Question-1

Find a pair with the given sum in an array

Given an unsorted integer array, find a pair with the given sum in it.

For example

Input: nums = [8, 7, 2, 5, 3, 1]target = 10 Output: Pair found (8, 2)orPair found (7, 3)

Ans)

```
def find_pair(nums, target):
    seen = set()
    for num in nums:
        complement = target - num
        if complement in seen:
            print("Pair found:", (complement, num))
            return True
        seen.add(num)
    print("Pair not found.")
    return False
nums = [8, 7, 2, 5, 3, 1]
```

target = 10

find_pair(nums, target)

Question-2

Given an integer array, replace each element with the product of every other element without using the division operator.

For example,

Input: { 1, 2, 3, 4, 5 }Output: { 120, 60, 40, 30, 24 } Input: { 5, 3, 4, 2, 6, 8 }Output: { 1152, 1920, 1440, 2880, 960, 720 }

Ans) def product_except_self(nums):

```
    n = len(nums)
    left_products = [1] * n
    right_products = [1] * n
    left_product = 1
    for i in range(1, n):
        left_product *= nums[i - 1]
        left_products[i] = left_product
```

```
    right_product = 1
    for i in range(n - 2, -1, -1):
        right_product *= nums[i + 1]
        right_products[i] = right_product
    result = [left_products[i] * right_products[i] for i in range(n)]

    return result
nums1 = [1, 2, 3, 4, 5]
nums2 = [5, 3, 4, 2, 6, 8]

output1 = product_except_self(nums1)
output2 = product_except_self(nums2)

print("Output for nums1:", output1)
print("Output for nums2:", output2)
```

Question-3

Maximum Sum Circular Subarray

Given a circular integer array, find a subarray with the largest sum in it.

For example :Input: {2, 1, -5, 4, -3, 1, -3, 4, -1} Output: Subarray with the largest sum is {4, -1, 2, 1} with sum 6.

Ans) def max_subarray_sum_circular(nums):

```
    def kadane(arr):
        max_sum = float('-inf')
        current_sum = 0
        for num in arr:
            current_sum = max(num, current_sum + num)
            max_sum = max(max_sum, current_sum)
        return max_sum

    total_sum = sum(nums)
    max_kadane = kadane(nums)

    # If all numbers are negative, max subarray sum is the maximum element itself
    if max_kadane < 0:
```

```
        return max_kadane

    # Calculate max subarray sum when wrapping around the circular array
    max_wrap = total_sum - min_kadane(nums)
    return max(max_kadane, max_wrap)

def min_kadane(arr):
    min_sum = float('inf')
    current_sum = 0
    for num in arr:
        current_sum = min(num, current_sum + num)
        min_sum = min(min_sum, current_sum)
    return min_sum
nums = [2, 1, -5, 4, -3, 1, -3, 4, -1]
result = max_subarray_sum_circular(nums)
print("Subarray with the largest sum is:", result)
```

Question-4:

Find the maximum difference between two array elements that satisfies the given constraints

Given an integer array, find the maximum difference between two elements in it such that the smaller element appears before the larger element.

For example:Input: { 2, 7, 9, 5, 1, 3, 5 } Output: The maximum difference is 7. The pair is (2, 9)

Ans)
```
def max_difference(nums):
    if len(nums) < 2:
        return "Not enough elements in the array"

    min_element = nums[0]
    max_difference = nums[1] - min_element

    for num in nums[1:]:
        if num - min_element > max_difference:
            max_difference = num - min_element

        if num < min_element:
```

```
            min_element = num

    return max_difference
nums = [2, 7, 9, 5, 1, 3, 5]
result = max_difference(nums)
print("The maximum difference is:", result)
```

Question:5

Given an array of integers of size N, the task is to find the first non-repeating element in this array.

Examples:

Input: {-1, 2, -1, 3, 0}

Output: 2

Explanation: The first number that does not repeat is : 2

Input: {9, 4, 9, 6, 7, 4}

Output: 6

Ans) def first_non_repeating_element(nums):

```
    element_count = {}

    for num in nums:
        if num in element_count:
            element_count[num] += 1
        else:
            element_count[num] = 1

    for num in nums:
        if element_count[num] == 1:
            return num

    return None  # If no non-repeating element is found

# Example usage
nums1 = [-1, 2, -1, 3, 0]
nums2 = [9, 4, 9, 6, 7, 4]

result1 = first_non_repeating_element(nums1)
result2 = first_non_repeating_element(nums2)
```

print("First non-repeating element in nums1:", result1)

print("First non-repeating element in nums2:", result2)

Question:6

Minimize the maximum difference between the heights

Given the heights of N towers and a value of K, Either increase or decrease the height of every tower by K (only once) where K > 0. After modifications, the task is to minimize the difference between the heights of the longest and the shortest tower and output its difference.

Examples:

Input: arr[] = {1, 15, 10}, k = 6

Output: Maximum difference is 5.

Explanation: Change 1 to 7, 15 to 9 and 10 to 4. Maximum difference is 5 (between 4 and 9). We can’t get a lower difference.

Input: arr[] = {1, 5, 15, 10}, k = 3

Output: Maximum difference is 8, arr[] = {4, 8, 12, 7}

Ans) def minimize_max_difference(arr, k):

    n = len(arr)

    # Step 1: Sort the array

    arr.sort()

    # Step 2: Initialize the result as the difference between the first and last elements

    result = arr[n - 1] - arr[0]

    # Step 3: For each tower, try increasing and decreasing its height by K

    for i in range(1, n - 1):

        small = min(arr[0] + k, arr[i] - k)

        large = max(arr[i] + k, arr[n - 1] - k)

        result = min(result, large - small)

    return result

Example usage

```
arr1, k1 = [1, 15, 10], 6
arr2, k2 = [1, 5, 15, 10], 3

result1 = minimize_max_difference(arr1, k1)
result2 = minimize_max_difference(arr2, k2)

print("Maximum difference for arr1:", result1)
print("Maximum difference for arr2:", result2)
```