# Islamic University of Technology (IUT)

Organization of Islamic Cooperation (OIC)

Department of Electrical and Electronic Engineering (EEE)

# EEE 4416: Simulation Lab

# Lab – 05 Assignment

## Exercise - 01

**Problem statement:** Pentatope number

Pentatope numbers (also known as 4-simplex numbers) are figurate numbers that represent a 4-dimensional analog of a tetrahedron. Write a function called 'pentatope_number' that returns the n-th pentatope number.

$$\text{n-th pentatope number} = \frac{n(n+1)(n+2)(n+3)}{24}$$

**Test case – 01**

- Input: 1
- Output: 1

**Test case – 02**

- Input: 4
- Output: 35

**Test case – 03**

- Input: 11
- Output: 1001

**Test case – 04**

- Input: 50
- Output: 292825

**Test case – 05**

- Input: 1:5
- Output: [1, 5, 15, 35, 70]

*Asif Newaz, Aseer Imad Keats, Salman Hossain*
*Lecturer, Department of EEE, IUT*

# Exercise – 02

**Problem statement:** Least common multiples

Given an array of integers, find the least common multiple (lcm) of the numbers. Write a function named 'lcm_array' that takes an array as the input and returns the lcm as output.

**Test Case - 01**

- Input: [6, 7]
- Output: 42

**Test case – 02**

- Input: [4, 12]
- Output: 12

**Test case – 03**

- Input: [4, 12, 25]
- Output: 300

**Test case – 04**

- Input: [ 4, 12, 25, 2, 3, 14, 52, 23, 45]
- Output: 1883700

**Hint:** MATLAB offers a built-in function called ***lcm (a,b)*** which returns the lcm of integers a and b. However, the function has limitations. It cannot provide the lcm of more than 2 integers or arrays. You need to find a way to overcome this limitation.

- ✓ There is a way in MATLAB to get the lcm of an array. You need to specify specify the array as a symbolic vector. This is part of the symbolic math toolbox. However, for this problem, you are not allowed to solve the problem this way.

- ✓ Try to do the same for the Greatest common divisor (GCD). A similar built-in function for this purpose is ***gcd(a,b)***.

*Asif Newaz, Aseer Imad Keats, Salman Hossain*
*Lecturer, Department of EEE, IUT*

# Exercise – 03

**Problem statement:** Perfect number

A perfect number is a positive integer that is equal to the sum of its proper positive divisors, excluding itself. For instance, 6 is a perfect number. Its divisors are 1, 2, and 3 – the sum of which is 6.

If the sum of its proper divisors is greater than the number itself, it is called an abundant number. For instance, 12 is an abundant number. Its divisors are 1, 2, 3, 4, and 6 – the sum of which is 16.

The number is called a deficient number if the sum of divisors < number.

Write a function called 'number_category' that takes an integer as input and classifies it as one of the 3 categories.

**Test case – 01**

- Input: 8
- Output: 'Deficient'

**Test case – 02**

- Input: 12
- Output: 'Abundant'

**Test case – 03**

- Input: 8128
- Output: 'Perfect'

**Test case – 04**

- Input: 198
- Output: 'Abundant'

**Test case – 05**

- Input: 11,088
- Output: 'Abundant'

**Test case – 06**

- Input: 33,550,336
- Output: 'Perfect'

- ✓ Perfect numbers are extremely rare. 33550336 is actually the $5^{th}$ perfect number.
- ✓ You have already created your own function for figuring out the proper divisors of a number in a previous lab. You can just simply use it here same as a built-in function. This way, your created functions will come in very handy while solving many different problems.
- ✓ Make sure your user-defined function for proper divisors is in the same directory.

*Asif Newaz, Aseer Imad Keats, Salman Hossain*
*Lecturer, Department of EEE, IUT*

# Exercise – 04

**Problem statement:**

Write a function called 'flipped_diag' that takes two inputs. The 1st input is a matrix and the 2nd input is a string.

- If the string is 'main', flip the main diagonal of the matrix.
- If the string is 'anti', flip the anti-diagonal of the matrix.
- If the string is anything else, return the same matrix.

**Test case – 01**

- Input: magic (5), 'main'
- Output:

```
 9    24     1     8    15

23    21     7    14    16

 4     6    13    20    22

10    12    19     5     3

11    18    25     2    17
```

**Test case – 02**

- Input - 1: [1 5
  6 3]
- Input - 2: 'anti'
- Output: [1 6
  5 3]

**Test case – 03**

- Input - 1: [1 5
  6 3]
- Input - 2: 'aaaa'
- Output: [1 5
  6 3]

*Asif Newaz, Aseer Imad Keats, Salman Hossain*
*Lecturer, Department of EEE, IUT*

# Exercise - 05

**Problem Statement:**

Write a user-defined function named 'blockStats' that accepts a row vector containing 100 numeric elements. It should compute the following for each consecutive 5-element segment of the input vector:

- The mean of each segment
- The maximum value in each segment
- The minimum value in each segment

The function must verify that the input is a 1×100 row vector. If the input is not a row vector of length 100, it should display an appropriate error message and terminate execution. The function must **return three separate row vectors** representing the **mean**, **maximum**, and **minimum** values of each segment, respectively.

*Your implementation must use **vectorized operations only**. Do not use any loops (for, while, etc.) in your code.*

**Test case**

- Input: randi(208,1,100)
- Output:

```
Mean_value = 1×20
[139.0000 116.0000 74.6000 119.6000 110.8000 92.6000 76.6000 49.4000 122.6000
83.0000 108.8000 144.2000 102.2000 87.8000 111.8000 124.0000 138.0000 114.6000
80.2000 75.0000]

Max_value = 1×20
[177 178 113 195 125 159 174 103 150 146 187 198 177 137 198 147 178 178 189
197]

Min_value = 1×20
[101 56 29 2 84 28 32 15 72 18 63 56 1 17 7 92 71 23 4 20]
```

***Hints:** You may find functions like 'isrow', 'error', 'reshape', 'mean', 'max', and 'min' helpful.*

*Asif Newaz, Aseer Imad Keats, Salman Hossain*
*Lecturer, Department of EEE, IUT*

# Exercise - 06

**Problem Statement:**

Write a user-defined function named ‘vectorOps’ that accepts two row vectors of the same length and a string input specifying the operation to perform. It should return a new vector containing the result of the specified operation applied element-wise. The function should support the following operations:

- 'add' – Element-wise addition
- 'subtract' – Element-wise subtraction
- 'multiply' – Element-wise multiplication
- 'divide' – Element-wise division (handle divide-by-zero errors with a warning and return NaN)

The function must verify whether:

- Both input vectors are row vectors of the same length
- Operation is a valid string from the supported list

If the input is invalid, the function should display an appropriate error message and stop execution.

**Test case – 01**

- Input - 1: [10, 20, 30, 40]
- Input - 2: [2, 5, 0, 10]
- Input - 3: 'add'
- Output: [12  25  30  50]

**Test case – 02**

- Input - 1: [10, 20, 30, 40]
- Input - 2: [2, 5, 0, 10]
- Input - 3: 'divide'
- Output:

  Warning: Division by zero encountered. The result will contain NaN.

  [5  4  NaN  4]

***Hints:*** *You may find functions like ‘strcmp’, ‘error’, and ‘warning’ helpful.*

*Asif Newaz, Aseer Imad Keats, Salman Hossain*
*Lecturer, Department of EEE, IUT*

# Exercise - 07

**Problem Statement:**
You are given wind speed data for three locations, each measured over five consecutive hours. At each location, three different wind turbines operate, each with distinct specifications. Your task is to compute the total power output for each location by considering the hourly wind speeds and the individual turbine properties.

The power generated by a wind turbine can be estimated using the following equation:

$$P = 0.5 * \rho * A * Cp * v^3$$

Where:

- P = Power output in Watts
- ρ (rho) = Air density
- A = Swept area of the turbine blades ($m^2$)
- Cp = Power coefficient
- v = Wind speed in m/s

In this problem, three locations each have different hourly wind speed data and three different turbines with various rotor sizes (area), power coefficient Cp, and rated power capacity (in Watts). For each turbine at each location, calculate the power output for each day, and then compute the total power output from all turbines across all locations.

Write a function called 'total_wind_power' that takes no input, and inside it:

1. Uses nested for loops to simulate 3 locations, each with 3 turbines, and 5 hours of wind speed.

2. Computes the output power separately for each location using the above formula.

3. If the calculated power is greater than the turbine's rated power**,** use the rated power instead (this means the turbine can't produce more than its limit).

4. Returns the total combined power output from all turbines across all locations and all days.

Use the following data:

- Air density ρ (rho) = 1.225
- Turbine parameters
  **areas** = [30, 50, 70] ($m^2$)
  **Cps** = [0.35, 0.4, 0.45]
  **rated_powers** = [15000, 30000, 50000] (in Watts)

*Asif Newaz, Aseer Imad Keats, Salman Hossain*
*Lecturer, Department of EEE, IUT*

- Wind speed (in m/s) for 3 locations:

**wind_speeds** =

```
[
   5.0, 6.0, 5.5, 7.0, 6.5;   % Location 1 (hourly wind speeds in m/s)
   6.0, 7.5, 6.5, 8.0, 7.0;   % Location 2 (hourly wind speeds in m/s)
   4.0, 4.5, 5.0, 5.5, 6.0   % Location 3 (hourly wind speeds in m/s)
];
```

**Test Case – 01**

- Output: Total Power Output = [X Y Z]

- X – total output from 3 wind turbines combined for location 1
- Y – total output from 3 wind turbines combined for location 2
- Z – total output from 3 wind turbines combined for location 3

# Exercise - 08

**Problem statement:** Take an array or a matrix as the input and provide the following –

I. Count the no. of negative numbers.
II. Find the indices of 0 (linear and subscripts both).
III. Extract the positive numbers [sequence should be maintained]
IV. Find the mean and standard deviation of the numbers.
V. Extract the square numbers from the list.
VI. Rearrange the array placing the even elements first then 0 (if present) then the odd elements.

- Solve this question using a vectorization approach. Try not to use any loop to solve the problem.
- For a matrix, sequence means linear index sequence.
- Square numbers are 1,4,9,16, … …

**Test Case – 01:**

- Input: a= [4, -3, 21, 23, -5, 6, 0, -8, 6, 4, -2, 4, 5, 5, 11, 3, 6, -5, 6, 0, -77, -6, -90, -6, 0, 9, 8]

**Test Case – 02:**

- Input: a= randi ([-123,440], 6, 9)

- Standard deviation can be found using the *std( )* function.

*Asif Newaz, Aseer Imad Keats, Salman Hossain*
*Lecturer, Department of EEE, IUT*