Tutorial

A Concise Introduction to Computational Tools

Contents

List of Figures

List of Tables

Preface

A Concise Introduction to Computational Tools provides a series of tutorials derived from lecture notes, offering clear and focused guidance on essential topics. No prior programming experience is required.

Students are expected to be comfortable with college-level mathematics and science and are strongly encouraged to consult reliable reference texts. For example, foundational and practical texts in mathematics, applied mechanics and computational tools, including Bird (2021), Hannah & Hillier (1995), Russell et al. (2021), (Russell et al., 2022), Bolton (2021), Shaw (2017), Sweigart (2021), and Sweigart (2020).

The tutorials also assume working familiarity with either macOS or Microsoft Windows. For best results, these materials should be used on a laptop or desktop computer rather than a tablet or smartphone.

Study Guide

First and foremost, solve the problem sets using only pen, paper, and a calculator. After completing a problem by hand, slightly modify the conditions or variables and solve it again. Then, input the new values into the Python script and compare the results with your hand-calculated solution.

- Buddy system: Study with a classmate. Teaching and helping one another significantly deepens understanding of the material. Students are strongly encouraged to work through problem sets collaboratively. For example, one student solves the odd-numbered problems while the other solves the even-numbered ones; then each explains their solutions to the other.

- Practice, practice, practice: As the saying goes, "practice makes perfect." More accurately in this context, "good practice makes perfect." That means pen-and-paper-first practice. Use Python as a verification tool, not a crutch. Always derive the solution analytically on paper first. Only after obtaining a complete hand-calculated result should you run the corresponding Python script. This approach echoes a proverb commonly attributed to Confucius:

  > I hear and I forget
  >
  > I see and I remember
  >
  > I do and I understand

- Maintain a solution notebook: For each major problem, keep a single document (physical or digital) containing:

  - The full hand derivation
  - Key numerical results with units
  - A brief paragraph summarizing insights or reconciling any intermediate discrepancies

  This record becomes an invaluable review resource before exams.

Python Tutorial

This tutorial introduces Python programming, covering basic concepts with examples to illustrate key points. We will start by using Python as a calculator, then explore variables, functions, and control flow.

Requirements

To follow this tutorial, the easiest way to get started is by using a web-based Python environment. This lets you write and run Python code right in your browser; no downloads or setup needed. I recommend python-fiddle.com, an easy-to-use online editor that lets you experiment with Python instantly and solve your problem sets effortlessly.

If you prefer working on your own computer, make sure you have Python (version 3.10 or later) installed. Python works on Windows, macOS, and Linux. You'll also need a text editor or an Integrated Development Environment (IDE) to write your code. I recommend Positron, a beginner-friendly IDE with a built-in terminal, though other editors like VS Code or PyCharm are also good options.

Basic Syntax

Python uses indentation (typically four spaces) to define code blocks. A colon (`:`) introduces a block, and statements within the block must be indented consistently. Python is case-sensitive, so `Variable` and `variable` are distinct identifiers. *Statements typically end with a newline, but you can use a backslash (\) to continue a statement across multiple lines.*

```python
total = 1 + 2 + 3 + 4 + 5
print(total)  # Output: 15
```

Basic syntax rules:

- Comments start with `#` and extend to the end of the line.

- Strings can be enclosed in single quotes (`'`), double quotes (`"`), or triple quotes (`'''` or `"""`) for multi-line strings.
- Python is case-sensitive, so `Variable` and `variable` are different identifiers.

The `print()` Function

The `print()` function displays output in Python.

```
name = "Rudolf Diesel"
year = 1858
print(f"{name} was born in {year}.")
```

Output: `Rudolf Diesel was born in 1858.`

The following table illustrates common f-string formatting options for the `print()` function:

Format	Code	Example	Output
Round to 2 decimals	`f"{x:.2f}"`	`print(f"{3.1415:.2f}")`	`3.14`
Round to whole number	`f"{x:.0f}"`	`print(f"{3.9:.0f}")`	`4`
Thousands separator	`f"{x:,.2f}"`	`print(f"{1234.567:,.2f}")`	`1,234.57`
Percentage	`f"{x:.1%}"`	`print(f"{0.756:.1%}")`	`75.6%`
Currency (JPY)	`f"¥{x:,.0f}"`	`print(f"¥{1234:,.0f}")`	`¥1,234`

Variables and Data Types

Variables store data and are assigned values using the `=` operator.

```
x = 10
y = 3.14
name = "Rudolph"
```

Python has several built-in data types, including:

- Integers (`int`): Whole numbers, e.g., `10`, `-5`
- Floating-point numbers (`float`): Decimal numbers, e.g., `3.14`, `-0.001`
- Strings (`str`): Text, e.g., `"Hello"`, `'World'`
- Booleans (`bool`): `True` or `False`

Arithmetic Operations

```
a = 10
b = 3
print(a + b)  # Addition: 13
print(a - b)  # Subtraction: 7
print(a * b)  # Multiplication: 30
print(a / b)  # Division: 3.3333...
print(a // b) # Integer Division: 3
print(a ** b) # Exponentiation: 1000
```

String Operations

```
first_name = "Rudolph"
last_name = "Diesel"
full_name = first_name + " " + last_name  # Concatenation using +
print(full_name)  # Output: Rudolph Diesel
print(f"{first_name} {last_name}")  # Concatenation using f-string
print(full_name * 2)  # Repetition: Rudolph DieselRudolph Diesel
print(full_name.upper())  # Uppercase: RUDOLPH DIESEL
```

Note: String repetition (`*`) concatenates the string multiple times without spaces. For example, `full_name * 2` produces `Rudolph DieselRudolph Diesel`.

Python as a Calculator in Interactive Mode

Python's interactive mode allows you to enter commands and see results immediately, ideal for quick calculations. To start, open a terminal (on macOS, Linux, or Windows) and type:

```
python3  # Use 'python' on Windows if 'python3' is not recognized
```

You should see the Python prompt:

```
>>>
```

Enter expressions and press **Enter** to see results:

```
2 + 3  # Output: 5
7 - 4  # Output: 3
6 * 9  # Output: 54
8 / 2  # Output: 4.0
8 // 2 # Output: 4
2 ** 3 # Output: 8
```

Parentheses for Grouping

```
(2 + 3) * 4  # Output: 20
2 + (3 * 4)  # Output: 14
```

Variables

```
x = 10
y = 3
x / y  # Output: 3.3333333333333335
```

Exiting Interactive Mode

To exit, type:

```
exit()
```

Alternatively, use: - **Ctrl+D** (macOS/Linux) - **Ctrl+Z** then Enter (Windows)

Control Flow

Control flow statements direct the execution of code based on conditions.

Conditional Statements

Conditional statements allow you to execute different code blocks based on specific conditions. Python provides three keywords for this purpose:

- **`if`**: Evaluates a condition and executes its code block if the condition is `True`.
- **`elif`**: Short for "else if," it checks an additional condition if the preceding `if` or `elif` conditions are `False`. You can use multiple `elif` statements to test multiple conditions sequentially, and Python will execute the first `True` condition's block, skipping the rest.
- **`else`**: Executes a code block if none of the preceding `if` or `elif` conditions are `True`. It serves as a fallback and does not require a condition.

The following example uses age to categorize a person as a Minor, Adult, or Senior, demonstrating how `if`, `elif`, and `else` work together.

```
# Categorize a person based on their age
age = 19
if age < 18:
    print("Minor")
elif age <= 64:
    print("Adult")
```

```python
else:
    print("Senior")
```

Output: `Adult`

For Loop

A `for` loop iterates over a sequence (e.g., list or string).

```python
components = ["piston", "liner", "connecting rod"]
for component in components:
    print(component)
```

Output:

```
piston
liner
connecting rod
```

While Loop

A `while` loop executes as long as a condition is true. Ensure the condition eventually becomes false to avoid infinite loops.

```python
count = 0
while count <= 5:
    print(count)
    count += 1
```

Output:

```
0
1
2
3
4
5
```

Functions

The `def` Keyword

Functions are reusable code blocks defined using the `def` keyword. They can include default parameters for optional arguments.

```python
def add(a, b=0):
    return a + b
print(add(5))      # Output: 5
print(add(5, 3))   # Output: 8

def multiply(*args):
    result = 1
    for num in args:
        result *= num
    return result
print(multiply(2, 3, 4))  # Output: 24
```

The `lambda` Keyword

The `lambda` keyword creates anonymous functions for short, one-off operations, often used in functional programming.

```python
celsius_to_fahrenheit = lambda c: (c * 9 / 5) + 32
print(celsius_to_fahrenheit(25))  # Output: 77.0
```

The `math` Module

The `math` module provides mathematical functions and constants.

```python
import math
print(math.sqrt(16))  # Output: 4.0
print(math.pi)        # Output: 3.141592653589793
```

```python
import math
angle = math.pi / 4  # 45 degrees in radians
print(math.sin(angle))  # Output: 0.7071067811865475
print(math.cos(angle))  # Output: 0.7071067811865476
print(math.tan(angle))  # Output: 1.0
```

Note: Floating-point arithmetic may result in small precision differences, as seen in the `sin` and `cos` outputs.

```python
import math
print(math.log(10))       # Natural logarithm of 10: 2.302585092
print(math.log(100, 10))  # Logarithm of 100 with base 10: 2.0
```

Converting Between Radians and Degrees

The `math` module provides `math.radians()` to convert degrees to radians and `math.degrees()` to convert radians to degrees, which is useful for trigonometric calculations.

⚠ Warning

In Python, the trigonometric functions in the `math` module (`sin`, `cos`, `tan`, etc.) expect angles in radians, not degrees. Always convert degrees to radians before using these functions.

```python
import math
degrees = 180
radians = math.radians(degrees)
print(f"{degrees} degrees is {radians:.3f} radians")
radians = math.pi
degrees = math.degrees(radians)
print(f"{radians:.3f} radians is {degrees:.1f} degrees")
```

Writing Python Scripts

Write Python code in a `.py` file and run it as a script. Create a file named `script.py`:

```python
# script.py
import math
print("Square root of 16 is:", math.sqrt(16))
print("Value of pi is:", math.pi)
print("Sine of 90 degrees is:", math.sin(math.pi / 2))
print("Natural logarithm of 10 is:", math.log(10))
print("Logarithm of 100 with base 10 is:", math.log(100, 10))
```

To run the script, open a terminal, navigate to the directory containing `script.py` using the `cd` command (e.g., `cd /path/to/directory`), and type:

```
python3 script.py  # or python script.py on Windows
```

Output:

```
Square root of 16 is: 4.0
Value of pi is: 3.141592653589793
Sine of 90 degrees is: 1.0
Natural logarithm of 10 is: 2.302585092994046
Logarithm of 100 with base 10 is: 2.0
```

Python Style Guide

> Code is read far more often than it is written. — Guido van Rossum, creator of the Python programming language.

Adhere to PEP 8 as the primary style guide. This document outlines the coding conventions for Python code in the core standard library of the main Python distribution.

- Employ descriptive names for variables and functions. Avoid abbreviations or single-letter names unless they are widely conventional (e.g., i as a loop index).
- Organize code with consistent sectioning and clear headers or separators to enhance readability.
- Restrict line length to a maximum of 79 characters to ensure compatibility with print media and prevent horizontal overflow.
- Include meaningful inline comments where necessary to explain the purpose and intent of the code, particularly for complex logic.

Many modern Python projects enhance PEP 8 compliance by employing Black, the uncompromising code formatter. Black applies a deterministic style that is a strict subset of PEP 8, with a default line length of 88 characters for improved readability on contemporary displays. It can be configured to use 79 characters if required.

The following scripts illustrate the earlier examples in accordance with PEP 8 guidelines:

```
"""
Degrees-Radians Conversion
"""

import math

# Conversion from degrees to radians
input_degrees = 180.0
converted_radians = math.radians(input_degrees)

# Conversion from radians to degrees
input_radians = math.pi
converted_degrees = math.degrees(input_radians)

# Display results
print("Degrees to Radians Conversion:")
print(f"{input_degrees:.0f} degrees = {converted_radians:.3f} radians")
print()
print("Radians to Degrees Conversion:")
print(f"{input_radians:.3f} radians = {converted_degrees:.1f} degrees")
```

```
"""
Math Module Examples
```

```
Demonstrates common functions from the Python math module:
square root, pi constant, sine, natural logarithm,
and base-10 logarithm.
"""

import math

# Compute and display mathematical operations
sqrt_result = math.sqrt(16)
pi_value = math.pi
sine_90_degrees = math.sin(math.pi / 2)  # 90 degrees =  /2 radians
natural_log_10 = math.log(10)
log_base10_100 = math.log(100, 10)

# Display results with clear descriptions
print("Square root of 16 is:", sqrt_result)
print("Value of pi is:", pi_value)
print("Sine of 90 degrees is:", sine_90_degrees)
print("Natural logarithm of 10 is:", natural_log_10)
print("Logarithm of 100 with base 10 is:", log_base10_100)
```

Summary

You have now learned Python basics: syntax, variables, control flow, functions, and the math module. Practice all examples instantly at python-fiddle.com.

Chapter 1

SI Units

The International System of Units (SI) is the globally accepted standard for measurement. Established to provide a consistent framework for scientific and technical measurements, SI units facilitate clear communication and data comparison across various fields and countries. The system is based on seven fundamental units: the meter for length, the kilogram for mass, the second for time, the ampere for electric current, the kelvin for temperature, the mole for substance, and the candela for luminous intensity.

Table 1.1: Base SI units.

Physical Quantity	SI Base Unit	Symbol
Length	Meter	m
Mass	Kilogram	kg
Time	Second	s
Electric Current	Ampere	A
Temperature	Kelvin	K
Amount of Substance	Mole	mol
Luminous Intensity	Candela	cd

Table 1.2: Derived SI units.

Physical Quantity	Derived SI Unit	Symbol
Area	Square meter	m^2
Volume	Cubic meter	m^3
Speed	Meter per second	m/s
Acceleration	Meter per second squared	m/s^2
Force	Newton	N
Pressure	Pascal	Pa
Energy	Joule	J
Power	Watt	W
Electric Charge	Coulomb	C
Electric Potential	Volt	V
Resistance	Ohm	Ω
Capacitance	Farad	F
Frequency	Hertz	Hz
Luminous Flux	Lumen	lm
Illuminance	Lux	lx
Specific Energy	Joule per kilogram	J/kg
Specific Heat Capacity	Joule per kilogram Kelvin	$J/(kg \cdot K)$

Table 1.3: Common multiples and submultiples for SI units.

Factor	Prefix	Symbol
10^9	giga	G
10^6	mega	M
10^3	kilo	k
10^2	hecto	h
10^1	deca	da
10^{-1}	deci	d
10^{-2}	centi	c
10^{-3}	milli	m
10^{-6}	micro	µ

1.1 Metric Multipliers for Length, Area, and Volume

The metric system uses powers of ten to scale measurements. When converting **area** or **volume**, remember that the conversion factor must be **squared** or **cubed**, respectively.

Unit	Symbol	Multiplier (to metres)	1 m = ?	Area (1 m² = ?)	Volume (1 m³ = ?)
Millimetre	mm	(10^{-3})	1000 mm	$(1000)^2 = 10^6\text{mm}^2$	$(1000)^3 = 10^9\text{mm}^3$
Centimetre	cm	(10^{-2})	100 cm	$(100)^2 = 10^4\text{cm}^2$	$(100)^3 = 10^6\text{cm}^3$
Decimetre	dm	(10^{-1})	10 dm	$(10)^2 = 10^2\text{dm}^2$	$(10)^3 = 10^3\text{dm}^3$
Metre	m	(10^0)	—	—	—
Decametre	dam	(10^1)	0.1 dam	$(0.1)^2 = 10^{-2}\text{dam}^2$	$(0.1)^3 = 10^{-3}\text{dam}^3$
Hectometre	hm	(10^2)	0.01 hm	$(0.01)^2 = 10^{-4}\text{hm}^2$	$(0.01)^3 = 10^{-6}\text{hm}^3$
Kilometre	km	(10^3)	0.001 km	$(0.001)^2 = 10^{-6}\text{km}^2$	$(0.001)^3 = 10^{-9}\text{km}^3$

1.1.1 Quick Reference

Conversion Type	Relationship
Length	1 m = 10^3 mm = 10^2 cm
Area	1 $\text{m}^2 = 10^6\ \text{mm}^2 = 10^4\ \text{cm}^2$
Volume	1 $\text{m}^3 = 10^9\ \text{mm}^3 = 10^6\ \text{cm}^3$

Note: When converting between metric units:

- Multiply by 10^n when going to a smaller unit.
- Divide by 10^n when going to a larger unit.
- For **area**, square the length factor; for **volume**, cube it.

1.2 Unity Fraction

The **unity fraction** method, or **unit conversion using unity fractions**, is a systematic way to convert one unit of measurement into another. This method relies on multiplying by fractions that are equal to one, where the numerator and the denominator represent the same quantity in different units. Since any number multiplied by one remains the same, unity fractions allow for seamless conversion without changing the value.

The principle of unity fractions is based on:

1. **Setting up equal values**: Write a fraction where the numerator and denominator are equivalent values in different units, so the fraction equals one. For example, $\frac{1km}{1000m}$ is a unity fraction because 1 km equals 1000 m.
2. **Multiplying by unity fractions**: Multiply the initial quantity by the unity fraction(s) so that the undesired units cancel out, leaving only the desired units.

1.2.1 Classwork

Example 1.1. Suppose we want to convert 5 kilometers to meters.

1. Start with 5 kilometers:
$$5\,\text{km}$$
2. Multiply by a unity fraction that cancels kilometers and introduces meters. We use $(\frac{1000\,\text{m}}{1\,\text{km}}), since\, 1\,\text{km} = 1000\,\text{m}$:
$$5\,\text{km} \times \frac{1000\,\text{m}}{1\,\text{km}} = 5000\,\text{m}$$
3. The kilometers km cancel out, leaving us with meters m:

$$5\,\text{km} = 5000\,\text{m}$$

This step-by-step approach illustrates how the unity fraction cancels the undesired units and achieves the correct result in meters.

Unity fractions can be extended by using multiple conversion steps. For example, converting hours to seconds would require two unity fractions: one to convert hours to minutes and another to convert minutes to seconds. This approach ensures accuracy and is widely used in science, engineering, and other fields that require precise unit conversions.

Example 1.2. Convert $15\,\text{m/s}$ to km/h.

1. Start with $15\,\text{m/s}$.
2. To convert meters to kilometers, multiply by $\frac{1\,\text{km}}{1000\,\text{m}}$.
3. To convert seconds to hours, multiply by $\frac{3600\,\text{s}}{1\,\text{h}}$.

$$15\,\text{m/s} \times \frac{1\,\text{km}}{1000\,\text{m}} \times \frac{3600\,\text{s}}{1\,\text{h}} = 54\,\text{km/h}$$

The meters and seconds cancel out, leaving kilometers per hour: $54\,\text{km/h}$.

1.2.2 Self-Test

Instructions:

1. Use unity fraction to convert between derived SI units.
2. Show each step of your work to ensure accuracy.
3. Simplify your answers and include correct units.

---

1. **Speed**
   Convert $72\,\text{km/h}$ to m/s.
2. **Force**
   Convert $980\,\text{N}$ (newtons) to $\text{kg} \cdot \text{m/s}^2$.
3. **Energy**
   Convert $2500\,\text{J}$ (joules) to kJ.
4. **Power**
   Convert $1500\,\text{W}$ (watts) to kW.
5. **Pressure**
   Convert $101325\,\text{Pa}$ (pascals) to kPa.
6. **Volume Flow Rate**
   Convert $3\,\text{m}^3/\text{min}$ to L/s.

7. **Density**
   Convert $1000\,\mathrm{kg/m^3}$ to $\mathrm{g/cm^3}$.
8. **Acceleration**
   Convert $9.8\,\mathrm{m/s^2}$ to $\mathrm{cm/s^2}$.
9. **Torque**
   Convert $50\,\mathrm{N\cdot m}$ to $\mathrm{kN\cdot cm}$.
10. **Frequency**
    Convert $500\,\mathrm{Hz}$ (hertz) to kHz.
11. **Work to Energy Conversion**
    A force of $20\,\mathrm{N}$ moves an object $500\,\mathrm{cm}$. Convert the work done to joules.
12. **Kinetic Energy Conversion**
    Calculate the kinetic energy in kilojoules of a $1500\,\mathrm{kg}$ car moving at $72\,\mathrm{km/h}$.
13. **Power to Energy Conversion**
    A machine operates at $2\,\mathrm{kW}$ for 3 hours. Convert the energy used to megajoules.
14. **Pressure to Force Conversion**
    Convert a pressure of $200\,\mathrm{kPa}$ applied to an area of $0.5\,\mathrm{m^2}$ to force in newtons.
15. **Density to Mass Conversion**
    Convert $0.8\,\mathrm{g/cm^3}$ for an object with a volume of $250\,\mathrm{cm^3}$ to mass in grams.

1.2.3 Answer Key

1. $72\,\mathrm{km/h} = 20\,\mathrm{m/s}$
2. $980\,\mathrm{N} = 980\,\mathrm{kg\cdot m/s^2}$
3. $2500\,\mathrm{J} = 2.5\,\mathrm{kJ}$
4. $1500\,\mathrm{W} = 1.5\,\mathrm{kW}$
5. $101325\,\mathrm{Pa} = 101.325\,\mathrm{kPa}$
6. $3\,\mathrm{m^3/min} = 50\,\mathrm{L/s}$
7. $1000\,\mathrm{kg/m^3} = 1\,\mathrm{g/cm^3}$
8. $9.8\,\mathrm{m/s^2} = 980\,\mathrm{cm/s^2}$
9. $50\,\mathrm{N\cdot m} = 5\,\mathrm{kN\cdot cm}$
10. $500\,\mathrm{Hz} = 0.5\,\mathrm{kHz}$
11. $20\,\mathrm{N} \times 5\,\mathrm{m} = 100\,\mathrm{J}$
12. Kinetic energy $= 1500\,\mathrm{kg} \times (20\,\mathrm{m/s})^2/2 = 300\,\mathrm{kJ}$
13. $2\,\mathrm{kW} \times 3\,\mathrm{hours} = 21.6\,\mathrm{MJ}$
14. $200\,\mathrm{kPa} \times 0.5\,\mathrm{m^2} = 100,000\,\mathrm{N}$

15. $0.8\,\mathrm{g/cm}^3 \times 250\,\mathrm{cm}^3 = 200\,\mathrm{g}$

1.3 Condenser Vacuum

Condenser vacuum gauge reads 715 mmHg when barometer stands at 757 mmHg. State the absolute pressure in kN/m² and bar.

1.3.1 Given Data

$$P_{atm} = 757 \text{ mmHg}, \quad P_{vac} = 715 \text{ mmHg}$$

1.3.2 Absolute Pressure in mmHg

$$P_{abs} = P_{atm} - P_{vac} = 757 - 715 = 42 \text{ mmHg}$$

1.3.3 Convert mmHg → kN/m²

$$P = \rho g h = 13{,}600 \cdot 9.81 \cdot 0.001 = 133.416 \text{ Pa} = 0.133416 \text{ kN/m}^2$$

$$P_{abs} = 42 \cdot 0.133416 = 5.6034 \text{ kN/m}^2$$

1.3.4 Convert kN/m² → bar

$$P_{abs} = \frac{5.6034}{100} = 0.056 \text{ bar}$$

1.3.5 Final Answers

$$\boxed{P_{abs} = 42 \text{ mmHg} = 5.6034 \text{ kN/m}^2 = 0.056 \text{ bar}}$$

1.3.6 Code

```
P_atm_mmHg = 757
P_vac_mmHg = 715
MMHG_TO_KN_M2 = 0.133416
KNM2_TO_BAR = 1 / 100
P_abs_mmHg = P_atm_mmHg - P_vac_mmHg
P_abs_kNm2 = P_abs_mmHg * MMHG_TO_KN_M2
P_abs_bar = P_abs_kNm2 * KNM2_TO_BAR
print(f"Absolute Pressure = {P_abs_mmHg:.2f} mmHg")
print(f"Absolute Pressure = {P_abs_kNm2:.3f} kN/m²")
print(f"Absolute Pressure = {P_abs_bar:.4f} bar")
```

1.4 Oil Flow in Tubes

Oil flows full bore at a velocity of $V = 2$ m/s through 16 tubes of diameter $d = 30$ mm. Density of oil: $\rho = 0.85$ g/mL. Find **volume flow rate** (L/s) and **mass flow rate** (kg/min).

1.4.1 Cross-sectional area of one tube

$$A = \pi \frac{d^2}{4} = \pi \frac{0.03^2}{4} \approx 7.0686 \times 10^{-4}\ \text{m}^2$$

1.4.2 Total area and volume flow rate

$$A_{\text{total}} = 16 \cdot 7.0686 \times 10^{-4} \approx 0.01131\ \text{m}^2$$

$$\dot{v} = A_{\text{total}} \cdot V \approx 0.02262\ \text{m}^3/\text{s} \approx 22.62\ \text{L/s}$$

1.4.3 Mass flow rate

$$\dot{m} = \rho \cdot \dot{v} = 850 \cdot 0.02262 \approx 19.227\ \text{kg/s} \approx 1153.6\ \text{kg/min}$$

1.4.4 Final Answers

$$\text{Volume flow rate: } \dot{v} \approx 22.6\ \text{L/s}$$

$$\text{Mass flow rate: } \dot{m} \approx 1154\ \text{kg/min}$$

1.4.5 Code

```
import math
v = 2.0
N = 16
d = 0.03
rho = 0.85 * 1000
A = math.pi * d**2 / 4
A_total = N * A
v_dot_m3_s = A_total * v
v_dot_L_s = v_dot_m3_s * 1000
m_dot_kg_s = rho * v_dot_m3_s
m_dot_kg_min = m_dot_kg_s * 60
print(f"Volume flow rate: {v_dot_L_s:.2f} L/s")
print(f"Mass flow rate: {m_dot_kg_min:.2f} kg/min")
```

1.5 Gauge Pressure

An oil of specific gravity (relative density) SG = 0.8 is contained in a vessel to a depth of $h = 2$ m. Find the **gauge pressure** at this depth in kPa.

1.5.1 Gauge Pressure

$$P_g = \rho g h$$

where

ρ = density of fluid (kg/m^3)

g= acceleration due to gravity (9.81 m/s^2)

h = depth (m)

1.5.2 Compute the density of oil using specific gravity

Specific gravity is defined as

$$\text{SG} = \frac{\rho_{\text{oil}}}{\rho_{\text{water}}}$$

where $\rho_{\text{water}} = 1000$ kg/m^3. Thus,

$$\rho_{\text{oil}} = \text{SG} \times \rho_{\text{water}} = 0.8 \times 1000 = 800 \text{ kg/m}^3$$

1.5.3 Compute the gauge pressure

$$P_g = \rho g h = 800 \times 9.81 \times 2$$

$$P_g = 15696 \text{ Pa} \approx 15.7 \text{ kPa}$$

1.5.4 Answer

The **gauge pressure** at a depth of 2 m in the oil is:

$$\boxed{15.7 \text{ kPa}}$$

1.5.5 Code

```
# Gauge Pressure Calculation for Oil

# Given data
specific_gravity = 0.8  # SG of oil
depth_m = 2.0           # depth in meters
g = 9.81                # acceleration due to gravity in m/s²
rho_water = 1000        # density of water in kg/m³

# Compute density of oil
rho_oil = specific_gravity * rho_water

# Compute gauge pressure (Pa)
P_g_Pa = rho_oil * g * depth_m

# Convert to kPa
P_g_kPa = P_g_Pa / 1000

# Print results
print(f"Density of oil: {rho_oil:.1f} kg/m³")
print(f"Gauge pressure at {depth_m} m depth: {P_g_Pa:.1f} Pa ({P_g_kPa:.2f} kPa)")
```

1.6 Absolute Pressure from Manometer Reading

A water manometer shows a pressure in a vessel of 400 mm **below atmospheric pressure**. The atmospheric pressure is measured as 763 mmHg. Determine the **absolute pressure** in the vessel in kPa.

1.6.1 Relationship between absolute and gauge pressure

$$P_{\text{abs}} = P_{\text{atm}} + P_{\text{gauge}}$$

Since the manometer shows a pressure **below atmospheric**, the gauge pressure is negative:

$$P_{\text{gauge}} = -\rho_{\text{water}} g h$$

1.6.2 Convert atmospheric pressure to Pa using

$$P = \rho g h = 13{,}600 \cdot 9.81 \cdot 0.001 = 133.416 \text{ Pa}$$

So

$$P_{\text{atm}} = 763 \times 133.416 \approx 101,801 \text{ Pa} \approx 101.8 \text{ kPa}$$

1.6.3 Compute gauge pressure

Water column height:

$$h = 400 \text{ mm} = 0.4 \text{ m}$$

Density of water: $rho_{\text{water}} = 1000 \text{ kg/m}^3$, $g = 9.81 \text{ m/s}^2$

$$P_{\text{gauge}} = -\rho g h = -1000 \times 9.81 \times 0.4$$

$$P_{\text{gauge}} = -3924 \text{ Pa} \approx -3.92 \text{ kPa}$$

1.6.4 Compute absolute pressure

$$P_{\text{abs}} = P_{\text{atm}} + P_{\text{gauge}} \approx 101.8 - 3.92 \approx 97.9 \text{ kPa}$$

1.6.5 Answer

The **absolute pressure** in the vessel is:

$$\boxed{97.9 \text{ kPa}}$$

1.6.6 Code

```
# Absolute Pressure Calculation from Water Manometer

# Given data
h_mm = 400                # manometer reading in mm (below atmospheric)
atm_mmHg = 763            # atmospheric pressure in mmHg
rho_water = 1000          # density of water in kg/m³
g = 9.81                  # gravity in m/s²
mmHg_to_Pa = 133.416      # conversion factor

# Convert manometer height to meters
h_m = h_mm / 1000

# Convert atmospheric pressure to Pa
P_atm_Pa = atm_mmHg * mmHg_to_Pa

# Gauge pressure (negative because below atmospheric)
P_gauge_Pa = - rho_water * g * h_m

# Absolute pressure
```

```python
P_abs_Pa = P_atm_Pa + P_gauge_Pa

# Convert to kPa
P_abs_kPa = P_abs_Pa / 1000

# Print results
print(f"Atmospheric pressure: {P_atm_Pa:.1f} Pa ({P_atm_Pa/1000:.1f} kPa)")
print(f"Gauge pressure: {P_gauge_Pa:.1f} Pa ({P_gauge_Pa/1000:.2f} kPa)")
print(f"Absolute pressure in the vessel: {P_abs_Pa:.1f} Pa ({P_abs_kPa:.2f} kPa)")
```

Chapter 2

Heat and Work

2.1 Heat Required to Heat Steel

A steel block of mass $m = 5$ kg and specific heat capacity $c = 480$ J/kg $\cdot$ K is heated from $T_1 = 15°\text{C}$ to $T_2 = 100°\text{C}$. Determine the **heat required**.

2.1.1 The formula for heat

$$Q = mc\Delta T \tag{2.1}$$

where

$\Delta T = T_2 - T_1$ is the temperature change.

2.1.2 Compute the temperature change

$$\Delta T = T_2 - T_1 = 100 - 15 = 85 \text{ K}$$

2.1.3 Compute the heat required

$$Q = mc\Delta T = 5 \times 480 \times 85$$

$$Q = 204,000 \text{ J}$$

2.1.4 Answer

The **heat required** to raise the temperature of the steel is:

$$\boxed{204 \text{ kJ}}$$

2.1.5 Code

```
# Heat Required to Heat Steel

# Given data
mass = 5             # kg
specific_heat = 480 # J/kg·K
T_initial = 15      # °C
T_final = 100       # °C

# Temperature change
delta_T = T_final - T_initial

# Heat required (in J)
Q_J = mass * specific_heat * delta_T

# Convert to kJ
Q_kJ = Q_J / 1000

# Print results
print(f"Temperature change: {delta_T} K")
print(f"Heat required: {Q_J:.0f} J ({Q_kJ:.0f} kJ)")
```

```
# Interactive Heat Calculation

# Get user input
mass = float(input("Enter the mass of the object (kg): "))
specific_heat = float(input("Enter the specific heat capacity (J/kg·K): "))
T_initial = float(input("Enter the initial temperature (°C): "))
T_final = float(input("Enter the final temperature (°C): "))

# Calculate temperature change
delta_T = T_final - T_initial

# Calculate heat required
Q_J = mass * specific_heat * delta_T
Q_kJ = Q_J / 1000

# Display results
print("\nCalculation Results:")
print(f"Temperature change: {delta_T:.2f} K")
print(f"Heat required: {Q_J:.2f} J ({Q_kJ:.2f} kJ)")
```

2.2 Finding Specific Heat Capacity

A liquid of mass $m = 4$ kg is heated from $T_1 = 15^\circ$C to $T_2 = 100^\circ$C. The heat supplied is $Q = 714$ kJ. Determine the **specific heat capacity** c of the liquid.

2.2.1 Recall the formula for heat

$$Q = mc\Delta T$$

where $\Delta T = T_2 - T_1$.

2.2.2 Convert heat to joules

$$Q = 714 \text{ kJ} = 714 \times 1000 = 714,000 \text{ J}$$

2.2.3 Compute temperature change

$$\Delta T = T_2 - T_1 = 100 - 15 = 85 \text{ K}$$

2.2.4 Solve for specific heat capacity

$$c = \frac{Q}{m\Delta T} = \frac{714,000}{4 \times 85}$$

$$c = \frac{714,000}{340} \approx 2100 \text{ J/kg} \cdot \text{K}$$

2.2.5 Answer

The **specific heat capacity** of the liquid is:

$$\boxed{c \approx 2100 \text{ J/kg} \cdot \text{K}}$$

2.2.6 Code

```
# Specific Heat Capacity Calculation

# Given data
mass = 4            # kg
T_initial = 15      # °C
T_final = 100       # °C
Q_kJ = 714          # heat supplied in kJ

# Convert heat to joules
Q_J = Q_kJ * 1000
```

```
# Temperature change
delta_T = T_final - T_initial

# Calculate specific heat capacity
c = Q_J / (mass * delta_T)

# Print results
print(f"Temperature change: {delta_T} K")
print(f"Specific heat capacity: {c:.0f} J/kg·K")
```

```
# Interactive Specific Heat Capacity Calculator

# Get user input
mass = float(input("Enter the mass of the liquid (kg): "))
T_initial = float(input("Enter the initial temperature (°C): "))
T_final = float(input("Enter the final temperature (°C): "))
Q_kJ = float(input("Enter the heat supplied (kJ): "))

# Convert heat to joules
Q_J = Q_kJ * 1000

# Temperature change
delta_T = T_final - T_initial

# Calculate specific heat capacity
c = Q_J / (mass * delta_T)

# Display results
print("\nCalculation Results:")
print(f"Temperature change: {delta_T:.2f} K")
print(f"Specific heat capacity: {c:.2f} J/kg·K")
```

2.3 Work Done by Fluid in Cylinder

A fluid in a cylinder is at pressure $P = 700$ kPa. It is **expanded at constant pressure** from a volume of $V_1 = 0.28$ m^3 to $V_2 = 1.68$ m^3. Determine the **work done**.

2.3.1 The formula for work done at constant pressure

$$W = P\Delta V \tag{2.2}$$

where

$$\Delta V = V_2 - V_1$$

2.3.2 Compute the change in volume

$$\Delta V = 1.68 - 0.28 = 1.40 \text{ m}^3$$

2.3.3 Convert pressure to Pa

$$P = 700 \text{ kPa} = 700 \times 10^3 \text{ Pa} = 700,000 \text{ Pa}$$

2.3.4 Compute the work done

$$W = P\Delta V = 700,000 \times 1.40$$

$$W = 980,000 \text{ J} \approx 980 \text{ kJ}$$

2.3.5 Answer

The **work done** by the fluid during expansion is:

$$\boxed{980 \text{ kJ}}$$

2.3.6 Code

```
# Work Done by Fluid at Constant Pressure

# Given data
P_kPa = 700          # pressure in kPa
V1 = 0.28            # initial volume in m³
V2 = 1.68            # final volume in m³

# Convert pressure to Pa
P_Pa = P_kPa * 1000

# Compute change in volume
delta_V = V2 - V1

# Compute work done (J)
W_J = P_Pa * delta_V

# Convert to kJ
W_kJ = W_J / 1000

# Print results
print(f"Pressure: {P_Pa} Pa")
print(f"Change in volume: {delta_V:.2f} m³")
print(f"Work done: {W_J:.0f} J ({W_kJ:.0f} kJ)")
```

```
# Interactive Work Done Calculator

# User input
P_kPa = float(input("Enter the constant pressure (kPa): "))
V1 = float(input("Enter the initial volume (m³): "))
V2 = float(input("Enter the final volume (m³): "))

# Convert pressure to Pa
P_Pa = P_kPa * 1000

# Compute change in volume
delta_V = V2 - V1

# Compute work done (J)
W_J = P_Pa * delta_V

# Convert to kJ
W_kJ = W_J / 1000

# Display results
print("\nCalculation Results:")
print(f"Pressure: {P_Pa:.0f} Pa ({P_kPa:.0f} kPa)")
print(f"Change in volume: {delta_V:.2f} m³")
print(f"Work done: {W_J:.0f} J ({W_kJ:.2f} kJ)")
```

Chapter 3

Thermal Expansion

Thermal expansion is the tendency of materials to change their shape, area, and volume in response to a change in temperature. When most substances are heated, their particles move more vigorously and tend to occupy more space, leading to an increase in dimensions. Conversely, when substances are cooled, they generally contract. This phenomenon occurs in solids, liquids, and gases, although the degree and nature of expansion vary depending on the material's state and properties.

3.1 Linear Expansion

This occurs along a specific dimension or direction, primarily in long, narrow objects (like rods or beams). When the temperature of a solid object increases, its length expands by an amount proportional to its original length and the temperature change. The equation for linear expansion is:

$$\Delta L = \alpha L_0 \Delta T \tag{3.1}$$

where:

- ΔL is the change in length,
- α is the coefficient of linear expansion (unique to each material),
- L_0 is the original length, and
- ΔT is the temperature change.

3.2 Superficial Expansion

Applicable to two-dimensional surfaces, such as sheets or plates. Here, both length and width expand, leading to an increase in surface area. The formula for area expansion is:

$$\Delta A = 2\alpha A_0 \Delta T \tag{3.2}$$

where:

- ΔA is the change in area,
- A_0 is the initial area, and
- ΔT is the temperature change.

3.3 Volumetric Expansion

Relevant for three-dimensional objects (like solids, liquids, and gases). The volume of an object expands with temperature, especially in fluids where this effect is more pronounced. The formula is:

$$\Delta V = \beta V_0 \Delta T \tag{3.3}$$

where:

- ΔV is the change in volume,
- β is the coefficient of volumetric expansion, which is approximately three times the linear expansion coefficient for isotropic solids,
- V_0 is the initial volume, and
- ΔT is the temperature change.

3.4 Thermal Expansion of a Steel Pipeline

A steel section of pipeline is 75 m long when out of service at an ambient temperature of 20°C. In service, it transports steam at 203°C. Assuming the pipe is free to expand, find its length at the operating temperature.
(The coefficient of linear expansion for steel is $\alpha = 12 \times 10^{-6}$ /°C.)

Given:

$$\begin{aligned} L_0 &= 75 \text{ m}, \\ T_1 &= 20^\circ\text{C}, \\ T_2 &= 203^\circ\text{C}, \\ \alpha &= 12 \times 10^{-6} \text{ /}^\circ\text{C}. \end{aligned}$$

The temperature rise is:

$$\Delta T = T_2 - T_1 = 203 - 20 = 183^\circ\text{C}.$$

For linear expansion:

$$L = L_0(1 + \alpha \Delta T).$$

Substitute the values:

$$\begin{aligned} L &= 75\left(1 + 12 \times 10^{-6} \times 183\right) \\ &= 75\left(1 + 0.002196\right) \\ &= 75.1647 \text{ m}. \end{aligned}$$

Final Answer

$$\boxed{L = 75.165 \text{ m}}$$

Note: The pipe expands by:

$$\Delta L = L - L_0 = 0.165 \text{ m} = 165 \text{ mm}.$$

3.4.1 Code

```
# Linear Expansion of a Steel Pipeline
# ------------------------------------
# Given:
#   L0 = 75 m   (initial length)
#   T1 = 20 °C  (ambient temperature)
#   T2 = 203 °C (operating temperature)
#      = 12 × 10  /°C (coefficient of linear expansion for steel)
#
# Find:
#   Final length L at 203 °C.

# Given values
L0 = 75.0                  # m
T1 = 20.0                  # °C
T2 = 203.0                 # °C
alpha = 12e-6              # /°C

# Temperature change
delta_T = T2 - T1

# Final length using linear expansion formula
L = L0 * (1 + alpha * delta_T)
```

```
# Expansion amount
expansion = L - L0

# Display results
print(f"Initial Length (m): {L0:.2f}")
print(f"Final Length (m):   {L:.3f}")
print(f"Expansion (m):      {expansion:.3f}")
```

3.5 Thermal Expansion of a Brass Cube

If a solid brass cube measures **50 mm × 50 mm × 50 mm** at **10°C**, what volume will it occupy when heated to **78°C**?

Coefficient of linear expansion for brass, $\alpha = 18.4 \times 10^{-6}$ /°C.

Step 1: Given Data

$$\begin{aligned} L_0 &= 50 \text{ mm} = 0.05 \text{ m}, \\ T_1 &= 10°\text{C}, \\ T_2 &= 78°\text{C}, \\ \Delta T &= T_2 - T_1 = 68°\text{C}, \\ \alpha &= 18.4 \times 10^{-6} \,/°\text{C}. \end{aligned}$$

Step 2: Coefficient of Volumetric Expansion

$$\beta = 3\alpha = 3(18.4 \times 10^{-6}) = 55.2 \times 10^{-6} \,/°\text{C}.$$

Step 3: Initial and Final Volumes

Initial volume:

$$V_0 = L_0^3 = (0.05)^3 = 0.000125 \text{ m}^3.$$

Expanded volume:

$$V = V_0(1 + \beta \Delta T)$$

$$V = 0.000125[1 + (55.2 \times 10^{-6})(68)].$$

$$V = 0.000125(1 + 0.0037536) = 0.00012547 \text{ m}^3.$$

Step 4: Final Answer

$$\boxed{V = 0.00012547\ \text{m}^3}$$

Note: The multiplier from m^3 to mm^3 is 1×10 . The brass cube expands from **125000 mm^3** to **125469.20 mm^3** when heated from **10°C** to **78°C**.

3.5.1 Code

```
# Thermal Expansion of a Brass Cube
# --------------------------------
# Given:
#   L0 = 50 mm  (initial side length)
#   T1 = 10 °C  (initial temperature)
#   T2 = 78 °C  (final temperature)
#      = 18.4 × 10  /°C  (linear expansion coefficient for brass)
#
# Find:
#   Final volume V at 78 °C (in mm³).

# Given values
L0 = 50e-3                # convert mm to m
T1 = 10
T2 = 78
alpha = 18.4e-6           # /°C

# Temperature change
delta_T = T2 - T1

# Volumetric expansion coefficient
beta = 3 * alpha

# Initial and final volumes (in m³)
V0 = L0 ** 3
V = V0 * (1 + beta * delta_T)

# Convert to mm³ for output (1 m³ = 1e9 mm³)
V0_mm3 = V0 * 1e9
V_mm3 = V * 1e9
delta_V = V_mm3 - V0_mm3

# Display results
print(f"Initial Volume (mm³): {V0_mm3:.2f}")
print(f"Final Volume (mm³):   {V_mm3:.2f}")
print(f"Increase in Volume (mm³): {delta_V:.2f}")
```

Chapter 4

Heat Transfer

4.1 Conduction

Conduction transfers heat through **direct contact**, as faster-vibrating molecules pass energy to slower ones. In solids, heat moves from molecule to molecule, and between objects in contact, it flows from the hotter region to the cooler one until thermal equilibrium is reached.

An example of conduction is an iron bar with one end placed in a flame. The other end soon becomes hot as heat is conducted along the bar from molecule to molecule through the metal.

The rate of heat transfer by conduction through a solid is given by:

$$Q = \frac{k\,A\,t\,\Delta T}{s} \tag{4.1}$$

where:

- Q = heat transferred (J)
- k = thermal conductivity of the material (W/m · K)
- A = cross-sectional area through which heat flows (m^2)
- t = time of heat transfer (s)
- ΔT = temperature difference across the material (K or °C)
- s = thickness or length of the material through which heat is conducted (m)

This equation shows that heat conduction increases with greater thermal conductivity, larger surface area, longer time, and larger temperature difference but decreases as the material's thickness increases.

4.2 Temperature Difference Across a Heat Exchanger Endplate

The shell diameter of a large heat exchanger is 1.8 m. The flat endplate (cover) at one end is made of carbon steel, 45 mm thick, and is uninsulated. If the maximum allowable heat loss through the cover, to avoid insulation is 150 MJ/h, determine the temperature difference permitted across the endplate. (For steel, k = 55 W/m°C)

Given

- Shell diameter: (D = 1.8 m)
- Endplate thickness: (s = 45 mm = 0.045 m)
- Material: carbon steel, thermal conductivity: (k = 55 W/m · K)
- Maximum allowable heat loss: ($Q_{\max}$ = 150 MJ/h)
- Endplate is **uninsulated**

Calculate the **temperature difference** ΔT across the endplate to limit heat loss.

1. **Area of the circular endplate**

$$A = \pi \left(\frac{D}{2}\right)^2 = \pi \left(\frac{1.8}{2}\right)^2$$

$$A = \pi \times 0.9^2 = 2.5447 \text{ m}^2$$

2. **Convert heat loss to Watts**

$$Q_{\max} = 150 \text{ MJ/h} = 150 \times 10^6 \text{ J/h}$$

$$1 \text{ h} = 3600 \text{ s} \quad \Rightarrow \quad Q = \frac{150 \times 10^6}{3600} = 41666.7 \text{ W}$$

3. **Use law of conduction for a flat plate**

$$Q = \frac{kA\Delta T}{s} \quad \Rightarrow \quad \Delta T = \frac{Qs}{kA}$$

Substitute values:

$$\Delta T = \frac{41666.7 \times 0.045}{55 \times 2.5447}$$

$$\Delta T = \frac{1875.0}{139.96} = 13.39 \text{ K}$$

Result

$$\boxed{\Delta T = 13.4^\circ\text{C}}$$

Notes:

- The maximum allowable **temperature difference across the endplate** is 13.4°C to keep heat loss below 150 MJ/h.
- If the temperature difference exceeds this, insulation would be required.

4.2.1 Code

```
# Given data
D = 1.8                # diameter of endplate, m
L = 0.045              # thickness, m
k = 55                 # thermal conductivity, W/m.K
Q_MJ_per_h = 150       # maximum heat loss, MJ/h

# Convert heat loss to W
Q_W = Q_MJ_per_h * 1e6 / 3600  # W

# Area of the circular plate
import math
A = math.pi * (D/2)**2

# Temperature difference
delta_T = Q_W * L / (k * A)

# Display results
print(f"Endplate area: {A:.4f} m²")
print(f"Heat loss: {Q_W:.2f} W")
print(f"Temperature difference across the endplate: {delta_T:.2f} °C")
```

4.3 Convection Heat Transfer

Convection transfers heat through the movement of fluids (liquids or gases). When a fluid is heated, it expands, becomes less dense, and rises, while cooler, denser fluid moves in to replace it, creating a convection current that distributes heat.

Natural convection occurs without mechanical aid, whereas forced convection involves devices such as pumps or fans. When fluid movement is driven by a pump or fan, heat is transferred by **forced convection**. Examples include:

- A pump circulating hot water through a building's heating system,
- A fan forcing air through an automobile radiator, or
- A forced draft fan pushing hot gases through a boiler.

The **total heat transferred** between a solid surface and a moving fluid over a specified time is given by:

$$Q = h_A \, A \, t \, (T_s - T_f) \tag{4.2}$$

where:

- Q = total heat transferred (J)
- h_A = surface (convective) heat transfer coefficient ($\mathrm{W/m}^2 \cdot \mathrm{K}$)
- A = surface area of heat transfer (m^2)
- t = time during which heat transfer occurs (s), with $t = 1\,\mathrm{s}$ for unit time
- T_s = surface temperature of the solid (K or °C)
- T_f = temperature of the surrounding fluid (K or °C)

This equation describes how heat energy is transferred between a surface and a fluid over a unit time of 1 second, depending on the **temperature difference**, the **surface area**, the **time**, and the fluid's heat-carrying effectiveness, represented by the **convective heat transfer coefficient** (h_A)

4.4 Hot Metal Convection Heat Transfer

A hot metal plate measuring **1.2 m × 0.8 m** is exposed to air at **25°C**. The surface temperature of the plate is maintained at **85°C**. If the **convective heat transfer coefficient** (surface heat transfer coefficient) between the plate and air is $h_A = 18\,\mathrm{W/m}^2 \cdot {}^\circ\mathrm{C}$, calculate the **rate of heat loss by convection** from the entire plate surface.

Given:

- Plate dimensions: (1.2 m × 0.8 m)
- Plate surface area: $A = 1.2 \times 0.8 = 0.96\,\mathrm{m}^2$
- Surface temperature: $T_s = 85^\circ\mathrm{C}$
- Air temperature: $T_f = 25^\circ\mathrm{C}$
- Convection coefficient: $h_A = 18\,\mathrm{W/m}^2 \cdot {}^\circ\mathrm{C}$

Solution

The rate of convective heat loss is given by:

$$Q = h_A\,A\,(T_s - T_f)$$

Substituting the values:

$$Q = 18 \times 0.96 \times (85 - 25)$$

$$Q = 18 \times 0.96 \times 60$$

$$Q = 1036.8\,\mathrm{W}$$

Answer

$$\boxed{Q = 1036.8\,\mathrm{W}}$$

4.4.1 Code

```
# Given data
L = 1.2         # length of the plate (m)
W = 0.8         # width of the plate (m)
T_surface = 85 # surface temperature (°C)
T_air = 25      # air temperature (°C)
hA = 18         # convective heat transfer coefficient (W/m²·°C)

# Calculate area of the plate
A = L * W

# Calculate heat loss rate
Q = hA * A * (T_surface - T_air)

# Display results
print(f"Plate area: {A:.2f} m²")
print(f"Temperature difference: {T_surface - T_air:.1f} °C")
print(f"Convective heat loss: {Q:.2f} W")
```

4.5 Radiation

Radiation transfers heat through electromagnetic waves that travel in straight lines and can pass through a vacuum. When these waves strike a surface, they may be absorbed (increasing temperature), reflected, or transmitted. Dark, rough surfaces absorb more radiation, while shiny, smooth ones reflect it.

The heat transfer by radiation is given by:

$$Q = \sigma\,\varepsilon\,A\,t\,(T_1^4 - T_2^4) \tag{4.3}$$

where:

- Q is the **heat energy transferred by radiation** (in kilojoules, kJ).
- σ is the **Stefan–Boltzmann constant**, equal to 5.6703×10^{-11} kW $\cdot$ m^{-2} $\cdot$ K^{-4}.
- ε is the **emissivity** of the surface (dimensionless, between 0 and 1), indicating how efficiently the surface emits or absorbs radiation compared to a perfect blackbody.
- A is the **surface area** of the radiating body (in square meters, m^2).
- t is the **time** during which heat transfer occurs (in seconds, s).
- T_1 is the **absolute temperature of the radiating surface** (in kelvin, K).
- T_2 is the **absolute temperature of the surroundings** (in kelvin, K).

This equation expresses the **net radiant heat transfer** between two bodies at different temperatures, taking into account the emissivity of the surface, its area, and the duration of heat exchange.

Examples include heat from the sun reaching Earth and radiant heat in a boiler furnace.

In a steam boiler, radiation occurs in the furnace. Any heating surfaces that are directly exposed to the furnace will receive heat directly by radiation from the flame. These include the waterwalls and some generating tubes of a watertube boiler, radiant superheater tubes (located at the outlet of the furnace), and the furnace walls of a firetube boiler.

4.6 Radiant Heat from a Flat Circular Plate

A flat circular plate is **400 mm** in diameter. Calculate the theoretical quantity of heat radiated per hour when its temperature is **227 °C** and the temperature of its surrounds is **27 °C**.

Given:

- Stefan–Boltzmann constant: $\sigma = 5.6703 \times 10^{-11}$ kW m^{-2} K^{-4}
- Emissivity: $\varepsilon = 1$ (ideal black body)

1. **Convert temperatures to Kelvin**

$$T_1 = 227 + 273 = 500 \text{ K}, \quad T_2 = 27 + 273 = 300 \text{ K}$$

2. **Calculate the area of the circular plate**

$$A = \pi \left(\frac{D}{2}\right)^2 = \pi \left(\frac{0.4}{2}\right)^2 \approx 0.125664 \text{ m}^2$$

3. **Apply the formula**

$$Q = \sigma \times \varepsilon \times A \times t \times (T_1^4 - T_2^4)$$

Substituting the values:

$$Q = 5.6703 \times 10^{-11} \times 1 \times 0.125664 \times 3600 \times (500^4 - 300^4)$$
$$Q = 1395.46 \text{ kWh}$$

Result

$$\text{Plate area: } 0.125664 \text{ m}^2$$

$$\text{Emissivity: } \varepsilon = 1$$

$$\boxed{\text{Heat radiated (per hour): } 1395.46 \text{ kWh}}$$

Note: This represents the **theoretical maximum radiation** for a perfect black body. Real materials would radiate less depending on their emissivity $\varepsilon < 1$.

4.6.1 Code

```python
import math

# Given data
sigma = 5.6703e-11    # kW/m^2/K^4
epsilon = 1.0        # emissivity (black body)
T1 = 227 + 273   # K (plate temperature)
T2 = 27 + 273    # K (surroundings)
d = 400 / 1000     # m (diameter)
A = math.pi * (d / 2)**2  # area in m²
```

```
# Heat radiated per second (kW)
Q_dot = sigma * epsilon * A * (T1**4 - T2**4)

# Heat radiated per hour (kWh)
Q_hour = Q_dot * 3600

# Display results
print(f"Plate area: {A:.6f} m²")
print(f"Emissivity (theoretical): {epsilon}")
print(f"Heat radiated per second: {Q_dot:.4f} kW")
print(f"Heat radiated per hour: {Q_hour:.2f} kWh")
```

Chapter 5

Perfect Gases

The terms “perfect gas” and “ideal gas” are often used interchangeably in physics and engineering, but “ideal gas” is the more standard term in modern scientific literature.

An ideal gas is a theoretical model that remains in the gaseous state under all conditions, as it does not account for phase changes such as condensation. Real gases, such as nitrogen, oxygen, and dry air, are often approximated as ideal gases under typical conditions (e.g., low pressures and high temperatures relative to their critical points), but they may condense under extreme conditions. Saturated steam, being at the liquid-vapor phase boundary, does not behave as an ideal gas due to significant intermolecular forces, whereas superheated steam approximates ideal gas behavior when sufficiently far from its condensation point.

In an ideal gas, pressure, temperature, and volume are interrelated, enabling the calculation of changes in one property based on the others. These relationships are governed by Boyle’s Law, Charles’s Law, and the ideal gas law, which collectively describe the behavior of an ideal gas under varying conditions.

5.1 Boyle’s Law

Robert Boyle (1627–1691), a physicist, investigated the behavior of an ideal gas at constant temperature. By controlling the addition or removal of heat during changes in the volume and pressure of a confined gas, he maintained a constant temperature. He discovered that, under these conditions, the absolute pressure of a gas is inversely proportional to its volume. That is, as the volume increases, the pressure decreases, and conversely, as the volume decreases, the pressure increases. This can be stated as:\index{Boyle’s Law}.

$$P \propto \frac{1}{V} \tag{5.1}$$

5.2 Charles's Law

Charles's Law, named after Jacques Charles (1746–1823), a French physicist, states that for an ideal gas maintained at constant pressure, the volume of the gas is directly proportional to its absolute temperature (measured in Kelvin). This means that as the temperature of a gas increases, its volume increases proportionally, provided the pressure remains constant. Conversely, as the temperature decreases, the volume decreases. The law assumes the gas behaves as an ideal gas. This can be stated as:\index{Charles's Law}.

$$V \propto \ T \tag{5.2}$$

5.3 Boyle's Law — Pressure–Volume Relationship

Assuming compression according to the law $pV =$ constant:

1. Calculate the final volume when $1\,\mathrm{m}^3$ of gas at a pressure of $120\,\mathrm{kN/m}^2$ is compressed to a pressure of $960\ \mathrm{kN/m}^2$.
2. Calculate the initial volume of a gas at a pressure of 1.05 bar that will occupy a volume of $5.6\,\mathrm{m}^3$ when it is compressed to a pressure of 42 bar.

5.3.1 Given:

The compression (or expansion) follows the law:

$$p_1V_1 = p_2V_2$$

5.3.2 Final Volume of Gas

Given: $p_1 = 120 \ \ \mathrm{kN/m^2}, \quad V_1 = 1 \ \ \mathrm{m^3}, \quad p_2 = 960 \ \ \mathrm{kN/m^2}$

From $p_1V_1 = p_2V_2$:

$$V_2 = \frac{p_1V_1}{p_2}$$

Substitute:

$$V_2 = \frac{120 \times 1}{960} = 0.125\ \text{m}^3$$

Final Volume

$\boxed{V_2 = 0.125\ \text{m}^3}$

5.3.3 Initial Volume of Gas

Given: $p_1 = 1.05\ \ \text{bar}, \quad V_2 = 5.6\ \ \text{m}^3, \quad p_2 = 42\ \ \text{bar}$

From $p_1V_1 = p_2V_2$:

$$V_1 = \frac{p_2V_2}{p_1}$$

Substitute:

$$V_1 = \frac{42 \times 5.6}{1.05}$$

$$V_1 = 224\ \text{m}^3$$

Initial Volume

$\boxed{V_1 = 224\ \text{m}^3}$

5.3.4 Code

```
# Boyle's Law Calculator (p1 * V1 = p2 * V2)

print("Boyle’s Law Calculator: p V  = p V \n")

# Get known values (enter 0 for the unknown)
p1 = float(input("Enter initial pressure p  (in kN/m² or bar): "))
V1 = float(input("Enter initial volume V  (m³): "))
p2 = float(input("Enter final pressure p  (in kN/m² or bar): "))
V2 = float(input("Enter final volume V  (m³): "))

print()

# Determine which variable is missing (the one entered as 0)
if V2 == 0:
    V2 = (p1 * V1) / p2
    print(f"Final volume V  = {V2:.4f} m³")
```

```
elif V1 == 0:
    V1 = (p2 * V2) / p1
    print(f"Initial volume V  = {V1:.4f} m³")

elif p2 == 0:
    p2 = (p1 * V1) / V2
    print(f"Final pressure p  = {p2:.4f}")

elif p1 == 0:
    p1 = (p2 * V2) / V1
    print(f"Initial pressure p  = {p1:.4f}")

else:
    print("All variables entered - nothing to calculate!")

print("\nUnits:")
print("• If you used bar, results are in bar.")
print("• If you used kN/m², results are in kN/m².")
```

5.4 Compression — Stroke Volume and Final Temperature

A gas is compressed in a cylinder from $p_1 = 1$ bar and $_1 = 35^\circ$C at the beginning of the stroke to $p_2 = 37$ bar at the end of the stroke. The clearance volume is $V_c = 850$ cm^3. The compression index is $n = 1.32$. Find the stroke (swept) volume V_s and the temperature at the end of compression T_2.

5.4.1 Solution

For a polytropic compression with $pV^n =$ constant (here $n = 1.32$ and with clearance V_c:

Start of compression (BDC) volume:

$$V_1 = V_c + V_s$$

End of compression (TDC) volume:

$$V_2 = V_c$$

From $(p_1 V_1^n = p_2 V_2^n)$ we get

$$\left(\frac{V_2}{V_1}\right)^n = \frac{p_1}{p_2}$$

so

$$\frac{V_c}{V_c + V_s} = \left(\frac{p_1}{p_2}\right)^{1/n}.$$

Rearrange to solve for V_s:

$$V_s = V_c\left[\left(\frac{p_2}{p_1}\right)^{1/n} - 1\right].$$

Substitute numbers (convert V_c to m^3: 850 cm^3 $= 850 \times 10^{-6}$m^3 $= 0.00085$ m^3:

$$V_s = 0.00085\Big[\big(\tfrac{37}{1}\big)^{1/1.32} - 1\Big]$$

Numerical evaluation:

$$V_s \approx 0.012255 \text{ m}^3 = 12.26 \text{ L}.$$

5.4.2 Final temperature

For a polytropic process the temperature ratio is

$$\frac{T_2}{T_1} = \left(\frac{p_2}{p_1}\right)^{\frac{n-1}{n}}.$$

Convert T_1 to kelvin: $T_1 = 35^\circ\text{C} = 308.15$ K.

$$T_2 = 308.15\left(\frac{37}{1}\right)^{\frac{1.32-1}{1.32}}$$

Numerical evaluation:

$$T_2 \approx 739.49 \text{ K} = 466.34^\circ\text{C}.$$

5.4.3 Answers

- Stroke (swept) volume: $\boxed{V_s \approx 0.01226 \text{ m}^3 \text{ (12.26 L)}}$
- Temperature at end of compression: $\boxed{T_2 \approx 739.5 \text{ K} \;=\; 466.3^\circ\text{C}}$.

5.4.4 Code

```python
# Given data
p1 = 1       # bar
p2 = 37      # bar
Vc_cm3 = 850 # clearance volume in cm³
n = 1.32     # compression index
T1_C = 35    # °C

# Unit conversions
Vc = Vc_cm3 * 1e-6      # m³
T1 = T1_C + 273.15       # K

# Stroke volume calculation
Vs = Vc * ((p2 / p1)**(1/n) - 1)

# Final temperature calculation
T2 = T1 * (p2 / p1)**((n - 1) / n)
T2_C = T2 - 273.15

print(f"Stroke volume Vs = {Vs:.5f} m³ ({Vs*1000:.2f} L)")
print(f"Final temperature T2 = {T2:.2f} K ({T2_C:.2f} °C)")
```

Chapter 6

IC Engines

Internal combustion (IC) engines derive their name from fuel burning inside the engine cylinder. The combustion heats the air in the cylinder, increasing its pressure and moving the piston, whose reciprocating motion is converted to rotary motion by the crankshaft.

IC engines are classified by their ignition method:

- Compression Ignition: Air is highly compressed, raising its temperature so injected fuel ignites spontaneously.
- Spark Ignition: A fuel–air mixture is compressed and then ignited by a spark plug.

6.0.1 Engine Performance Calculation

An engine working on the constant volume cycle is 150 mm bore by 165 mm stroke, and its clearance volume is 0.5 litre. The fuel used has a calorific value of 45250 kJ/kg, and the consumption is 0.334 kg/kWh. The mechanical efficiency of the engine is 81%. Take = 1.4

Calculate:

1. The compression ratio
2. The ideal efficiency
3. The indicated thermal efficiency
4. The efficiency ratio.

Given

Bore = 150 mm = 0.150 m

Stroke = 165 mm = 0.165 m

Clearance volume $V_c = 0.5$ L $= 0.0005$ m^3

Calorific value $CV = 45\,250$ kJ/kg

Fuel consumption $= 0.334$ kg/kWh (assume **brake** specific fuel consumption)

Mechanical efficiency $\eta_m = 0.81$

Ratio of specific heats $\gamma = 1.4$

Find:

1. Compression ratio r
2. Ideal (Otto) thermal efficiency η_{ideal}
3. Indicated thermal efficiency η_{ind}
4. Efficiency ratio $= \dfrac{\eta_{\text{ind}}}{\eta_{\text{ideal}}}$

1. Compression ratio

Swept volume (displacement) of one cylinder:

$$V_d = \frac{\pi}{4}D^2 \times \text{stroke} = \frac{\pi}{4}(0.150)^2(0.165) = 0.00291579 \text{ m}^3.$$

Compression ratio

$$r = \frac{V_d + V_c}{V_c} = \frac{0.00291579 + 0.0005}{0.0005} = 6.8316.$$

2. Ideal (Otto) thermal efficiency

For a constant-volume (Otto) cycle:

$$\eta_{\text{ideal}} = 1 - \frac{1}{r^{\gamma-1}}.$$

With $r = 6.8316,\ \gamma = 1.4$,

$$\eta_{\text{ideal}} = 1 - r^{1-\gamma} = 1 - r^{-0.4} \approx 0.53635 = 53.635\%.$$

3. Indicated thermal efficiency

First calculate the brake (output) energy per kWh of output: 1 kWh=3600 kJ

Fuel energy input per kWh of brake output:

$$E_{\text{fuel}} = \text{SFOC} \times CV = 0.334 \text{ kg/kWh} \times 45\,250 \text{ kJ/kg} = 15\,113.5 \text{ kJ/kWh}.$$

For 1 kW break power, $E_{\text{fuel}} = 15\,113.5$ kJ/h.

$$E_{\text{fuel}} = \frac{15\,113.5}{3600} = 4.1982 \text{ kJ/s} = 4.1982 \text{ kW}$$

$$\eta_m = \frac{BP}{IP} = \frac{1}{IP} = 0.81\%.$$

$$IP = \frac{1}{0.81} = 1.2346 \text{ kW}$$

$$\eta_{\text{ind}} = \frac{IP}{E_{\text{fuel}}} = \frac{1.2346\ kW}{4.1982\ kW} = 29.4\%$$

4. Efficiency ratio

$$\text{Efficiency ratio} = \frac{\eta_{\text{ind}}}{\eta_{\text{ideal}}} = \frac{0.294}{0.5363} \approx 0.5482 \quad (54.82\%).$$

6.0.2 Final Answers

- **A. Compression ratio:** $\boxed{r \approx 6.832}$
- **B. Ideal (Otto) efficiency:** $\boxed{\eta_{\text{ideal}} \approx 0.5363 = 53.63\%}$
- **C. Indicated thermal efficiency:** $\boxed{\eta_{\text{ind}} \approx 0.2941 = 29.4\%}$
- **D. Efficiency ratio:** $\boxed{\frac{\eta_{\text{ind}}}{\eta_{\text{ideal}}} \approx 0.5482 = 54.82\%}$

Chapter 7

Ideal Cycles

7.1 Ideal Constant-Volume (Otto) Cycle

The Ideal Constant-Volume (Otto) Cycle, commonly referred to as the Otto cycle, is a thermodynamic cycle that models the operation of a spark-ignition internal combustion engine, such as those used in gasoline-powered automobiles. It is an idealized model that assumes the working fluid is an ideal gas and describes the sequence of processes that convert thermal energy into mechanical work.

7.1.1 P-V Diagram Representation

The Otto cycle on a P-V diagram forms a closed loop:

- 1-2: Steep upward curve (pressure increases, volume decreases, adiabatic compression).
- 2-3: Vertical line upward (pressure and temperature increase, volume constant, heat addition).
- 3-4: Downward curve (pressure decreases, volume increases, adiabatic expansion).
- 4-1: Vertical line downward (pressure and temperature decrease, volume constant, heat rejection).

7.2 Otto Cycle Example

In an ideal constant-volume (Otto) cycle the temperature at the beginning of compression is 50°C. The volumetric compression ratio is $r = 5 : 1$. The heat supplied during the cycle is $q_{in} = 930$ kJ/kg of working fluid. Take $\gamma = 1.4$ and $c_v = 0.717$ kJ/kg $\cdot$ K.

Calculate:

A. The maximum temperature attained in the cycle.
B. The work done during the cycle per kg of working fluid.
C. The ideal thermal efficiency of the cycle.

7.2.1 Theory and formulas

For an ideal Otto cycle (constant-volume heat addition) the relevant steps and formulas are:

1. Isentropic compression $1 \to 2$:
$$T_2 = T_1\, r^{\gamma-1}$$
2. Constant-volume heat addition $2 \to 3$:
$$q_{in} = c_v(T_3 - T_2) \quad \Rightarrow \quad T_3 = T_2 + \frac{q_{in}}{c_v}$$
3. Isentropic expansion $3 \to 4$:
$$T_4 = T_3\, r^{1-\gamma}$$
4. Heat rejected:
$$q_{out} = c_v(T_4 - T_1)$$
5. Net work per unit mass:
$$w = q_{in} - q_{out}$$
6. Thermal efficiency:
$$\eta = \frac{w}{q_{in}} = 1 - \frac{q_{out}}{q_{in}}$$

(For the ideal Otto cycle the formula reduces to the well-known expression $\eta = 1 - r^{1-\gamma}$, which provides a check.)

7.2.2 Code

```
# Otto-cycle calculation (per kg)

# Given data
r = 5.0                 # compression ratio
gamma = 1.4
cv = 0.717              # kJ/kg·K
T1_C = 50.0             # °C
q_in = 930.0            # kJ/kg
```

```
# Conversions
T1 = T1_C + 273  # K

# Step 1: T2 (after isentropic compression)
T2 = T1 * r**(gamma - 1)

# Step 2: T3 (after constant-volume heat addition) -> Tmax
T3 = T2 + q_in / cv

# Step 3: T4 (after isentropic expansion)
T4 = T3 * r**(1 - gamma)

# Heat rejected
q_out = cv * (T4 - T1)

# Net work per kg
w_net = q_in - q_out

# Efficiency
eta = w_net / q_in

# Print results
print(f"T1 = {T1:.2f} K ({T1 - 273:.2f} °C)")
print(f"T2 = {T2:.2f} K ({T2 - 273:.2f} °C)")
print(f"T3 = {T3:.2f} K ({T3 - 273:.2f} °C)  <-- Tmax")
print(f"T4 = {T4:.2f} K ({T4 - 273:.2f} °C)")
print()
print(f"q_in = {q_in:.2f} kJ/kg")
print(f"q_out = {q_out:.2f} kJ/kg")
print(f"w_net = {w_net:.2f} kJ/kg")
print(f"efficiency = {eta*100:.2f} %")
```

7.2.3 Using Actual Volume Ratios Instead of Direct r

In the ideal Otto cycle we have written:

$$T_2 = T_1 r^{\gamma-1},$$

with $r = V_1/V_2$. Here we make the volume-dependence explicit using V_1 and V_2.

7.2.4 Given

- $T_1 = 50^{\circ}\text{C} = 323$ K

- Volumetric compression ratio $r = \dfrac{V_1}{V_2} = 5$ (we'll set $V_1 = 5,\ V_2 = 1$ so $V_1/V_2 = 5$
- Heat supplied $q_{in} = 930$ kJ/kg
- $\gamma = 1.4,\ c_v = 0.717$ kJ/kg $\cdot$ K

7.2.5 Volume-explicit relations

Isentropic compression (1→2) (using volumes):

$$\frac{T_2}{T_1} = \left(\frac{V_1}{V_2}\right)^{\gamma-1} \quad \Rightarrow \quad T_2 = T_1 \left(\frac{V_1}{V_2}\right)^{\gamma-1}.$$

Constant-volume heat addition (2→3):

$$T_3 = T_2 + \frac{q_{in}}{c_v}.$$

Isentropic expansion (3→4) (volume ratio inverted):

$$\frac{T_4}{T_3} = \left(\frac{V_2}{V_1}\right)^{\gamma-1} \quad \Rightarrow \quad T_4 = T_3 \left(\frac{V_2}{V_1}\right)^{\gamma-1}.$$

Heat rejected:

$$q_{out} = c_v(T_4 - T_1),$$

Net work and efficiency:

$$w = q_{in} - q_{out}, \qquad \eta = \frac{w}{q_{in}}.$$

7.2.6 Code

```
# Otto-cycle calculation using explicit volumes (per kg)

# Given data
V1 = 5.0        # arbitrary volume units (e.g. litres) at start of compression (BDC)
V2 = 1.0        # arbitrary volume units at end of compression (TDC)
gamma = 1.4
cv = 0.717      # kJ/kg·K
T1_C = 50.0     # °C
q_in = 930.0    # kJ/kg

# Conversions
T1 = T1_C + 273  # K
```

```
# Isentropic compression using volume ratio V1/V2
T2 = T1 * (V1 / V2)**(gamma - 1)

# Constant-volume heat addition
T3 = T2 + q_in / cv

# Isentropic expansion using inverted volume ratio V2/V1
T4 = T3 * (V2 / V1)**(gamma - 1)

# Heat rejected, net work and efficiency
q_out = cv * (T4 - T1)
w_net = q_in - q_out
eta = w_net / q_in

# Print results
print("--- Inputs ---")
print(f"V1 = {V1:.3f} (units), V2 = {V2:.3f} (units)  --> V1/V2 = {V1/V2:.3f}")
print(f"T1 = {T1:.2f} K ({T1-273:.2f} °C)")
print()
print("--- Results ---")
print(f"T2 = {T2:.2f} K ({T2-273:.2f} °C)")
print(f"T3 = {T3:.2f} K ({T3-273:.2f} °C)  <-- Tmax")
print(f"T4 = {T4:.2f} K ({T4-273:.2f} °C)")
print()
print(f"q_in = {q_in:.2f} kJ/kg")
print(f"q_out = {q_out:.2f} kJ/kg")
print(f"w_net = {w_net:.2f} kJ/kg")
print(f"efficiency = {eta*100:.2f} %")
```

7.3 Ideal Constant-Pressure (Diesel) Cycle

The Diesel cycle is a thermodynamic model of a compression-ignition engine, like those in marine diesel engines. Unlike the Otto cycle (constant-volume heat addition), the Diesel cycle assumes heat is added at constant pressure, reflecting the actual combustion in diesel engines.

The Diesel cycle consists of four thermodynamic processes—two isentropic (adiabatic and reversible), one constant-pressure, and constant-volume that describe the idealized behavior of a gas undergoing compression, combustion, expansion, and exhaust in a compression-ignition engine.

7.3.1 P-V Diagram Representation

The Diesel cycle on a P-V diagram forms a closed loop:

- 1-2: Steep upward curve (pressure increases, volume decreases, adiabatic compression).
- 2-3: Horizontal line (pressure constant, volume increases, constant-pressure heat addition).
- 3-4: Downward curve (pressure decreases, volume increases, adiabatic expansion).
- 4-1: Vertical line downward (pressure and temperature decrease, volume constant, heat rejection).

7.4 Diesel Cycle Example

Given

- $T_1 = 50^\circ\text{C} = 323$ K
- Compression ratio $r = \dfrac{V_1}{V_2} = 5$
- Heat supplied $q_{in} = 930$ kJ/kg (added at constant pressure)
- $\gamma = 1.4, \quad c_v = 0.717$ kJ/kg $\cdot$ K

Notes: For constant-pressure heat addition (2$\to$3),

$$q_{in} = c_p\,(T_3 - T_2), \qquad c_p = c_v\gamma.$$

The cut-off (volume) ratio is $\rho = \dfrac{V_3}{V_2} = \dfrac{T_3}{T_2}$ because $p_2 = p_3$

7.4.1 Code

```
# Diesel-cycle calculation (per kg)

import numpy as np
import matplotlib.pyplot as plt

# Given
r = 5.0                    # V1 / V2
gamma = 1.4
cv = 0.717                 # kJ/kg·K
cp = cv * gamma            # kJ/kg·K
T1 = 323.0                 # K (simplified 273 + 50)
q_in = 930.0               # kJ/kg

# State 2: isentropic compression 1->2
T2 = T1 * r**(gamma - 1)
```

```
# State 3: constant-pressure heat addition 2->3
# q_in = cp * (T3 - T2)  ->  T3 = T2 + q_in / cp
T3 = T2 + q_in / cp

# Cut-off ratio (V3/V2) since p2 = p3
rho = T3 / T2

# State 4: isentropic expansion 3->4 (V4 = V1)
# T4 = T3 * (V3/V4)^(gamma-1) = T3 * (rho / r)^(gamma-1)
T4 = T3 * (rho / r)**(gamma - 1)

# Heat rejected, net work and efficiency
q_out = cv * (T4 - T1)
w_net = q_in - q_out
eta = w_net / q_in

# Pressures (normalized) using ideal-gas relation P   T/V
P1 = 1.0
P2 = P1 * (T2/T1) * (V1 := r) / (V2 := 1.0)   # V1=r, V2=1 chosen for explicitness
P3 = P2
V3 = rho * V2
P4 = (T4 / T1) * (V1 / (V1)) * P1  # alternatively compute via ideal gas: P4 = P3 * (T4/T3) * (V3

# More consistent P4 via ideal gas:
P4 = P3 * (T4 / T3) * (V3 / V1)

# Curves for PV diagram
V_compression = np.linspace(V1, V2, 200)  # 1 -> 2 (compression)
P_compression = P1 * (V1 / V_compression)**gamma

V_expansion = np.linspace(V3, V1, 200)    # 3 -> 4 (expansion)
P_expansion = P3 * (V3 / V_expansion)**gamma

plt.figure(figsize=(8,5))
plt.plot(V_compression, P_compression, label="Isentropic compression (1→2)")
plt.plot([V2, V3], [P2, P3], 'r-', linewidth=2, label="Constant-pressure heat addition (2→3)")
plt.plot(V_expansion, P_expansion, label="Isentropic expansion (3→4)")
plt.plot([V1, V1], [P4, P1], 'r-', linewidth=2, label="Constant-volume heat rejection (4→1)")
# mark state points
plt.plot([V1, V2, V3, V1], [P1, P2, P3, P4], 'ko')
for (V,P,label) in [(V1,P1,"1"), (V2,P2,"2"), (V3,P3,"3"), (V1,P4,"4")]:
    plt.text(V, P*1.02, label, ha='center')

plt.xlabel("Volume (arbitrary units)")
plt.ylabel("Pressure (arbitrary units)")
plt.title("Ideal Diesel Cycle - P-V Diagram (normalized)")
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()

# Print numeric results
print("---- Diesel Cycle Results (per kg) ----")
print(f"T1 = {T1:.2f} K")
print(f"T2 = {T2:.2f} K")
print(f"T3 = {T3:.2f} K  (Tmax after constant-pressure heat addition)")
print(f"T4 = {T4:.2f} K")
print()
print(f"Cut-off ratio (rho = V3/V2) = {rho:.4f}")
print(f"q_in = {q_in:.2f} kJ/kg")
print(f"q_out = {q_out:.2f} kJ/kg")
print(f"Net work w = {w_net:.2f} kJ/kg")
print(f"Ideal thermal efficiency = {eta*100:.2f} %")
```

7.5 Dual Combustion Cycle

The Dual Combustion cycle, or mixed cycle, models engines that combine constant-volume (Otto cycle) and constant-pressure (Diesel cycle) combustion. Fuel burns in two phases: a fast initial phase at constant volume, then a slower phase at constant pressure, providing a realistic approximation of modern diesel engine combustion.

7.5.1 Efficiency of the Dual Combustion Cycle

The thermal efficiency (η) of the ideal Dual Combustion Cycle is defined as the ratio of net work output to total heat input:

$\eta = \frac{W_{\text{net}}}{Q_{\text{in}}} = \frac{Q_{\text{in},1}+Q_{\text{in},2}-Q_{\text{out}}}{Q_{\text{in},1}+Q_{\text{in},2}}$

Substituting the heat terms:

$Q_{\text{in},1} = mC_v(T_3 - T_2)$

$Q_{\text{in},2} = mC_p(T_4 - T_3)$

$Q_{\text{out}} = mC_v(T_5 - T_1)$

$\eta = 1 - \frac{Q_{\text{out}}}{Q_{\text{in},1}+Q_{\text{in},2}} = 1 - \frac{C_v(T_5-T_1)}{C_v(T_3-T_2)+C_p(T_4-T_3)}$

$$\eta = 1 - \frac{(T_5 - T_1)}{(T_3 - T_2) + \gamma(T_4 - T_3)} \tag{7.1}$$

7.5.2 P-V Diagram Representation

The Dual Combustion Cycle on a P-V diagram forms a closed loop:

- 1-2: Steep upward curve (pressure increases, volume decreases, adiabatic compression).
- 2-3: Vertical line upward (pressure and temperature increase, volume constant, heat addition).
- 3-4: Horizontal line (pressure constant, volume increases, heat addition).
- 4-5: Downward curve (pressure decreases, volume increases, adiabatic expansion).
- 5-1: Vertical line downward (pressure and temperature decrease, volume constant, heat rejection).

Appendix A

Script Template

Please use the following template to write your code or build your own template that suits your purpose. Adhering to these guidelines will ensure your work is professional and readable. As Python creator Guido van Rossum once noted, code is read far more frequently than it is written.

```python
"""
Please replace this text with a concise description of the script’s purpose.
For example, you can type in:
Determines the volumetric and mass flow rates in a pipe carrying oil,
given the internal diameter, fluid velocity, and relative density.
"""

import math

# Organize code with consistent sectioning and
# clear headers or separators to enhance readability

# ==============================
# INPUT PARAMETERS
# ==============================

pipe_internal_diameter_mm = 75.0  # Use descriptive variable names with units
oil_velocity_m_per_s = 1.2  # Include meaningful inline comments
oil_relative_density = 0.9  # Include meaningful inline comments
water_density_kg_per_m3 = 1000.0  # Density of water in kg/m³

# ==============================
# UNIT CONVERSIONS AND GEOMETRY
# ==============================
# Convert pipe diameter from millimeters to meters
pipe_internal_diameter_m = pipe_internal_diameter_mm / 1000.0

# Calculate pipe radius
pipe_radius_m = pipe_internal_diameter_m / 2.0

# Calculate cross-sectional area of the pipe (circular)
pipe_cross_sectional_area_m2 = math.pi * pipe_radius_m**2

# ==============================
# FLOW RATE CALCULATIONS
# ==============================

# Calculate volumetric flow rate
# Restrict line length to a maximum of 79 characters
# to ensure compatibility with print media and prevent horizontal overflow.
volumetric_flow_rate_m3_per_s = (
    pipe_cross_sectional_area_m2 * oil_velocity_m_per_s
)

# Calculate oil density from relative density
oil_density_kg_per_m3 = oil_relative_density * water_density_kg_per_m3

# Calculate mass flow rate
mass_flow_rate_kg_per_s = oil_density_kg_per_m3 * volumetric_flow_rate_m3_per_s

# Convert mass flow rate to tonnes per hour
mass_flow_rate_tonnes_per_hour = mass_flow_rate_kg_per_s * 3600 / 1000

# ==============================
# OUTPUT RESULTS
# ==============================

print("---Pipe Flow Calculation Results---")
print(f"Volumetric flow rate : {volumetric_flow_rate_m3_per_s:.4f} m³/s")
print(
    f"Mass flow rate       : {mass_flow_rate_tonnes_per_hour:.4f} "
    f"tonnes/hour"
)
```

Appendix B

Unicode Math Symbols

B.1 Python Script Comment Snippets

```
Area = 25 m²
Volume = 3.2 m³
Density of water = 1000 kg/m³
Angle = 45°
Angular speed = 12 s ¹

Area =  ·r²

Law of sines:
T  / sin( ) = T  / sin( ) = W / sin( )

Law of cosines:
cos( ) = (a² + b² − c²) / (2ab)
cos( ) = (J² + T² − P²) / (2·J·T)

Energy equation:
P  + ½ v ² +  gz  = P  + ½ v ² +  gz
```

B.2 Math Symbols

Table B.1: Math Symbols

Symbol	Name	SI Units
√	Square root	—
	Summation	—

Symbol	Name	SI Units
	Integral	—
∞	Infinity	—
	Less than or equal	—
	Greater than or equal	—
	Not equal	—
	Perpendicular	—
∅	Diameter	m

B.3 Kinematics & Dynamics

Table B.2: Kinematics & Dynamics

Symbol	Meaning	SI Units
	Angular velocity	$\mathrm{rad} \cdot \mathrm{s}^{1}$
	Angular acceleration	$\mathrm{rad} \cdot \mathrm{s}^{2}$
	Angular position	rad

B.4 Fluids

Table B.3: Fluids

Symbol	Meaning	SI Units
	Density	$\mathrm{kg} \cdot \mathrm{m}^{3}$
	Dynamic viscosity	$\mathrm{Pa} \cdot \mathrm{s}$
	Kinematic viscosity	$\mathrm{m}^2 \cdot \mathrm{s}^{1}$
p	Pressure	Pa
h	Head	m
$\dot{m}$	Mass flow rate	$\mathrm{kg} \cdot \mathrm{s}^{1}$
$\dot{v}$	Volumetric flow rate	$\mathrm{m}^3 \cdot \mathrm{s}^{1}$

B.5 Stress & Material Properties

Table B.4: Stress & Material Properties

Symbol	Meaning	SI Units
	Normal stress	Pa
	Shear stress	Pa

Symbol	Meaning	SI Units
	Normal strain	—
	Shear strain	—
E	Young's modulus	Pa
G	Shear modulus	Pa
	Poisson's ratio	—

B.6 Greek Letters

Table B.5: Unicode Greek Letters

Lower Case	Upper Case	Name
	Α	Alpha
	Β	Beta
	Γ	Gamma
	Δ	Delta
	Ε	Epsilon
	Ζ	Zeta
	Η	Eta
	Θ	Theta
	Ι	Iota
	Κ	Kappa
	Λ	Lambda
	Μ	Mu
	Ν	Nu
	Ξ	Xi
	Ο	Omicron
	Π	Pi
	Ρ	Rho
	Σ	Sigma
	Τ	Tau
	Υ	Upsilon
	Φ	Phi
	Χ	Chi
	Ψ	Psi
	Ω	Omega

Appendix C

Greek Letters

The following tables present the names of Greek letters and selected symbols commonly used in engineering courses, ensuring precise reference and avoiding reliance on informal descriptors such as “squiggle.”

Table C.1: Greek letters.

Lower Case	Upper Case	Name
α	A	alpha
β	B	beta
γ	Γ	gamma
δ	Δ	delta
ϵ	E	epsilon
ζ	Z	zeta
η	E	eta
θ	Θ	theta
ι	I	iota
κ	K	kappa
λ	Λ	lambda
μ	M	mu
ν	N	nu
ξ	Ξ	xi
o	O	omicron
π	Π	pi
ρ	P	rho
σ	Σ	sigma
τ	T	tau
υ	Υ	upsilon
ϕ	Φ	phi
χ	X	chi

Lower Case	Upper Case	Name
ψ	Ψ	psi
ω	Ω	omega

Table C.2: Commonly used symbols in engineering courses.

Symbol	Name	Use	Course
Δ	Delta	Change	Thermodynamics
Δ	Delta	Displacement	Naval Architecture
∇	Nabla	Volume	Naval Architecture
Σ	Sigma	Sum	Thermodynamics, Naval Architecture, Applied Mechanics
σ	Sigma	Stress	Thermodynamics, Applied Mechanics
ϵ	Epsilon	Modulus of elasticity	Thermodynamics, Applied Mechanics
η	Eta	Efficiency	Thermodynamics
μ	Mu	Friction	Thermodynamics, Applied Mechanics
ω	Omega	Angular velocity	Thermodynamics, Applied Mechanics
ρ	Rho	Density	Thermodynamics, Naval Architecture
τ	Tau	Torque	Thermodynamics, Applied Mechanics

Bird, J. O. (2021). *Bird's engineering mathematics* (Ninth edition). Routledge.

Bolton, W. (2021). *Engineering science* (Seventh edition). Routledge.

Hannah, J., & Hillier, M. J. (1995). *Applied mechanics* (3rd edition). Longman Pub Group.

Russell, P. A., Embleton, W., & Jackson, L. (2022). *Applied thermodynamics for marine engineers* (6th edition). Reeds.

Russell, P. A., Jackson, L., & Embleton, W. (2021). *Applied mechanics for marine engineers* (7th edition). Reeds.

Shaw, Z. (2017). *Learn python 3 the hard way: A very simple introduction to the terrifyingly beautiful world of computers and code* (4th edition). Addison-Wesley Professional.

Sweigart, A. (2020). *Automate the boring stuff with python, 2nd edition: Practical programming for total beginners.* No Starch Press.

Sweigart, A. (2021). *Beyond the basic stuff with python: Best practices for writing clean code.* No Starch Press.

Index