Quality Assignment Code Part

14.

a)

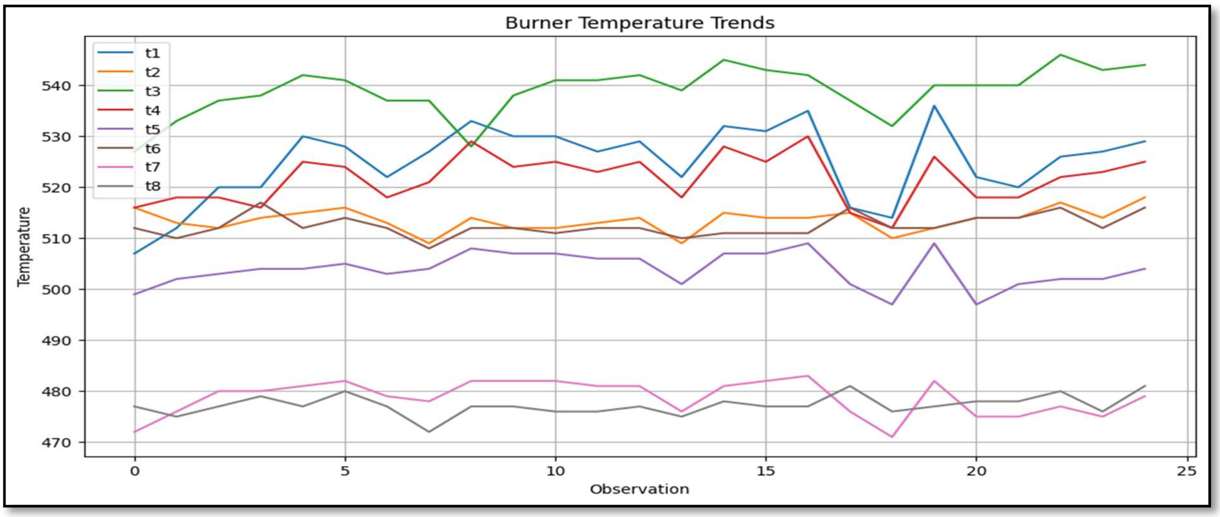
```python
# Python code for boiler temperature analysis
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Load data from file
data = pd.read_csv("/content/boiler.txt", sep='\s+')

# Summary statistics
data.describe()
```

Output:

	t1	t2	t3	t4	t5	t6	t7	t8
count	25.000000	25.00	25.000000	25.000000	25.000000	25.000000	25.00000	25.00000
mean	525.000000	513.56	538.920000	521.680000	503.800000	512.440000	478.72000	477.24000
std	7.348469	2.20	4.795136	4.723346	3.378856	2.122891	3.40979	1.96384
min	507.000000	509.00	527.000000	512.000000	497.000000	508.000000	471.00000	472.00000
25%	520.000000	512.00	537.000000	518.000000	502.000000	511.000000	476.00000	476.00000
50%	527.000000	514.00	540.000000	523.000000	504.000000	512.000000	480.00000	477.00000
75%	530.000000	515.00	542.000000	525.000000	507.000000	514.000000	482.00000	478.00000
max	536.000000	518.00	546.000000	530.000000	509.000000	517.000000	483.00000	481.00000

```python
# Plotting all burners
plt.figure(figsize=(12, 6))
for col in data.columns:
    plt.plot(data[col], label=col)
plt.title("Burner Temperature Trends")
plt.xlabel("Observation"
]]
plt.ylabel("Temperature")
plt.legend()
plt.grid(True)
plt.show()
```

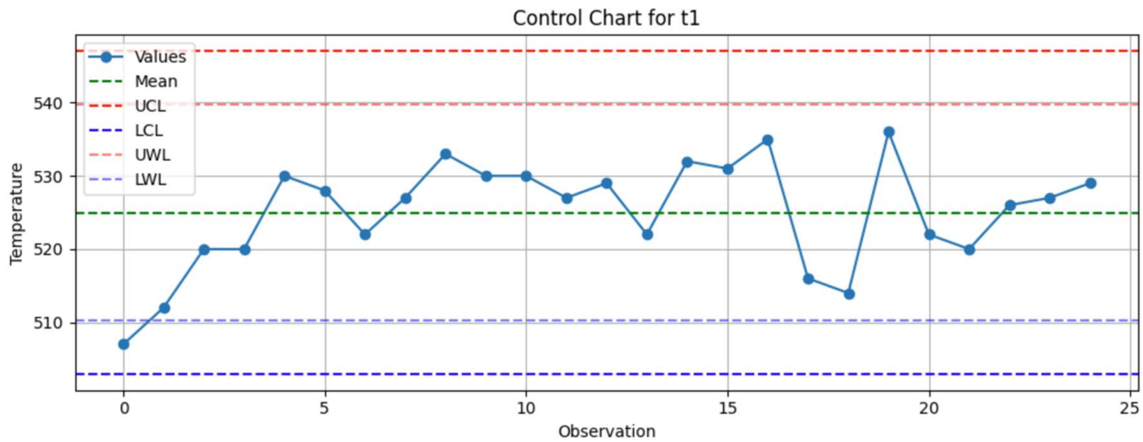
Output:

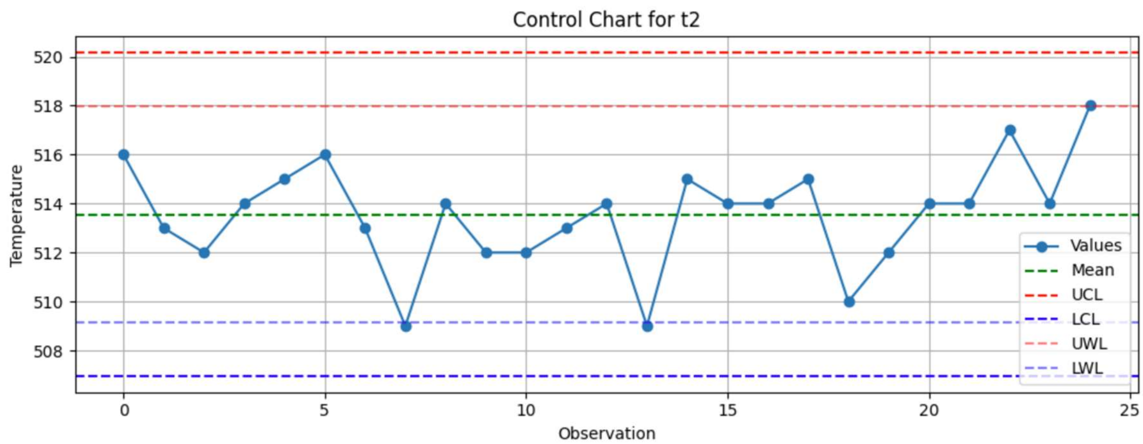


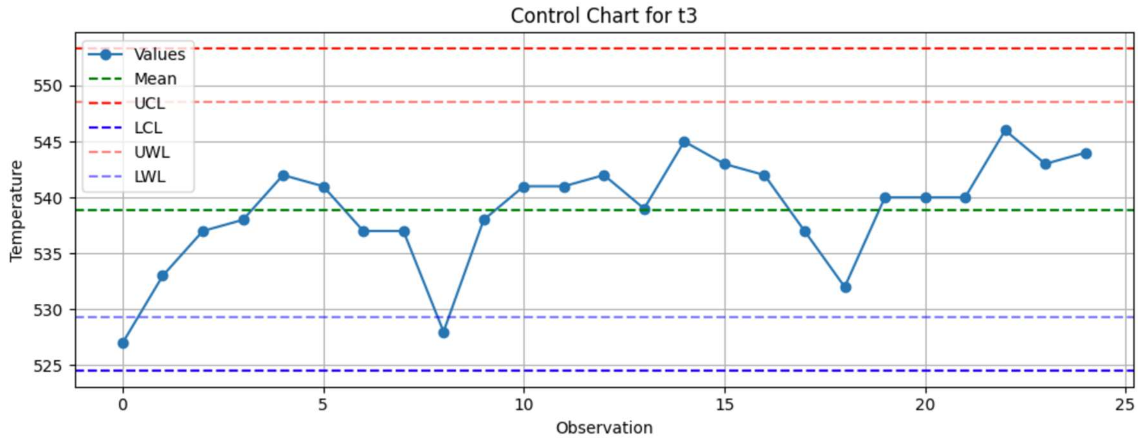
For most points the temperatures are not varying much. At 18th and 19th point there is dip in the temperature values in multiple sensors.

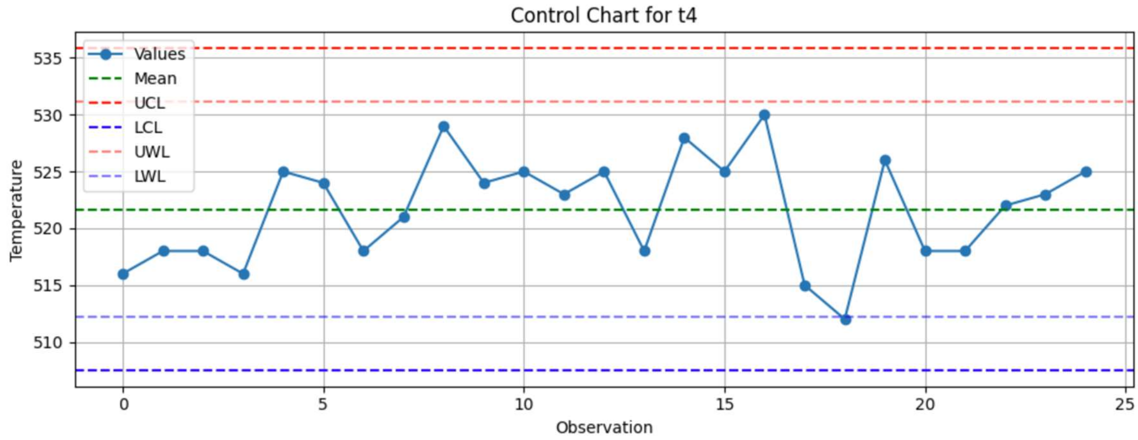
b)

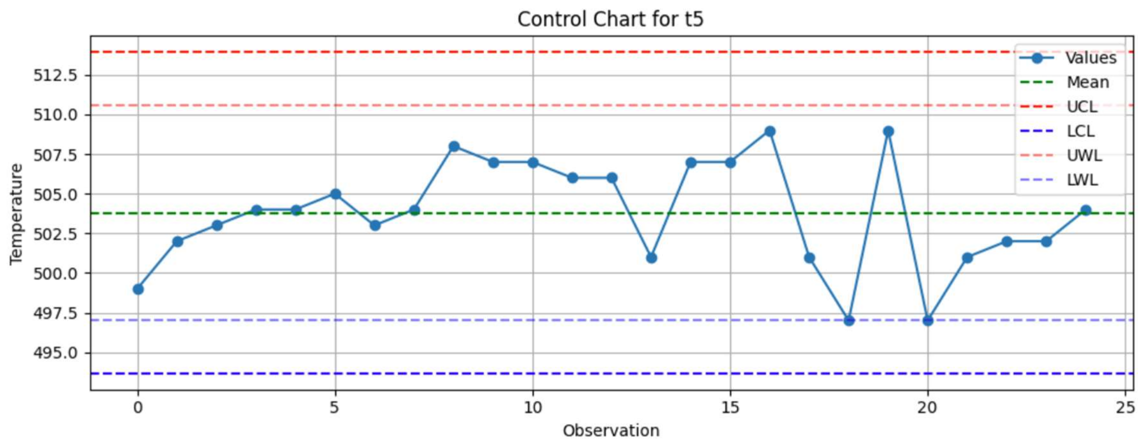
```
# Control charts - individual values with mean, 3-sigma limits and warning limits
for col in data.columns:
    mean = data[col].mean()
    std = data[col].std()
    ucl = mean + 3 * std
    lcl = mean - 3 * std
    uwl = mean + 2 * std
    lwl = mean - 2 * std

    plt.figure(figsize=(10, 4))
    plt.plot(data[col], marker='o', label='Values')
    plt.axhline(mean, color='green', linestyle='--', label='Mean')
    plt.axhline(ucl, color='red', linestyle='--', label='UCL')
    plt.axhline(lcl, color='blue', linestyle='--', label='LCL')
    plt.axhline(uwl, color='red',alpha = 0.5, linestyle='--', label='UWL')
    plt.axhline(lwl, color='blue',alpha = 0.5, linestyle='--', label='LWL')
    plt.title(f"Control Chart for {col}")
    plt.xlabel("Observation")
    plt.ylabel("Temperature")
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()
```

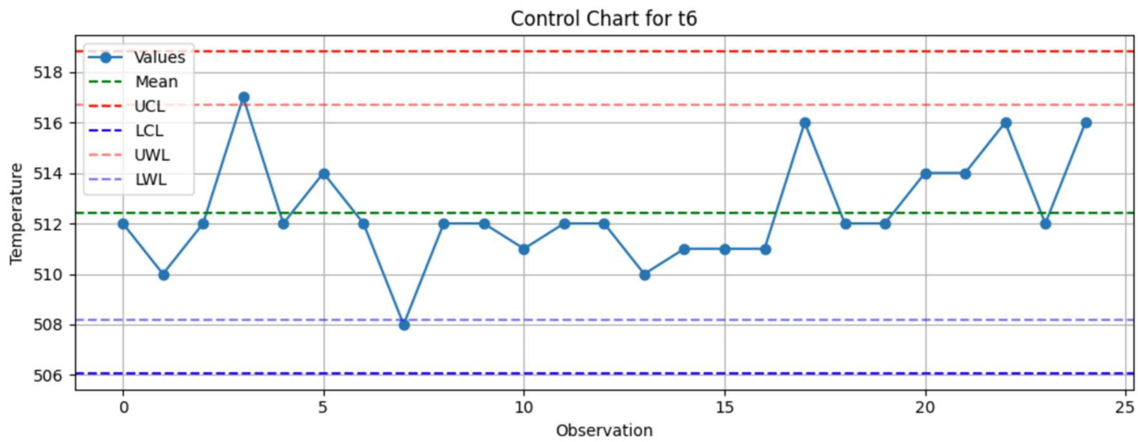


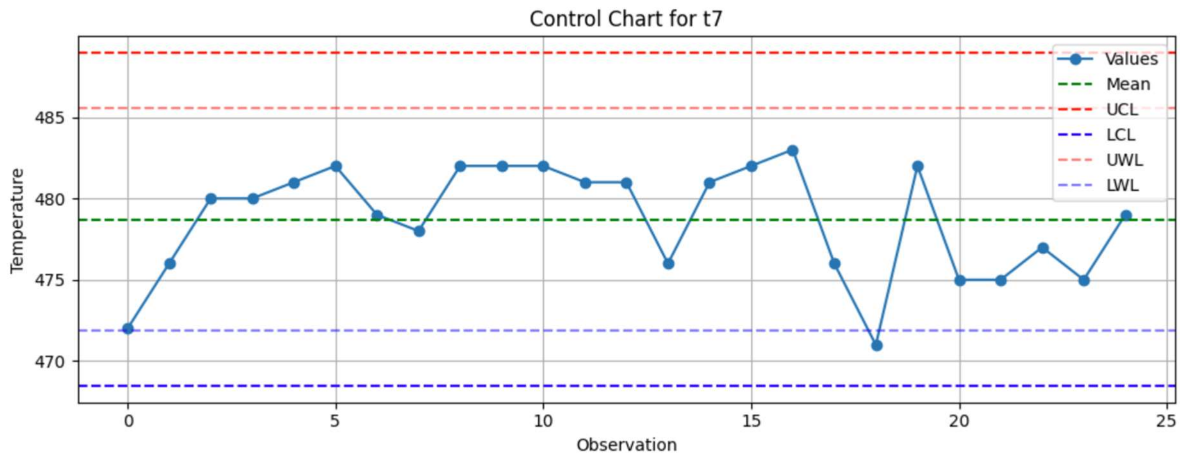


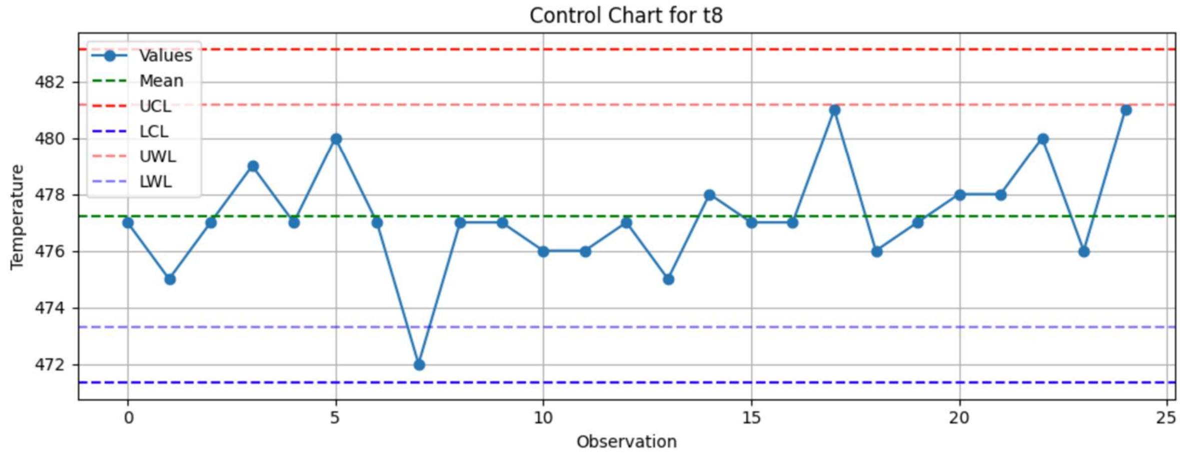

Control Chart for t3
Values
Mean
UCL
LCL
UWL
LWL
Temperature
Observation


Control Chart for t4
Values
Mean
UCL
LCL
UWL
LWL
Temperature
Observation


Control Chart for t5
Values
Mean
UCL
LCL
UWL
LWL
Temperature
Observation


Control Chart for t6
Values
Mean
UCL
LCL
UWL
LWL
Temperature
Observation





c)

Using the decision rules given by the Western Electric Statistical Control Handbook we can see that sensor **t6 can be treated as out of control** as more than **8 consecutive data points lie below the center line**.

```
# Process Capability Indices
usl = 550
lsl = 470
for col in data.columns:
    std = data[col].std()
    mean = data[col].mean()
    cp = (usl - lsl) / (6 * std)
    print(f"Burner: {col}")
    print(f"    Cp: {cp:.3f}")
```

Output:

```
Burner: t1
    Cp: 1.814
Burner: t2
    Cp: 6.061
Burner: t3
    Cp: 2.781
Burner: t4
    Cp: 2.823
Burner: t5
    Cp: 3.946
Burner: t6
    Cp: 6.281
Burner: t7
    Cp: 3.910
Burner: t8
    Cp: 6.789
```

15.

Initial Code:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

# Load data
df = pd.read_csv('/content/pistonrings.txt', sep='\s+')

# Separate into Phase I and Phase II
phase1 = df[df['trial'] == True]
phase2 = df[df['trial'] == False]
```

a)

Code:

```
# === (a) Summary Statistics ===
group1 = phase1.groupby('sample')['diameter']
group2 = phase2.groupby('sample')['diameter']

means1 = group1.mean()
stds1 = group1.std()
means2 = group2.mean()
stds2 = group2.std()

print("Phase I - Mean of means:", means1.mean())
print("Phase I - Mean of std devs:", stds1.mean())
print("Phase II - Mean of means:", means2.mean())
print("Phase II - Mean of std devs:", stds2.mean())
```

Output:

Phase I - Mean of means: 74.001176

Phase I - Mean of std devs: 0.009240036602285218

Phase II - Mean of means: 74.00765333333334

Phase II - Mean of std devs: 0.00976175748705033

Observations:

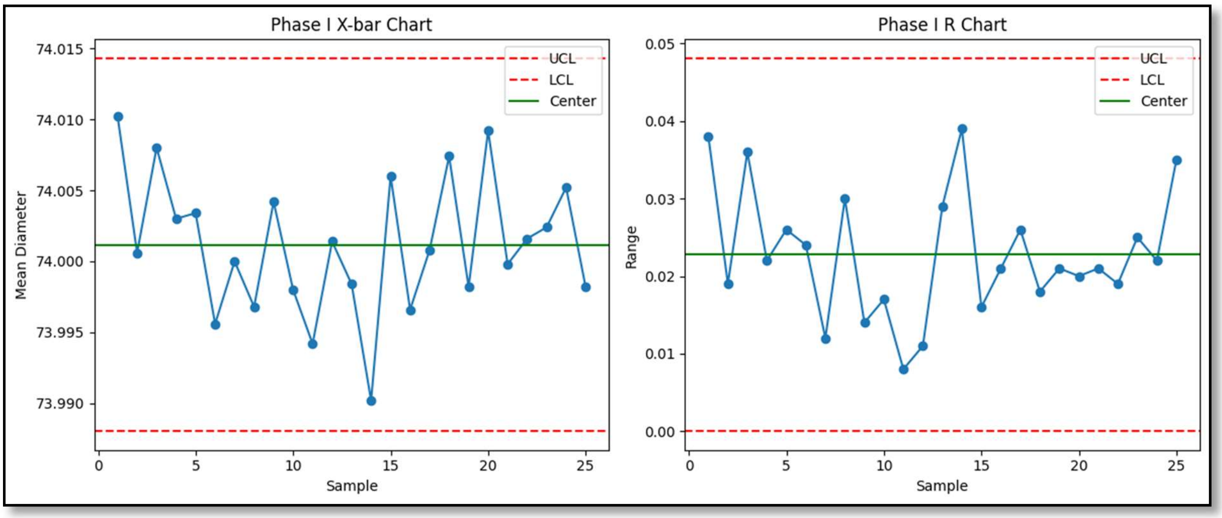
We can see that the mean of means as well as the standard deviation of means has slightly increased in the Phase II.

b)

Code:

```
# === (b) Control Charts for Phase I ===
xbar1 = means1
R1 = group1.max() - group1.min()

xbar_bar1 = xbar1.mean()
R_bar1 = R1.mean()

n = group1.count().iloc[0]
A2 = {2: 1.88, 3: 1.023, 4: 0.729, 5: 0.577}[n]
D3 = {2: 0, 3: 0, 4: 0, 5: 0}[n]
D4 = {2: 3.267, 3: 2.574, 4: 2.282, 5: 2.114}[n]

xbar_UCL1 = xbar_bar1 + A2 * R_bar1
xbar_LCL1 = xbar_bar1 - A2 * R_bar1
R_UCL1 = D4 * R_bar1
R_LCL1 = D3 * R_bar1

# Plot X-bar chart
plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.plot(xbar1, marker='o')
plt.axhline(xbar_UCL1, color='red', linestyle='--', label='UCL')
plt.axhline(xbar_LCL1, color='red', linestyle='--', label='LCL')
plt.axhline(xbar_bar1, color='green', linestyle='-', label='Center')
plt.title('Phase I X-bar Chart')
plt.xlabel('Sample')
plt.ylabel('Mean Diameter')
plt.legend()

# Plot R chart
plt.subplot(1, 2, 2)
plt.plot(R1, marker='o')
plt.axhline(R_UCL1, color='red', linestyle='--', label='UCL')
plt.axhline(R_LCL1, color='red', linestyle='--', label='LCL')
plt.axhline(R_bar1, color='green', linestyle='-', label='Center')
plt.title('Phase I R Chart')
plt.xlabel('Sample')
plt.ylabel('Range')
plt.legend()
plt.tight_layout()
plt.show()
```

Output:

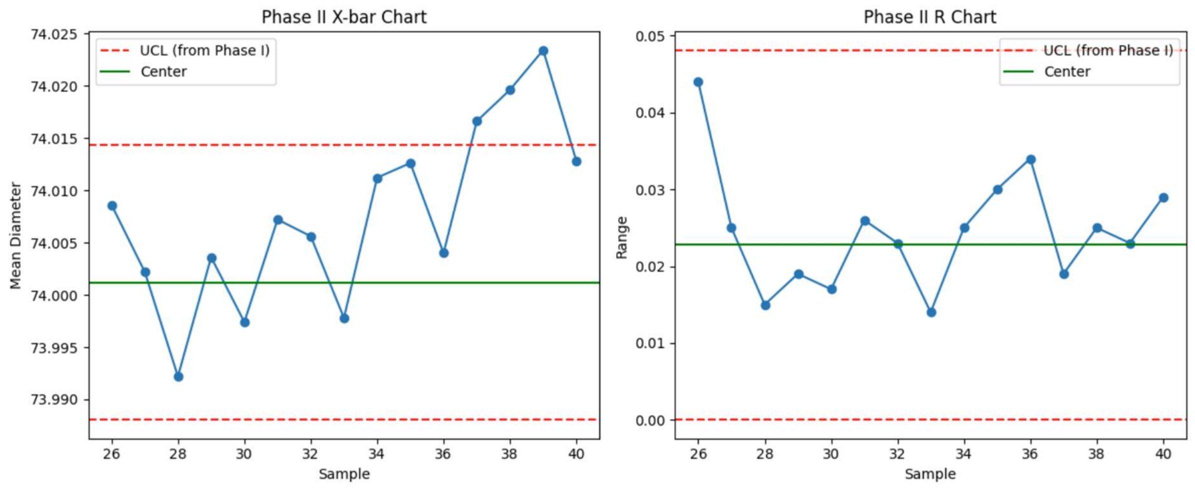


No sample points lie outside the control limits c)

Code:

```
# === (c) Apply Phase I Limits to Phase II ===
xbar2 = group2.mean()
R2 = group2.max() - group2.min()

plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
plt.plot(xbar2, marker='o')
plt.axhline(xbar_UCL1, color='red', linestyle='--', label='UCL (from Phase I)')
plt.axhline(xbar_LCL1, color='red', linestyle='--')
plt.axhline(xbar_bar1, color='green', linestyle='-', label='Center')
plt.title('Phase II X-bar Chart')
plt.xlabel('Sample')
plt.ylabel('Mean Diameter')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(R2, marker='o')
plt.axhline(R_UCL1, color='red', linestyle='--', label='UCL (from Phase I)')
plt.axhline(R_LCL1, color='red', linestyle='--')
plt.axhline(R_bar1, color='green', linestyle='-', label='Center')
plt.title('Phase II R Chart')
plt.xlabel('Sample')
plt.ylabel('Range')
plt.legend()
plt.tight_layout()
plt.show()

# Identify any out-of-control points
print("\nOut-of-control X-bar (Phase II):")
print(xbar2[(xbar2 > xbar_UCL1) | (xbar2 < xbar_LCL1)])

print("\nOut-of-control R (Phase II):")
print(R2[(R2 > R_UCL1) | (R2 < R_LCL1)])
```

Output:




```
Out-of-control X-bar (Phase II):
sample
37     74.0166
38     74.0196
39     74.0234
Name: diameter, dtype: float64

Out-of-control R (Phase II):
Series([], Name: diameter, dtype: float64)
```

As we can see that the means are steadily increasing but their range is fairly in control. This might be possible due to wear and tear caused by longer operation.

d)

Code:

```
# === Part (d) ===
# Process capability (Cp) from Phase I data
USL = 74.030
LSL = 73.970
target = 74.000
std_overall = phase1['diameter'].std()

Cp = (USL - LSL) / (6 * std_overall)
print(f"Process Capability Index (Cp): {Cp:.4f}")

if Cp >= 1.33:
    print("Process is capable.")
elif 1.0 <= Cp < 1.33:
    print("Process is marginally capable.")
else:
    print("Process is not capable. Adjustment needed.")
```

Output:

Process Capability Index (Cp): 0.9931

Process is not capable. Adjustment needed.

Observations:

It is clear from the process capability index value that some adjustment is needed, some of the adjustments in the manufacturing are as follows:

i. Tighten control over machine settings
ii. Improve operator training and standardize procedures.
iii. Implement preventive maintenance schedules.
iv. Verifying measurement system.