In [2]:
```
# Chance Wiese
# 1
```

In [3]:
```
# 2
x = 11
print(f"x times 153 = {x*153}")
print(f"x squared = {x**2}")
print(f"cube root of x = {x**(1/3)}")
```

```
x times 153 = 1683
x squared = 121
cube root of x = 2.2239800905693152
```

In [4]:
```
# 3
import random
targetnum = random.randint(1,100)
print(f"Target Number: {targetnum}")
guess = None
attempts = 0
while guess != targetnum:
    guess = int(input("Guess a number between 1 and 100: "))  # Instructions said v
    attempts += 1

    if guess < targetnum:
        print("Your guess was too low. Try again!")
    elif guess > targetnum:
        print("Your guess was too high. Try again!")
    else:
        print(f"Congratulations, you did it! It took you {attempts} guesses.")
```

```
Target Number: 8
Your guess was too low. Try again!
Your guess was too high. Try again!
Congratulations, you did it! It took you 3 guesses.
```

In [5]:
```
# 4
value_list = []
while len(value_list) < 5:
    value = input("Input an integer or press q to quit: ")
    try:
        int_value = int(value)
        value_list.append(int_value)
        print(f"Sum of last 5 values in the list: {sum(value_list[-5:])}")
    except ValueError:
        if value.lower() == "q":
            print(f"Thank you for your time. Have a nice day! Final sum: {sum(value
            break
        else:
            print("Invalid response. Please try again.")
```

```
Sum of last 5 values in the list: 1
Sum of last 5 values in the list: 3
Sum of last 5 values in the list: 6
Sum of last 5 values in the list: 10
Sum of last 5 values in the list: 15
```

In [6]:
```
# 5
import pymssql

conn = pymssql.connect(
     host='stairway.usu.edu', # Server name goes in quotes.
     user='dunn2100', # Username goes in quotes.
     password='databases4ever', # Password goes in quotes
     database='2100corgirace') # Database to use goes in quotes

cursor = conn.cursor()

query = """
SELECT c.Corgname,
    CASE
        WHEN c.Breed = 'Pem' THEN 'Pembroke'
        ELSE 'Cardigan'
        END AS Breed,
    ROUND(AVG(o.RaceTime), 3) AS AvgRaceTime,
    COUNT(r.RaceID) AS NumRaces
FROM Corgi c
JOIN Outcome o ON c.CorgID = o.CorgID
JOIN Race r ON o.RaceID = r.RaceID
GROUP BY c.Corgname, c.Breed
ORDER BY AvgRaceTime ASC;
"""

cursor.execute(query)

# Execute the SQL query
cursor.execute(query)


### I used ChatGPT to help me fetch these and format them into rows. I wasn't sure
# Fetch all rows from the query result
rows = cursor.fetchall()

# Print the header
print(f"{'Corgi Name':<25}   {'Breed':<15}   {'Avg Race Time':<15}   {'Number of Ra

# Print each row with the data points separated by three spaces
for row in rows:
    print(f"{row[0]:<25}   {row[1]:<15}   {row[2]:<15.3f}   {row[3]}")

conn.close()
```

```
Corgi Name                  Breed           Avg Race Time     Number of Races
Bucco                       Pembroke        48.648            4
Fruitcup                    Cardigan        52.656            5
Golden Graham               Pembroke        54.203            4
Hywel the Woofer            Pembroke        54.837            6
Baroness von Fluffyshanks   Pembroke        55.150            4
Gareth Bale                 Pembroke        55.253            4
Stampy McDog                Pembroke        55.938            6
Stonewall                   Cardigan        55.950            4
Mary Queen of Corgs         Pembroke        56.875            4
Penny the Shedmonster       Pembroke        57.538            6
Holden Corgfield            Pembroke        58.125            6
Jedediah                    Pembroke        59.053            4
Stanley                     Pembroke        59.452            6
Snappy Ginger               Pembroke        59.454            7
Snarla June                 Pembroke        59.847            7
Smiley Shortdog             Cardigan        60.242            6
Drederick                   Pembroke        61.127            7
Prometheus                  Pembroke        61.177            7
Kraken                      Pembroke        61.260            7
Sweet Dee                   Pembroke        62.198            4
Cowboy Dan                  Pembroke        62.585            4
Oatmeal                     Pembroke        62.795            4
Foxy Stumptail              Pembroke        63.314            7
Lady Wigglebottom           Pembroke        63.562            5
Miss Juniper                Pembroke        64.242            6
```

In [7]:

```python
# 6
with open('lubbock2022.csv', 'r') as file:
    rows = file.readlines()


# Initialize variables
num_records = 0
total_age = 0
college_count = 0
hs_count = 0
total_experience = 0
outdoors_counts = {'high': 0, 'medium': 0, 'low': 0}
performance_counts = {'best': 0, 'good': 0, 'acceptable': 0, 'unacceptable': 0}
good_enuf_count = 0


# Set up the header list and indices
header = rows[0].strip().split(',')
id_idx = header.index('ID')
age_idx = header.index('Age')
college_idx = header.index('College')
hs_idx = header.index('HS')
experience_idx = header.index('Experience')
outdoors_idx = header.index('Outdoors')
performance_idx = header.index('Performance')
good_enuf_idx = header.index('GoodEnuf')


# Process each line (besides the header)
```

```
for row in rows[1:]:
    values = row.strip().split(',')   # Split each line and create a list of values

    if '' in values:   # Skip over any lines with blank values
        continue

    num_records += 1   # Count non-empty records

    total_age += int(values[age_idx])   # Add up total age to divide by number of p

    if values[college_idx].lower() == 'yes':  # Count up college graduates to divid
        college_count += 1

    if values[hs_idx].lower() == 'yes':    # Count up high school graduates to divi
        hs_count += 1

    total_experience += int(values[experience_idx])   # Add up total experience to

    outdoors = values[outdoors_idx].lower()
    if outdoors in outdoors_counts:
        outdoors_counts[outdoors] += 1

    performance = values[performance_idx].lower()
    if performance in performance_counts:
        performance_counts[performance] += 1

    if values[good_enuf_idx] == '1':
        good_enuf_count += 1


# Calculate results
average_age = total_age / num_records
average_experience_years = total_experience / num_records
college_percentage = (college_count / num_records) * 100
hs_percentage = (hs_count / num_records) * 100

outdoors_high_percentage = (outdoors_counts['high'] / num_records) * 100
outdoors_medium_percentage = (outdoors_counts['medium'] / num_records) * 100
outdoors_low_percentage = (outdoors_counts['low'] / num_records) * 100

performance_best_percentage = (performance_counts['best'] / num_records) * 100
performance_good_percentage = (performance_counts['good'] / num_records) * 100
performance_acceptable_percentage = (performance_counts['acceptable'] / num_records
performance_unacceptable_percentage = (performance_counts['unacceptable'] / num_rec

# Print results
print(f"Number of Records: {num_records}")
print(f"Average Age: {average_age:.1f}")
print(f"Percentage with College Education: {college_percentage:.1f}%")
print(f"Percentage with HS Education: {hs_percentage:.1f}%")
print(f"Average Years of Experience: {average_experience_years:.1f}")
print(f"Percentage with High Outdoors Experience: {outdoors_high_percentage:.1f}%")
print(f"Percentage with Medium Outdoors Experience: {outdoors_medium_percentage:.1f
print(f"Percentage with Low Outdoors Experience: {outdoors_low_percentage:.1f}%")
print(f"Percentage with Best Performance: {performance_best_percentage:.1f}%")
print(f"Percentage with Good Performance: {performance_good_percentage:.1f}%")
```

```
print(f"Percentage with Acceptable Performance: {performance_acceptable_percentage:
print(f"Percentage with Unacceptable Performance: {performance_unacceptable_percent
print(f"Number of GoodEnuf Salespeople: {good_enuf_count}")
```

```
Number of Records: 916
Average Age: 33.9
Percentage with College Education: 41.2%
Percentage with HS Education: 87.9%
Average Years of Experience: 6.3
Percentage with High Outdoors Experience: 17.8%
Percentage with Medium Outdoors Experience: 27.0%
Percentage with Low Outdoors Experience: 55.2%
Percentage with Best Performance: 3.9%
Percentage with Good Performance: 16.8%
Percentage with Acceptable Performance: 39.7%
Percentage with Unacceptable Performance: 39.4%
Number of GoodEnuf Salespeople: 555
```