Assignment-4

Sol 1-

```python
def find_common_elements(arr1, arr2, arr3):
    common_elements = []
    i, j, k = 0, 0, 0

    while i < len(arr1) and j < len(arr2) and k < len(arr3):
        if arr1[i] == arr2[j] == arr3[k]:
            common_elements.append(arr1[i])
            i += 1
            j += 1
            k += 1
        elif arr1[i] < arr2[j]:
            i += 1
        elif arr2[j] < arr3[k]:
            j += 1
        else:
            k += 1

    return common_elements


arr1 = [1, 2, 3, 4, 5]
arr2 = [1, 2, 5, 7, 9]
arr3 = [1, 3, 4, 5, 8]

result = find_common_elements(arr1, arr2, arr3)
print(result)
```

Sol 2-

```python
def find_missing_elements(nums1, nums2):
    set1 = set(nums1)
    set2 = set(nums2)

    missing_nums1 = list(set1 - set2)
    missing_nums2 = list(set2 - set1)

    return [missing_nums1, missing_nums2]

def new_func():
    new_var = nums1 = [1, 2, 3]
        nums2 = [2, 4, 6]
        result = find_missing_elements(nums1, nums2)
        print(result)
    new_var

new_func()
```

Sol 3-

```
def transpose(matrix):
    rows = len(matrix)
    cols = len(matrix[0])

    transpose_matrix = [[0] * rows for _ in range(cols)]

    for i in range(rows):
        for j in range(cols):
            transpose_matrix[j][i] = matrix[i][j]

    return transpose_matrix

matrix = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

result = transpose(matrix)
print(result)
```

Ans 4-

```
def array_pair_sum(nums):
    nums.sort()  # Sort the array in ascending order
    n = len(nums)
    max_sum = 0

    for i in range(0, n, 2):
        max_sum += nums[i]

    return max_sum

nums = [1, 4, 3, 2]

result = array_pair_sum(nums)
print(result)
```

Ans 5-

```
def arrange_coins(n):
    k = 0
    while n >= 0:
        k += 1
        n -= k

    return k - 1

n = 5

result = arrange_coins(n)
print(result)
```

Ans 6-

```
def sorted_squares(nums):
    n = len(nums)
    result = [0] * n
    left = 0
    right = n - 1
    index = n - 1

    while left <= right:
        if abs(nums[left]) > abs(nums[right]):
            result[index] = nums[left] ** 2
            left += 1
        else:
            result[index] = nums[right] ** 2
            right -= 1
        index -= 1

    return result

nums = [-4, -1, 0, 3, 10]

result = sorted_squares(nums)
print(result)
```

Ans 7-

```
def max_count(m, n, ops):
    if not ops:
        return m * n

    min_a = min(op[0] for op in ops)
    min_b = min(op[1] for op in ops)

    return min_a * min_b
```

```
m = 3
n = 3
ops = [[2, 2], [3, 3]]

result = max_count(m, n, ops)
print(result)
```

Ans 8-

```
def shuffle(nums, n):
    result = []
    for i in range(n):
        result.append(nums[i])
        result.append(nums[i+n])
    return result

nums = [2, 5, 1, 3, 4, 7]
n = 3

result = shuffle(nums, n)
print(result)
```