# Artificial Intelligence

# Assignment 3

***Topic :***

***Python Challenges at hackerrank***


Submitted by: Seemab Khan

PhD RIME (432826)

Submitted to: Dr. Yasar Ayaz

Department of Robotics and Intelligent Machine Engineering

SMME-NUST H-12 Islamabad

Date: Jan 1st ,2024

# 1. Write a function

```
def is_leap(year):
    # If the year is evenly divisible by 4
    if year % 4 == 0:
        # If the year is evenly divisible by 100
        if year % 100 == 0:
            # If the year is evenly divisible by 400, it is a leap year
            if year % 400 == 0:
                return True
            # If the year is evenly divisible by 100 but not by 400, it is
not a leap year
            else:
                return False
        # If the year is evenly divisible by 4 but not by 100, it is a
leap year
        else:
            return True
    # If the year is not evenly divisible by 4, it is not a leap year
    else:
        return False

# Example usage:
year = int(input("Enter a year: "))
result = is_leap(year)
print(result)
```

output:

```
Enter a year: 2025
False
```

# 2. Minion Game

```
def minion_game(string):
    string_length = len(string)
    vowels = "AEIOU"

    # Initialize scores
    stuart_score = 0  # For words starting with consonants
    kevin_score = 0  # For words starting with vowels

    for i in range(string_length):
        if string[i] in vowels:
```

```
                # Kevin's turn
                kevin_score += string_length - i
            else:
                # Stuart's turn
                stuart_score += string_length - i

        # Determine the winner and print the result
        if stuart_score > kevin_score:
            print(f"Stuart {stuart_score}")
        elif kevin_score > stuart_score:
            print(f"Kevin {kevin_score}")
        else:
            print("Draw")

    # Example usage
    if __name__ == "__main__":
        input_string = input("Enter a string: ").upper()
        minion_game(input_string)

```

output

```
Enter a string: an
Kevin 2
```

# 3. Time Delta

```
from datetime import datetime, timedelta

def time_difference_in_seconds(time1, time2):
    date_format = '%a %d %b %Y %H:%M:%S %z'
    time1 = datetime.strptime(time1, date_format)
    time2 = datetime.strptime(time2, date_format)

    time_difference = abs(time1 - time2).total_seconds()
    return int(time_difference)

# Example usage:
time1 = input("Enter the first timestamp: ")
time2 = input("Enter the second timestamp: ")

result = time_difference_in_seconds(time1, time2)
print(f"The absolute difference in seconds is: {result} seconds")
```

output:

```
Enter the first timestamp: Sun 10 May 2015 13:54:36 -0700
Enter the second timestamp: Sun 10 May 2015 13:54:36 -0000
The absolute difference in seconds is: 25200 seconds
```

# 4. Find Angle MBC

```python
import math

def find_angle(AB, BC):
    # Calculate the angle in radians
    theta_rad = math.atan2(AB, BC)

    # Convert the angle to degrees and round to the nearest integer
    theta_deg = round(math.degrees(theta_rad))

    return theta_deg

# Input lengths of sides AB and BC
AB = float(input("Enter the length of side AB: "))
BC = float(input("Enter the length of side BC: "))

# Calculate and print the angle
angle = find_angle(AB, BC)
print(f"The angle θ is approximately {angle} degrees.")
```

outout:

```
Enter the length of side AB: 45
Enter the length of side BC: 89
The angle θ is approximately 27 degrees.
```

# 5. Word order

```
5
apple
orange
banana
kivi
peach
1 1 1 1 1
```

output:

# 6. String Compression

```
def compress_string(s):
    compressed_string = ""

    for key, group in groupby(s):
        compressed_string += key + str(len(list(group)))

    return compressed_string

# Input the string
s = input()

# Output the modified string
result = compress_string(s)
print(result)
```

# Output:

```
1233445643
1121324251614131
```

# 7. Company logo

```
from collections import Counter

def top_three_characters(company_name):
    # Count the occurrences of each character
    char_count = Counter(company_name)

    # Get the three most common characters
    top_three = char_count.most_common(3)

    # Sort the result based on occurrence count and then alphabetically
    sorted_result = sorted(top_three, key=lambda x: (-x[1], x[0]))

    # Print the result
    for char, count in sorted_result:
        print(f"{char} {count}")

# Input the company name
company_name = input("Enter the company name: ")

# Output the top three characters and their occurrence count
top_three_characters(company_name)
```

output:

```
Enter the company name: Apple
p 2
A 1
l 1
```

# 8. Pilling up

```
def can_stack_cubes(test_cases, cubes):
    left = 0
    right = len(cubes) - 1
    prev_cube = float('inf')

    while left <= right:
        if cubes[left] >= cubes[right] and cubes[left] <= prev_cube:
            prev_cube = cubes[left]
            left += 1
        elif cubes[right] > cubes[left] and cubes[right] <= prev_cube:
            prev_cube = cubes[right]
            right -= 1
        else:
            return "No"

    return "Yes"

def main():
    # Input the number of test cases
    t = int(input())

    for _ in range(t):
        # Input the number of cubes
        n = int(input())

        # Input the side lengths of cubes
        cubes = list(map(int, input().split()))

        # Output Yes or No based on whether it's possible to stack the
cubes
        result = can_stack_cubes(t, cubes)
        print(result)

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

output:
```
4
2
7
Yes
```

# 9. Iteratools and iterations

```
from itertools import combinations

def probability_of_letter_a(n, letters, k):
    total_combinations = list(combinations(range(n), k))
    favorable_combinations = [comb for comb in total_combinations if 'a'
in [letters[i] for i in comb]]

    probability = len(favorable_combinations) / len(total_combinations)

    return round(probability, 3)

# Input the length of the list
n = int(input())

# Input the elements of the list
letters = input().split()

# Input the number of indices to be selected
k = int(input())

# Output the probability that at least one of the selected indices
contains the letter 'a'
result = probability_of_letter_a(n, letters, k)
print(result)
```

output:
```
6
a s d d f f
3
0.5
```