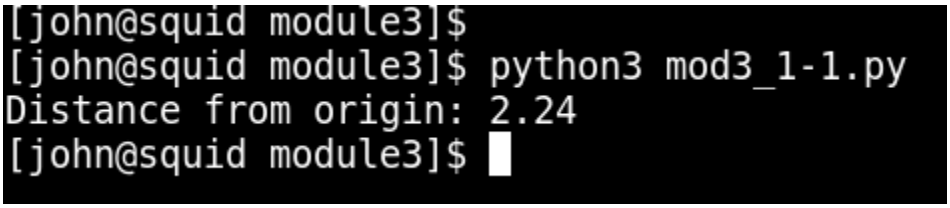
Module 3: Functions, OOP, modules, Errors & Exceptions

Case Study I

1. A Robot moves in a Plane starting from the origin point (0,0). The robot can move toward UP, DOWN, LEFT, RIGHT. The trace of Robot movement is as given following:
UP 5
DOWN 3
LEFT 3
RIGHT 2
The numbers after directions are steps. Write a program to compute the distance current position after sequence of movements. Hint: Use math module.

```
import math

# Initialize starting point
x, y = 0, 0

# Movements: direction and steps
movements = [("UP", 5), ("DOWN", 3), ("LEFT", 3), ("RIGHT", 2)]

# Process each movement
for direction, steps in movements:
    if direction == "UP":
        y += steps
    elif direction == "DOWN":
        y -= steps
    elif direction == "LEFT":
        x -= steps
    elif direction == "RIGHT":
        x += steps

# Calculate distance from origin
distance = math.sqrt(x**2 + y**2)

print(f"Distance from origin: {distance:.2f}")
```



2 Data of XYZ company is stored in sorted list. Write a program for searching specific data from that list. Hint: Use if/elif to deal with conditions.

```
def binary_search(sorted_list, target):
    """
    Perform binary search on a sorted list to find the target value.

    :param sorted_list: List of sorted elements
    :param target: The value to search for
    :return: The index of the target if found, otherwise -1
    """
    left, right = 0, len(sorted_list) - 1

    while left <= right:
        mid = (left + right) // 2
        mid_value = sorted_list[mid]

        if mid_value == target:
            return mid
        elif mid_value < target:
            left = mid + 1
        else:
            right = mid - 1

    return -1

# Example usage
sorted_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
target = 5

index = binary_search(sorted_list, target)
if index != -1:
    print(f"Target found at index: {index}")
else:
    print("Target not found in the list.")
```

```
[john@squid module3]$
[john@squid module3]$ python3 mod3_1-2.py
Target found at index: 4
[john@squid module3]$
```

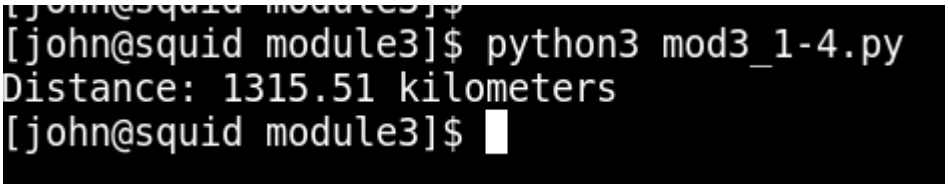
3. Weather forecasting organization wants to show is it day or night. So, write a program for such organization to find whether is it dark outside or not. Hint: Use time module.

```
import time

def is_it_dark():
    """
    Determine if it's dark based on the current local time.
    Considering dark hours from 7pm to 6am.
    """
    current_time = time.localtime()  # Get struct_time for local time
    hour = current_time.tm_hour  # Extract the hour from struct_time

    # Check if it's dark
    if hour >= 19 or hour < 6:
        return True
    else:
        return False

# Example usage
if is_it_dark():
    print("It's dark outside.")
else:
    print("It's not dark outside.")
```

```
[john@squid module3]$
[john@squid module3]$ python3 mod3_1-3.py
It's not dark outside.
[john@squid module3]$
```

4. Write a program to find distance between two locations when their latitude and longitudes are given.
Hint: Use math module.

```
import math

def calculate_distance(lat1, lon1, lat2, lon2):
    """
    Calculate the distance between two points on the earth (specified in decimal degrees).
    """
    # Convert decimal degrees to radians
    lat1, lon1, lat2, lon2 = map(math.radians, [lat1, lon1, lat2, lon2])

    # Haversine formula
    dlon = lon2 - lon1
    dlat = lat2 - lat1
    a = math.sin(dlat/2)**2 + math.cos(lat1) * math.cos(lat2) * math.sin(dlon/2)**2
    c = 2 * math.asin(math.sqrt(a))
    r = 6371  # Radius of earth in kilometers. Use 3956 for miles
    return c * r

# Example usage
lat1 = 52.2296756
lon1 = 21.0122287
lat2 = 41.8919300
lon2 = 12.5113300

distance = calculate_distance(lat1, lon1, lat2, lon2)
print(f"Distance: {distance:.2f} kilometers")
```



5. Design a software for bank system. There should be options like cash withdraw, cash credit and change password. According to user input, the software should provide required output.
Hint: Use if else statements and functions.

```
class BankAccount:
    def __init__(self, account_number, name, balance, password):
        self.account_number = account_number
        self.name = name
        self.balance = balance
        self.password = password

    def withdraw_cash(self, amount, password):
        if password == self.password:
            if amount <= self.balance:
                self.balance -= amount
                print(f"Withdrawal successful. Your new balance is: ${self.balance:.2f}")
            else:
                print("Insufficient funds.")
        else:
            print("Invalid password.")

    def credit_cash(self, amount):
        self.balance += amount
        print(f"Cash credited. Your new balance is: ${self.balance:.2f}")

    def change_password(self, old_password, new_password):
        if old_password == self.password:
            self.password = new_password
            print("Password changed successfully.")
        else:
            print("Invalid old password.")

def main():
    # Example account setup
    account = BankAccount("12345678", "John Doe", 1000.0, "password123")

    while True:
        print("\nBank System Menu:")
        print("1. Withdraw Cash")
        print("2. Credit Cash")
        print("3. Change Password")
        print("4. Exit")
        choice = input("Enter your choice: ")

        if choice == "1":
            amount = float(input("Enter amount to withdraw: "))
```

```
            password = input("Enter your password: ")
            account.withdraw_cash(amount, password)
        elif choice == "2":
            amount = float(input("Enter amount to credit: "))
            account.credit_cash(amount)
        elif choice == "3":
            old_password = input("Enter your old password: ")
            new_password = input("Enter your new password: ")
            account.change_password(old_password, new_password)
        elif choice == "4":
            print("Exiting the bank system.")
            break
        else:
            print("Invalid choice. Please choose again.")

if __name__ == "__main__":
    main()
```

```
[john@squid module3]$
[john@squid module3]$ python3 mod3_1-5.py

Bank System Menu:
1. Withdraw Cash
2. Credit Cash
3. Change Password
4. Exit
Enter your choice: 2
Enter amount to credit: 1000.0
Cash credited. Your new balance is: $2000.00

Bank System Menu:
1. Withdraw Cash
2. Credit Cash
3. Change Password
4. Exit
Enter your choice: 2
Enter amount to credit: 500.0
Cash credited. Your new balance is: $2500.00

Bank System Menu:
1. Withdraw Cash
2. Credit Cash
3. Change Password
4. Exit
Enter your choice: 1
Enter amount to withdraw: 1500.0
Enter your password: password123
Withdrawal successful. Your new balance is: $1000.00

Bank System Menu:
1. Withdraw Cash
2. Credit Cash
3. Change Password
4. Exit
Enter your choice: 4
Exiting the bank system.
[john@squid module3]$
```

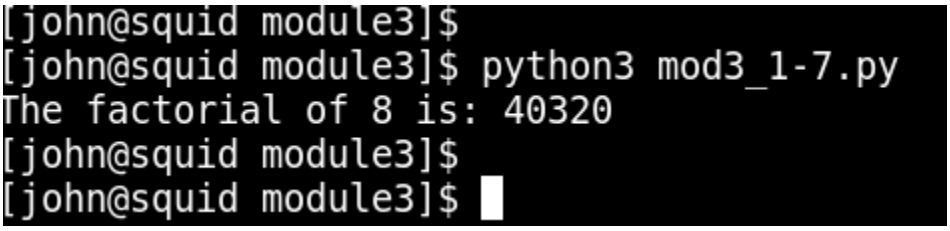
6. Write a program which will find all such numbers which are divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included). The numbers obtained should be printed in a comma-separated sequence on a single line.

```
# Find numbers divisible by 7 but not a multiple of 5, between 2000 and 3200
numbers = [str(number) for number in range(2000, 3201) if number % 7 == 0 and number % 5 != 0]

# Print the numbers in a comma-separated sequence
print(",".join(numbers))
```

```
[john@squid module3]$ python3 mod3_1-6.py
2002,2009,2016,2023,2037,2044,2051,2058,2072,2079,2086,2093,2107,2114,2121,2128,2142,
2149,2156,2163,2177,2184,2191,2198,2212,2219,2226,2233,2247,2254,2261,2268,2282,2289,
2296,2303,2317,2324,2331,2338,2352,2359,2366,2373,2387,2394,2401,2408,2422,2429,2436,
2443,2457,2464,2471,2478,2492,2499,2506,2513,2527,2534,2541,2548,2562,2569,2576,2583,
2597,2604,2611,2618,2632,2639,2646,2653,2667,2674,2681,2688,2702,2709,2716,2723,2737,
2744,2751,2758,2772,2779,2786,2793,2807,2814,2821,2828,2842,2849,2856,2863,2877,2884,
2891,2898,2912,2919,2926,2933,2947,2954,2961,2968,2982,2989,2996,3003,3017,3024,3031,
3038,3052,3059,3066,3073,3087,3094,3101,3108,3122,3129,3136,3143,3157,3164,3171,3178,
3192,3199
[john@squid module3]$
```

7. Write a program which can compute the factorial of a given numbers. Use recursion to find it. Hint: Suppose the following input is supplied to the program:
8
Then, the output should be:
40320

```
def factorial(n):
    """
    Compute the factorial of a given number using recursion.
    """
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)

# Example usage
n = 8
print(f"The factorial of {n} is: {factorial(n)}")
```



8. Write a program that calculates and prints the value according to the given formula:
Q = Square root of [(2 * C * D)/H]
Following are the fixed values of C and H: C is 50. H is 30.
D is the variable whose values should be input to your program in a comma-separated sequence.
Example:
Let us assume the following comma separated input sequence is given to the
program:
100,150,180
The output of the program should be:
18,22,24

```
import math

def calculate_q(d_values):
    """
    Calculate and print the value of Q for each given D, using the formula:
    Q = Square root of [(2 * C * D)/H]
    """
    C = 50
    H = 30
    results = []
    for D in d_values:
        Q = math.sqrt((2 * C * D) / H)
        results.append(int(Q))
    return results

# Example usage
d_values = [100, 150, 180]
results = calculate_q(d_values)
print(",".join(map(str, results)))
```

```
[john@squid module3]$
[john@squid module3]$ python3 mod3_1-8.py
18,22,24
[john@squid module3]$
```

9. Write a program which takes 2 digits, X,Y as input and generates a 2-dimensional array. The element value in the i-th row and j-th column of the array should be i*j.
Note: i=0,1.., X-1; j=0,1,¡-Y-1.
Example:
Suppose the following inputs are given to the program:
3,5
Then, the output of the program should be:
[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]

```
def generate_2d_array(x, y):
    """
    Generate a 2D array where the element value in the i-th row and j-th column is i*j.
    """
    return [[i*j for j in range(y)] for i in range(x)]

# Example usage
x, y = 3, 5
array = generate_2d_array(x, y)
print(array)
```

```
[john@squid module3]$ python3 mod3_1-9.py
[[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]
[john@squid module3]$
```

10. Write a program that accepts a comma separated sequence of words as input and prints the words in a comma-separated sequence after sorting them alphabetically.
Suppose the following input is supplied to the program:
without,hello,bag,world
Then, the output should be:
bag,hello,without,world

```
def sort_words(input_string):
    """
    Sort a comma-separated sequence of words alphabetically.
    """
    words = input_string.split(',')
    words.sort()
    return ','.join(words)

# Example usage
input_string = "without,hello,bag,world"
sorted_string = sort_words(input_string)
print(sorted_string)
```

```
[john@squid module3]$
[john@squid module3]$ python3 mod3_1-10.py
bag,hello,without,world
[john@squid module3]$
```

11. Write a program that accepts sequence of lines as input and prints the lines after making all characters in the sentence capitalized.
Suppose the following input is supplied to the program:
Hello world
Practice makes perfect
Then, the output should be:
HELLO WORLD
PRACTICE MAKES PERFECT

```
def capitalize_lines():
    """
    Accepts sequence of lines as input and prints the lines after capitalizing all characters.
    """
    lines = []
    while True:
        line = input("Enter line (or leave blank to finish): ")
        if not line:
            break
        lines.append(line.upper())
```

```
    for line in lines:
        print(line)

# Example usage
capitalize_lines()
```

```
[john@squid module3]$
[john@squid module3]$ python3 mod3_1-11.py
Enter line (or leave blank to finish): Hello world
Enter line (or leave blank to finish): Practice makes perfect
Enter line (or leave blank to finish):
HELLO WORLD
PRACTICE MAKES PERFECT
[john@squid module3]$
```

12. Write a program that accepts a sequence of whitespace separated words as input and prints the words after removing all duplicate words and sorting them alphanumerically.
Suppose the following input is supplied to the program:
hello world and practice makes perfect and hello world again
Then, the output should be:
again and hello makes perfect practice world

```
def remove_duplicates_sort(input_string):
    """
    Remove duplicate words and sort the remaining words alphanumerically.
    """
    words = input_string.split()
    unique_words = sorted(set(words))
    return ' '.join(unique_words)

# Example usage
input_string = "hello world and practice makes perfect and hello world again"
result = remove_duplicates_sort(input_string)
print(result)
```

```
[john@squid module3]$
[john@squid module3]$ python3 mod3_1-12.py
again and hello makes perfect practice world
[john@squid module3]$
```

13. Write a program which accepts a sequence of comma separated 4 digit binary numbers as its input and then check whether they are divisible by 5 or not. The numbers that are divisible by 5 are to be printed in a comma separated sequence.
Example:
0100,0011,1010,1001
Then the output should be:
1010

```
def filter_divisible_by_5(binary_numbers):
    """
    Check which binary numbers are divisible by 5 and print them in a comma-separated sequence.
    """
    divisible_by_5 = [num for num in binary_numbers.split(',') if int(num, 2) % 5 == 0]
    return ','.join(divisible_by_5)

# Example usage
binary_numbers = "0100,0011,1010,1001"
result = filter_divisible_by_5(binary_numbers)
print(result)
```

```
[john@squid module3]$
[john@squid module3]$ python3 mod3_1-13.py
1010
[john@squid module3]$
```

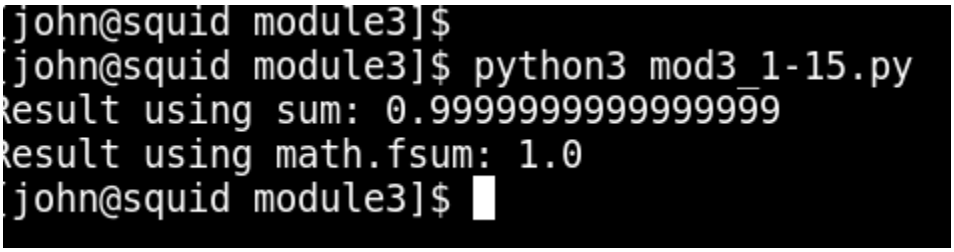
14. Write a program that accepts a sentence and calculate the number of upper case letters and lower case letters.
Suppose the following input is supplied to the program:
Hello world!
Then, the output should be:
UPPER CASE 1
LOWER CASE 9

```
def count_case(sentence):
    """
    Calculate the number of upper case and lower case letters in a sentence.
    """
    upper_case = sum(1 for char in sentence if char.isupper())
    lower_case = sum(1 for char in sentence if char.islower())
    return f"UPPER CASE {upper_case}\nLOWER CASE {lower_case}"

# Example usage
sentence = "Hello world!"
result = count_case(sentence)
print(result)
```

```
[john@squid module3]$ python3 mod3_1-14.py
UPPER CASE 1
LOWER CASE 9
[john@squid module3]$
```

15.Give example of fsum and sum function of math library .

```
import math

# Example list of floating point numbers
numbers = [0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1]

# Using sum
sum_result = sum(numbers)
print(f"Result using sum: {sum_result}")

# Using math.fsum for better precision with floating point numbers
fsum_result = math.fsum(numbers)
print(f"Result using math.fsum: {fsum_result}")
```


john@squid module3]$
john@squid module3]$ python3 mod3_1-15.py
esult using sum: 0.9999999999999999
esult using math.fsum: 1.0
john@squid module3]$