Lab Work — Frequently Used Methods

Objective

This lab introduces Python’s most commonly used built-in methods that simplify data handling in engineering applications. You will practice using functions such as enumerate, zip, map, filter, sorted, reversed, range, any, all, type checking, and numeric utilities. These tools help process measurements, analyze datasets, perform calculations, and structure information efficiently. By applying them to electrical and real-world examples, the lab strengthens your ability to write cleaner, faster, and more Pythonic code.

Task 1 — enumerate()

List of voltages: print index + reading.

List of resistors: print index + value.

```
In [2]: voltages= [10, 12, 15, 20, 30]
        for SerialNumber, voltage in enumerate(voltages):
            print(SerialNumber, voltage)
```

```
0 10
1 12
2 15
3 20
4 30
```

```
In [3]: resistances=[10,20,30,40,50]
        for serial_number, resistance in enumerate(resistances):
            print(serial_number, resistance )
```

```
0 10
1 20
2 30
3 40
4 50
```

Manipulated

```
In [4]: resistances=[10,20,30,40,50]
        for serial_number, resistance in enumerate(resistances, start=1):
            print(serial_number, resistance )
```

```
1 10
2 20
3 30
4 40
5 50
```

Task 2 — zip()

Pair current readings with voltage readings.

Pair component names with ratings.

In [5]:
```
currents=[10,20,30]  # Amperes
voltages=[4,6,8]     # Volts
for a,b in zip(currents,voltages):
    print(a,b)
```

```
10 4
20 6
30 8
```

In [6]:
```
components=["Resistor", "Capacitor","Inductor"]
ratings=["220 Ω","110 µF","50 mH"]
for a,b in zip(components, ratings):
    print(a,b)
```

```
Resistor 220 Ω
Capacitor 110 µF
Inductor 50 mH
```

Task 3 — map()

Convert Celsius list → Fahrenheit.

Convert voltages → kilovolts.

In [7]:
```
celsius=[10,20,30,40]
#conversion using map and lambda
fahrenheit= list(map(lambda c:(c*9/5)+32,celsius))
print(fahrenheit)
```

```
[50.0, 68.0, 86.0, 104.0]
```

In [8]:
```
voltages=[20,30,40,2000]   # in volts
kilovolts= list(map(lambda a: a/1000, voltages))
print(kilovolts)
```

```
[0.02, 0.03, 0.04, 2.0]
```

Task 4 — filter()

Filter current readings > 10 A.

Filter capacitor values > 50 µF.

In [9]:
```
current=[10,30,5,6,78,89,11,1103]
high_currents= list(filter(lambda c: c>10, current))
print(high_currents)
```

```
[30, 78, 89, 11, 1103]
```

In [10]:
```
capacitorsRatings=( 12, 20, 50, 100, 70, 89, 113)   #µf
```

```
high_rating= list(filter(lambda a: a>50, capacitorsRatings))
print(high_rating)
```

```
[100, 70, 89, 113]
```

Task 5 — sorted()

Sort resistance values ascending.

Sort student marks descending.

In [11]:

```
resistance=[30,45,89,34,67,88,113]
print(sorted(resistance))
```

```
[30, 34, 45, 67, 88, 89, 113]
```

In [12]:

```
marks=[56,78,89,65,12,3,100,98]
print(sorted(marks,reverse=True))
```

```
[100, 98, 89, 78, 65, 56, 12, 3]
```

In [13]:

```
marks=[56,78,89,65,12,3,100,98]
print(sorted(marks))
print(sorted(marks,reverse=True))
```

```
[3, 12, 56, 65, 78, 89, 98, 100]
[100, 98, 89, 78, 65, 56, 12, 3]
```

Task 6 — reversed()

Reverse daily power consumption list.

Reverse current measurement sequence.

In [14]:

```
power_consumption=(340,56,78,90,43)
print(list(reversed(power_consumption)))
```

```
[43, 90, 78, 56, 340]
```

In [15]:

```
current_rec=( 12,45,67,89,23,56)
print(list(reversed(current_rec)))
```

```
[56, 23, 89, 67, 45, 12]
```

Task 7 — range()

Time slots 0 → 10 seconds.

Resistor IDs 1 → 20.

In [16]:

```
for t in range(0,11):
    print(t)
```

```
0
1
2
3
4
5
6
7
8
9
10
```

In [17]:

```python
for i in range(1,21):
    print(i)
```

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
```

Task 8 — any()

Check if any voltage > 220 V.

Check if any motor overloaded.

In [18]:

```python
voltages = [210, 215, 220, 225, 230]
high_voltage = any(v > 220 for v in voltages)
print( high_voltage)
```

```
True
```

In [19]:

```python
motor_status = [False, False, True, False]
overloaded = any(motor_status)
print(overloaded)
```

```
True
```

Task 9 — all()

Check if all resistors > 100 Ω.

Check if all solar panels active

In [20]:

```
resistances = [120, 150, 200, 180]
all_above_100 = all(r > 100 for r in resistances)
print( all_above_100)
```

```
True
```

In [21]:

```
solar_panels=[True, True , True]
active=all(solar_panels)
print(active)
```

```
True
```

In [22]:

```
solar_panels=[True, True , True, False]
active=all(solar_panels)
print(active)
```

```
False
```

Task 10 — isinstance()

Check if input current is integer.

Check if component rating is string or number.

In [23]:

```
current = 10  #  input current
is_integer = isinstance(current, int)
print( is_integer)
```

```
True
```

In [24]:

```
component_rating = "220Ω"
is_string = isinstance(component_rating, str)
is_number = isinstance(component_rating, (int, float))  # Can be int or float
print(is_string)
print( is_number)
```

```
True
False
```

In [25]:

```
component_rating = 220 #Ω
is_string = isinstance(component_rating, str)
is_number = isinstance(component_rating, (int, float))  # Can be int or float
print(is_string)
print( is_number)
```

```
False
True
```

Task 11 — type()

Type of resistance value (float).

Type of list of voltages.

In [26]:

```
resistance = 220.5
print(type(resistance))
```

```
<class 'float'>
```

In [27]:

```
voltages=[ 220,34,56,110,11]
print(type(voltages))
```

```
<class 'list'>
```

Task 12 — len()

Count resistors in project list.

Count characters in project name.

In [28]:

```
R=[110,340,45,2,4,7,8,9,5,1,2,3,33]
print(len(R))
```

```
13
```

In [29]:

```
project="FSO_PERFORMANCE"
print(len(project))
```

```
15
```

Task 13 — sum()

Total power consumed.

Total series resistance.

In [30]:

```
pow_Consumed=[10,20,45,30,11,100]
print(sum(pow_Consumed))
```

```
216
```

In [31]:

```
R=[10,20,30,40,50,60]
R_eq=sum(R)
print(R_eq)
```

```
210
```

Task 14 — max()

Maximum voltage in list.

Maximum motor current.

In [32]:

```
voltage=[10,45,78,90,100,69,113]
print(max(voltage))
```

```
113
```

In [33]:

```
C=[10,20,30,40,56,78]
print(max(C))
```

```
78
```

Task 15 — min()

Minimum temperature in list.

Minimum resistor value.

In [34]:

```
temp=[23,34,56,41]
print(min(temp))
```

```
23
```

In [35]:

```
R=[12,3,45,78.9,100]
print(min(R))
```

```
3
```

Task 16 — abs()

Absolute value of AC readings.

Magnitude of negative voltage

In [36]:

```
ac_currents=[-12,-34,56,-11]
abs_currents = list(map(abs, ac_currents))
print(abs_currents)
```

```
[12, 34, 56, 11]
```

In [37]:

```
voltage= -11
print(abs(voltage))
```

```
11
```

Task 17 — round()

Round voltage to 2 decimals.

Round frequency to nearest intege

In [38]:

```
voltage= 220.56789
print(round(voltage,2))
```

```
220.57
```

In [39]:

```
fre_readings=[11.23,56.00,345.678,78.89]
```

```
rounded_fre=[round(f) for f in fre_readings]
print(rounded_fre)
```

```
[11, 56, 346, 79]
```

Task 18 — divmod()

Divide 87 cables into bundles + remainder.

Divide 42 LEDs into groups of 12 + remainder.

In [40]:

```
total_cables = 87
duct_capacity = 10
filled_ducts, remaining_cables = divmod(total_cables, duct_capacity)
print(filled_ducts, remaining_cables)
```

```
8 7
```

In [41]:

```
total_groups=12
total_LEDs=42
groups, extra_leds = divmod(total_LEDs, total_groups)

print("Number of groups:", groups)
print("Extra LEDs:", extra_leds)
```

```
Number of groups: 3
Extra LEDs: 6
```

Task 19 — pow()

Square current values for power calculation.

In [42]:

```
current=[2,4,5,6]
squ_current=[i**2 for i in current]
print(squ_current)
```

```
[4, 16, 25, 36]
```

LAB ASSESSMENT:-

In [43]:

```
chairmen=[" Dr.Amjid Ali"," Dr.Umair","Dr.Farhan", "Dr.Usman","Dr.Shahid"]
dept=["Civil","Mechanical","Electrical","Computer Science","Industrial"]
for a,b in zip(chairmen,dept):
   print ( "The chairman of",b,"department is",a,".")
```

```
The chairman of Civil department is  Dr.Amjid Ali .
The chairman of Mechanical department is  Dr.Umair .
The chairman of Electrical department is Dr.Farhan .
The chairman of Computer Science department is Dr.Usman .
The chairman of Industrial department is Dr.Shahid .
```