, 100, 59, 18, 82, 41]
r-iterations:  0.013

7 . New deck:  [32, 64, 96, 23, 55, 87, 14, 46, 78, 5, 37, 69, 101, 28, 60, 92, 19, 51, 83, 10, 42, 74, 1, 33, 65, 97, 24, 56, 88, 15,
47, 79, 6, 38, 70, 102, 29, 61, 93, 20, 52, 84, 11, 43, 75, 2, 34, 66, 98, 25, 57, 89, 16, 48, 80, 7, 39, 71, 103, 30, 62, 94, 21, 53,
85, 12, 44, 76, 3, 35, 67, 99, 26, 58, 90, 17, 49, 81, 8, 40, 72, 104, 31, 63, 95, 22, 54, 86, 13, 45, 77, 4, 36, 68, 100, 27, 59, 91,
18, 50, 82, 9, 41, 73]
r-iterations:  0.016

8 . New deck:  [16, 32, 48, 64, 80, 96, 7, 23, 39, 55, 71, 87, 103, 14, 30, 46, 62, 78, 94, 5, 21, 37, 53, 69, 85, 101, 12, 28, 44, 60,
 76, 92, 3, 19, 35, 51, 67, 83, 99, 10, 26, 42, 58, 74, 90, 1, 17, 33, 49, 65, 81, 97, 8, 24, 40, 56, 72, 88, 104, 15, 31, 47, 63, 79,
95, 6, 22, 38, 54, 70, 86, 102, 13, 29, 45, 61, 77, 93, 4, 20, 36, 52, 68, 84, 100, 11, 27, 43, 59, 75, 91, 2, 18, 34, 50, 66, 82, 98,
9, 25, 41, 57, 73, 89]
r-iterations:  0.026

9 . New deck:  [8, 16, 24, 32, 40, 48, 56, 64, 72, 80, 88, 96, 104, 7, 15, 23, 31, 39, 47, 55, 63, 71, 79, 87, 95, 103, 6, 14, 22, 30,
38, 46, 54, 62, 70, 78, 86, 94, 102, 5, 13, 21, 29, 37, 45, 53, 61, 69, 77, 85, 93, 101, 4, 12, 20, 28, 36, 44, 52, 60, 68, 76, 84, 92,
 100, 3, 11, 19, 27, 35, 43, 51, 59, 67, 75, 83, 91, 99, 2, 10, 18, 26, 34, 42, 50, 58, 66, 74, 82, 90, 98, 1, 9, 17, 25, 33, 41, 49, 5
7, 65, 73, 81, 89, 97]
r-iterations:  0.05

10 . New deck:  [4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 52, 56, 60, 64, 68, 72, 76, 80, 84, 88, 92, 96, 100, 104, 3, 7, 11, 15,
19, 23, 27, 31, 35, 39, 43, 47, 51, 55, 59, 63, 67, 71, 75, 79, 83, 87, 91, 95, 99, 103, 2, 6, 10, 14, 18, 22, 26, 30, 34, 38, 42, 46,
50, 54, 58, 62, 66, 70, 74, 78, 82, 86, 90, 94, 98, 102, 1, 5, 9, 13, 17, 21, 25, 29, 33, 37, 41, 45, 49, 53, 57, 61, 65, 69, 73, 77, 8
1, 85, 89, 93, 97, 101]
r-iterations:  0.214

11 . New deck:  [2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62
, 64, 66, 68, 70, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92, 94, 96, 98, 100, 102, 104, 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25,
 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59, 61, 63, 65, 67, 69, 71, 73, 75, 77, 79, 81, 83, 85, 87, 89, 91, 93
, 95, 97, 99, 101, 103]
r-iterations:  0.486

12 . New deck:  [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32,
 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66
, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 1
00, 101, 102, 103, 104]
r-iterations:  1.0

13 . New deck:  [53, 1, 54, 2, 55, 3, 56, 4, 57, 5, 58, 6, 59, 7, 60, 8, 61, 9, 62, 10, 63, 11, 64, 12, 65, 13, 66, 14, 67, 15, 68, 16,
 69, 17, 70, 18, 71, 19, 72, 20, 73, 21, 74, 22, 75, 23, 76, 24, 77, 25, 78, 26, 79, 27, 80, 28, 81, 29, 82, 30, 83, 31, 84, 32, 85, 33
, 86, 34, 87, 35, 88, 36, 89, 37, 90, 38, 91, 39, 92, 40, 93, 41, 94, 42, 95, 43, 96, 44, 97, 45, 98, 46, 99, 47, 100, 48, 101, 49, 102
, 50, 103, 51, 104, 52]
r-iterations:  0.486

14 . New deck:  [79, 53, 27, 1, 80, 54, 28, 2, 81, 55, 29, 3, 82, 56, 30, 4, 83, 57, 31, 5, 84, 58, 32, 6, 85, 59, 33, 7, 86, 60, 34, 8
, 87, 61, 35, 9, 88, 62, 36, 10, 89, 63, 37, 11, 90, 64, 38, 12, 91, 65, 39, 13, 92, 66, 40, 14, 93, 67, 41, 15, 94, 68, 42, 16, 95, 69
, 43, 17, 96, 70, 44, 18, 97, 71, 45, 19, 98, 72, 46, 20, 99, 73, 47, 21, 100, 74, 48, 22, 101, 75, 49, 23, 102, 76, 50, 24, 103, 77, 5
1, 25, 104, 78, 52, 26]
r-iterations:  0.214

15 . New deck:  [92, 79, 66, 53, 40, 27, 14, 1, 93, 80, 67, 54, 41, 28, 15, 2, 94, 81, 68, 55, 42, 29, 16, 3, 95, 82, 69, 56, 43, 30, 1
7, 4, 96, 83, 70, 57, 44, 31, 18, 5, 97, 84, 71, 58, 45, 32, 19, 6, 98, 85, 72, 59, 46, 33, 20, 7, 99, 86, 73, 60, 47, 34, 21, 8, 100,
87, 74, 61, 48, 35, 22, 9, 101, 88, 75, 62, 49, 36, 23, 10, 102, 89, 76, 63, 50, 37, 24, 11, 103, 90, 77, 64, 51, 38, 25, 12, 104, 91,
78, 65, 52, 39, 26, 13]
r-iterations:  0.05
```



At the 12th iteration the cards are back in their original order. Therefore 15 runs is enough to return the original order.

THAT CONCLUDES OUR PROJECT REPORT