In [ ]:
```
import pandas as pd
import os
import math
```

In [ ]:
```
pwd = os.getcwd()
```

In [ ]:
```
dataset = pd.read_excel(pwd + '/Data - Exams.xlsx')
dataset
```

Out[ ]:

	gender	race/ethnicity	parental level of education	lunch	test preparation course	math score	reading score	writing score
0	female	group D	some college	standard	completed	59	70	78
1	male	group D	associate's degree	standard	none	96	93	87
2	female	group D	some college	free/reduced	none	57	76	77
3	male	group B	some college	free/reduced	none	70	70	63
4	female	group D	associate's degree	standard	none	83	85	86
...	...	...	...	...	...	...	...	...
995	male	group C	some college	standard	none	77	77	71
996	male	group C	some college	standard	none	80	66	66
997	female	group A	high school	standard	completed	67	86	86
998	male	group E	high school	standard	none	80	72	62
999	male	group D	high school	standard	none	58	47	45

1000 rows × 8 columns

In [ ]:
```
# 12. Create a dictionary of 5 students and their grades.
math_score_sample_dict = dataset.sample(5)['math score'].to_dict()
math_score_sample_dict
```

Out[ ]:
```
{546: 56, 309: 61, 650: 55, 413: 63, 845: 72}
```

In [ ]:
```
#18. Find the average of a list of numbers using for loops.
total_score = 0
```

```
count = 0

for score in dataset['math score']:
        total_score += score
        count += 1

average_math_score = total_score / count if count else 0

average_math_score
```

```
Out[ ]: 67.81
```

```
In [ ]: #41. Convert a dictionary into a list of tuples.
        math_score_sample_tuple = list(math_score_sample_dict.items())
        math_score_sample_tuple
```

```
Out[ ]: [(546, 56), (309, 61), (650, 55), (413, 63), (845, 72)]
```

```
In [ ]: # ---#20. From an arbitrary random list of numbers, only print the even numbers (On
        #|   (part of section 1 & 2 if you really think about it)
        # ---#21. Write a loop that prints the even numbers between 1 and 50.
        even_numbers = []
        count = 0
        for num in dataset['math score']:
                if num % 2 == 0:
                        even_numbers.append(num)
                        count += 1
                        #Only printing 31 to match the next loop
                        if count == 31:
                                break


        first_51_math_scores = []
        for score in dataset['math score'].iloc[:51]:
                if score % 2 == 0:
                        first_51_math_scores.append(score)


        linked_scores = min(len(even_numbers), len(first_51_math_scores))

        for i in range(linked_scores):
                print(f"{even_numbers[i]} | {first_51_math_scores[i]}")
```

```
96 | 96
70 | 70
68 | 68
82 | 82
46 | 46
80 | 80
74 | 74
76 | 76
70 | 70
56 | 56
80 | 80
66 | 66
70 | 70
74 | 74
58 | 58
70 | 70
80 | 80
90 | 90
80 | 80
68 | 68
32 | 32
82 | 82
68 | 68
74 | 74
46 | 46
76 | 76
86 | 86
52 | 52
96 | 96
80 | 80
80 | 80
```

In [ ]:

```
#22. Write a loop that prints the sum of numbers from 1 to 100
#    Altered the prompt a little. Instead it prints the sums from a LIST of 1 to 10
group_sums = dataset.groupby('race/ethnicity')['math score'].sum()

group_size = dataset.groupby('race/ethnicity').size()

for group, sum_score in group_sums.items():
        print(f"The sumn of math scores for {group}({group_size[group]}) is: {sum_s
```

```
The sumn of math scores for group A(79) is: 5190
The sumn of math scores for group B(198) is: 12686
The sumn of math scores for group C(323) is: 21160
The sumn of math scores for group D(257) is: 17702
The sumn of math scores for group E(143) is: 11072
```

In [ ]:

```
#24. Generate the first 10 Fibonacci numbers using a loop.??? (filler question)
fib1, fib2 = 0, 1
count = 0

print(f"Fibonnaci nummber {count + 1}: {fib1}")

while count < 9:
        print(f"Fibonnaci nummber {count + 2}: {fib2}")
        fib1, fib2 = fib2, fib1 + fib2
        count += 1
```

```
Fibonnaci nummber 1: 0
Fibonnaci nummber 2: 1
Fibonnaci nummber 3: 1
Fibonnaci nummber 4: 2
Fibonnaci nummber 5: 3
Fibonnaci nummber 6: 5
Fibonnaci nummber 7: 8
Fibonnaci nummber 8: 13
Fibonnaci nummber 9: 21
Fibonnaci nummber 10: 34
```

In [ ]:

```
# ---#23. Write a loop that prints the product of numbers from 1 to 20. (filler que
#|   (part of section 2)
# ---#23. Write a loop that prints the product of numbers from 1 to 20. (filler que
product = 1

for number in range(1, 21):
        product *= number

print(f"(1)The product of numbers from 1 to 20 is: {product}")

#-----------------------------------------------------------------------------------

product = math.factorial(20)

print(f"(2)The product of numbers from 1 to 20 is: {product}")
```

```
(1)The product of numbers from 1 to 20 is: 2432902008176640000
(2)The product of numbers from 1 to 20 is: 2432902008176640000
```

In [ ]:

```
#25. Check if a string is a palindrome. (lowkey another filler question)
def is_palindrome(string):
        string = string.replace(' ', '').lower()
        return string == string[::-1]

sammple_string = "racecar"
print(f"Is {sammple_string} a palindrome?: {is_palindrome(sammple_string)}")

#Would've linked this to the dataset better if there were student names.
for column_name in dataset.columns:
        print(f"Is {column_name} a palindrome?: {is_palindrome(column_name)}")
```

```
Is racecar a palindrome?: True
Is gender a palindrome?: False
Is race/ethnicity a palindrome?: False
Is parental level of education a palindrome?: False
Is lunch a palindrome?: False
Is test preparation course a palindrome?: False
Is math score a palindrome?: False
Is reading score a palindrome?: False
Is writing score a palindrome?: False
```

In [ ]:

```
#26. Count the vowels in a string. (I refuse to have another filler so I'm going to
def count_vowels(string):
        vowels = 'aeiou'
        count = 0
        string = string.lower()
        for char in string:
                if char in vowels:
                        count += 1
        return count

example_string = "This is a string with vowels"
print(f"The number of vowels in '{example_string}' is: {count_vowels(example_string

#If there's 3 vowels, then assume the entry is female. Otherwise assume it's male"
column_name = dataset.columns[0]
first_5_gender = dataset[column_name].iloc[:5]

def guess_gender(word):
        vowel_count = count_vowels(word)
        return 'female' if vowel_count >= 3 else 'male'

first_5_entries = dataset[column_name].iloc[:5]
gender_guesses = [guess_gender(entry) for entry in first_5_entries]

for original, guess in zip(first_5_entries, gender_guesses):
        print(f"Entry: {original} | Guess: {guess}")
```

```
The number of vowels in 'This is a string with vowels' is: 7
Entry: female | Guess: female
Entry: male | Guess: male
Entry: female | Guess: female
Entry: male | Guess: male
Entry: female | Guess: female
```

Exercise: Data Anomaly Detection

Objective: Write a Python function that identifies and returns any anomalies in a list of numbers. An anomaly is defined as a number that is more than two standard deviations away from the mean of the list.

Instructions:

Calculate the mean of the list. Calculate the standard deviation of the list. Iterate over the list to find any numbers that are more than two standard deviations away from the mean. Return a list of anomalies.

Example Definition:

def find_anomalies(data): mean = sum(data) / len(data) variance = sum([((x - mean) ** 2) for x in data]) / len(data) std_deviation = variance ** 0.5 return [x for x in data if abs(x - mean) > 2 * std_deviation]

Example Usage:

data = [10, 12, 12, 13, 12, 11, 14, 13, 15, 102, 12, 14, 13, 12, 10, 11, 14] anomalies = find_anomalies(data) print(f"Anomalies in the data: {anomalies}")

In my case I'd say to double check the reading scores for exceptionally high or low scores, but I guess that wouldn't make much sense if posted in the discussion without context.