

▼ Chapter 1 Exercise

Exercise 1

```
def print_odd_numbers():  
    for number in range(101, 200, 2):  
        print(number)
```

```
print_odd_numbers()
```

```
↔ 101  
103  
105  
107  
109  
111  
113  
115  
117  
119  
121  
123  
125  
127  
129  
131  
133  
135  
137  
139  
141  
143  
145  
147  
149  
151  
153  
155  
157  
159  
161  
163  
165  
167  
169  
171  
173  
175  
177  
179  
181  
183  
185  
187  
189  
191  
193  
195  
197  
199
```

Exercise 2

```

def is_prime(n):
    """Check if a number is prime."""
    if n <= 1:
        return False
    for i in range(2, int(n ** 0.5) + 1):
        if n % i == 0:
            return False
    return True

def find_primes_in_range(start, end):
    """Find all prime numbers in a given range."""
    primes = []
    for num in range(start, end + 1):
        if is_prime(num):
            primes.append(num)
    return primes

# Define the range
start = 1000
end = 1500

# Find and display prime numbers in the range
prime_numbers = find_primes_in_range(start, end)
print(f"Prime numbers from {start} to {end}:")
for prime in prime_numbers:
    print(prime)

```

↩ Prime numbers from 1000 to 1500:

```

1009
1013
1019
1021
1031
1033
1039
1049
1051
1061
1063
1069
1087
1091
1093
1097
1103
1109
1117
1123
1129
1151
1153
1163
1171
1181
1187
1193
1201
1213
1217
1223
1229
1231
1237
1249
1259
1277
1279
1283
1289
1291
1297
1301
1303
1307
1319
1321
1327
1361
1367
1373
1381
1399

```

1409
1423
1427

Exercise 3

```
def find_smallest(num1, num2, num3):
    """Return the smallest of the three numbers."""
    return min(num1, num2, num3)

# Main program
try:
    # Taking input from the user
    number1 = float(input("Enter the first number: "))
    number2 = float(input("Enter the second number: "))
    number3 = float(input("Enter the third number: "))

    # Finding the smallest number
    smallest = find_smallest(number1, number2, number3)

    # Printing the result
    print(f"The smallest number among {number1}, {number2}, and {number3} is {smallest}.")
except ValueError:
    print("Please enter valid numbers.")
```

Enter the first number: 3
Enter the second number: 1
Enter the third number: 5
The smallest number among 3.0, 1.0, and 5.0 is 1.0.

Exercise 4

```
# Given list
Ten = ["92", "54", "23", "10", "44", "05", "67", "76", "34", "82"]

# Print the original list
print("Original List:", Ten)

# Create a new list which has the first four numbers
first_four = Ten[:4]
print("First Four Numbers:", first_four)

# Create a new list which has the last four numbers
last_four = Ten[-4:]
print("Last Four Numbers:", last_four)

# Create a new list which has numbers between 44 to 82
numbers_between_44_and_82 = [num for num in Ten if 44 <= int(num) <= 82]
print("Numbers Between 44 and 82:", numbers_between_44_and_82)

# Create a new list which has numbers at odd places
numbers_at_odd_places = Ten[::2]
print("Numbers at Odd Places:", numbers_at_odd_places)

# Create a new list which has numbers at even places
numbers_at_even_places = Ten[1::2]
print("Numbers at Even Places:", numbers_at_even_places)

# Create a new list which is the reverse of the list Ten
reversed_list = Ten[::-1]
print("Reversed List:", reversed_list)

# Create a new list which has the first two and last two numbers
first_two_and_last_two = Ten[:2] + Ten[-2:]
print("First Two and Last Two Numbers:", first_two_and_last_two)
```

Original List: ['92', '54', '23', '10', '44', '05', '67', '76', '34', '82']
First Four Numbers: ['92', '54', '23', '10']
Last Four Numbers: ['67', '76', '34', '82']
Numbers Between 44 and 82: ['54', '44', '67', '76', '82']
Numbers at Odd Places: ['92', '23', '44', '67', '34']
Numbers at Even Places: ['54', '10', '05', '76', '82']
Reversed List: ['82', '34', '76', '67', '05', '44', '10', '23', '54', '92']
First Two and Last Two Numbers: ['92', '54', '34', '82']

Exercise 5

```
import pandas as pd

# Data
students = ["student1", "student2", "student3", "student4", "student5", "student6", "student7"]
maths = [82, 80, 45, 36, 94, 59, 60]
science = [47, 33, 59, 62, 61, 64, 33]
english = [39, 32, 90, 29, 87, 75, 91]
hindi = [30, 63, 51, 57, 36, 59, 52]

# Create a dataframe
df = pd.DataFrame({
    'Name': students,
    'Maths': maths,
    'Science': science,
    'English': english,
    'Hindi': hindi
})

# Calculate total and percentage
df['Total'] = df[['Maths', 'Science', 'English', 'Hindi']].sum(axis=1)
df['Percentage'] = df['Total'] / 4

# Add Result column
df['Result'] = df['Percentage'].apply(lambda x: 'PASS' if x >= 40 else 'FAIL')

# Add Division column
def get_division(percentage):
    if percentage >= 60:
        return 'FIRST'
    elif 50 <= percentage < 60:
        return 'SECOND'
    elif 40 <= percentage < 50:
        return 'THIRD'
    else:
        return 'FAIL'

df['Division'] = df['Percentage'].apply(get_division)

# Sort the dataframe by Division
sorted_by_division = df.sort_values(by='Division')
print("DataFrame sorted by Division:")
print(sorted_by_division)

# Sort the dataframe by Total in ascending order
sorted_by_total = df.sort_values(by='Total')
print("\nDataFrame sorted by Total (ascending):")
print(sorted_by_total)

# Filter students with Percentage equal to 60
students_with_60_percentage = df[df['Percentage'] == 60]
print("\nStudents with 60% marks:")
print(students_with_60_percentage)

# Filter students with THIRD Division
third_division_students = df[df['Division'] == 'THIRD']
print("\nStudents with THIRD Division:")
print(third_division_students)

# Extract only the Division column
division_column = df[['Division']]
print("\nDivision column:")
print(division_column)

# Extract only the third row
third_row = df.iloc[2]
print("\nThird row:")
print(third_row)

# Create a new dataframe with Name and Division only
name_and_division_df = df[['Name', 'Division']]
print("\nNew DataFrame with Name and Division only:")
print(name_and_division_df)
```

```

4 student5 94 61 87 36 278 69.50 PASS FIRST
5 student6 59 64 75 59 257 64.25 PASS FIRST
1 student2 80 33 32 63 208 52.00 PASS SECOND
6 student7 60 33 91 52 236 59.00 PASS SECOND
0 student1 82 47 39 30 198 49.50 PASS THIRD
3 student4 36 62 29 57 184 46.00 PASS THIRD

```

DataFrame sorted by Total (ascending):

```

      Name Maths Science English Hindi Total Percentage Result Division
3 student4 36 62 29 57 184 46.00 PASS THIRD
0 student1 82 47 39 30 198 49.50 PASS THIRD
1 student2 80 33 32 63 208 52.00 PASS SECOND
6 student7 60 33 91 52 236 59.00 PASS SECOND
2 student3 45 59 90 51 245 61.25 PASS FIRST
5 student6 59 64 75 59 257 64.25 PASS FIRST
4 student5 94 61 87 36 278 69.50 PASS FIRST

```

Students with 60% marks:

Empty DataFrame

Columns: [Name, Maths, Science, English, Hindi, Total, Percentage, Result, Division]

Index: []

Students with THIRD Division:

```

      Name Maths Science English Hindi Total Percentage Result Division
0 student1 82 47 39 30 198 49.5 PASS THIRD
3 student4 36 62 29 57 184 46.0 PASS THIRD

```

Division column:

```

Division
0 THIRD
1 SECOND
2 FIRST
3 THIRD
4 FIRST
5 FIRST
6 SECOND

```

Third row:

```

Name      student3
Maths      45
Science    59
English    90
Hindi      51
Total     245
Percentage 61.25
Result     PASS
Division   FIRST
Name: 2, dtype: object

```

New DataFrame with Name and Division only:

```

      Name Division
0 student1 THIRD
1 student2 SECOND
2 student3 FIRST
3 student4 THIRD
4 student5 FIRST
5 student6 FIRST
6 student7 SECOND

```

Exercise 6

```

import statistics as st

# Given list of numbers
numbers = [17, 43, 33, 61, 62, 15, 26, 17, 28, 49]

# Use lambda function to identify ODD numbers
odd_numbers = list(filter(lambda x: x % 2 != 0, numbers))
print("Odd numbers:", odd_numbers)

# Use lambda function to calculate the cube of these numbers
cubed_numbers = list(map(lambda x: x ** 3, numbers))
print("Cubed numbers:", cubed_numbers)

# Calculate mean, median, mode, standard deviation, and variance
mean = st.mean(numbers)
median = st.median(numbers)
mode = st.mode(numbers)
stdev = st.stdev(numbers)
variance = st.variance(numbers)

print(f"Mean: {mean}")
print(f"Median: {median}")
print(f"Mode: {mode}")
print(f"Standard Deviation: {stdev}")
print(f"Variance: {variance}")

```

 Odd numbers: [17, 43, 33, 61, 15, 17, 49]
 Cubed numbers: [4913, 79507, 35937, 226981, 238328, 3375, 17576, 4913, 21952, 117649]
 Mean: 35.1
 Median: 30.5
 Mode: 17
 Standard Deviation: 17.785449733482203
 Variance: 316.3222222222222

✓ Exercise 7

```

def is_prime(number):
    """Check if a number is prime."""
    if number <= 1:
        return False
    for i in range(2, int(number ** 0.5) + 1):
        if number % i == 0:
            return False
    return True

# Main program
try:
    # Taking input from the user
    user_input = int(input("Enter a number: "))

    # Checking if the number is prime
    if is_prime(user_input):
        print(f"The number {user_input} is prime.")
    else:
        print(f"The number {user_input} is not prime.")
except ValueError:
    print("Please enter a valid integer.")

```

 Enter a number: 7
 The number 7 is prime.

✓ Exercise 8

```
def fibonacci_series(n):
    """Print Fibonacci series up to n."""
    a, b = 0, 1
    print("Fibonacci series up to", n, ":")
    while a <= n:
        print(a, end=" ")
        a, b = b, a + b
    print()

# Main program
try:
    # Taking input from the user
    user_input = int(input("Enter a number: "))

    # Printing the Fibonacci series up to the given number
    fibonacci_series(user_input)
except ValueError:
    print("Please enter a valid integer.")
```

```
➞ Enter a number: 11
   Fibonacci series up to 11 :
   0 1 1 2 3 5 8
```

✓ Exercise 9

```
def calculate_shaded_area(length, breadth):
    """Calculate the area of the shaded region."""
    if length <= 2 or breadth <= 2:
        return "Length and breadth must be greater than 2 meters."

    outer_area = length * breadth
    inner_length = length - 2 # Subtract 2 meters for the uniform width of 1 meter on each side
    inner_breadth = breadth - 2 # Subtract 2 meters for the uniform width of 1 meter on each side
    inner_area = inner_length * inner_breadth

    shaded_area = outer_area - inner_area
    return shaded_area

# Main program
try:
    # Taking input from the user
    length = float(input("Enter the length (L) of the bigger rectangle in meters: "))
    breadth = float(input("Enter the breadth (B) of the bigger rectangle in meters: "))

    # Calculate and print the area of the shaded region
    result = calculate_shaded_area(length, breadth)
    print(f"The area of the shaded region is: {result} square meters")
except ValueError:
    print("Please enter valid numerical values for length and breadth.")
```

```
➞ Enter the length (L) of the bigger rectangle in meters: 100
   Enter the breadth (B) of the bigger rectangle in meters: 20
   The area of the shaded region is: 236.0 square meters
```

✓ Exercise 10

```
import math

def calculate_circle_area(diameter):
    """Calculate the area of a circle given its diameter."""
    radius = diameter / 2
    area = math.pi * (radius ** 2)
    return area

# Main program
try:
    # Given diameter
    diameter = 2 # meters

    # Calculate the area of the circle
    area = calculate_circle_area(diameter)

    # Print the result
    print(f"The area of a circle with a diameter of {diameter} meters is: {area:.2f} square meters")
except ValueError:
    print("Please enter a valid numerical value for the diameter.")
```

→ The area of a circle with a diameter of 2 meters is: 3.14 square meters

✓ Exercise 11

```
import math

def calculate_hypotenuse(base, height):
    """Calculate the hypotenuse of a right-angled triangle given its base and height."""
    hypotenuse = math.sqrt(base**2 + height**2)
    return hypotenuse

# Main program
try:
    # Given base and height
    base = 10 # meters
    height = 12 # meters

    # Calculate the hypotenuse
    hypotenuse = calculate_hypotenuse(base, height)

    # Print the result
    print(f"The hypotenuse of a right-angled triangle with base {base} meters and height {height} meters is: {hypotenuse:.2f} meters")
except ValueError:
    print("Please enter valid numerical values for base and height.")
```

→ The hypotenuse of a right-angled triangle with base 10 meters and height 12 meters is: 15.62 meters

✓ Exercise 12

```
import math

def calculate_distance(x1, y1, x2, y2):
    """Calculate the distance between two points (x1, y1) and (x2, y2)."""
    distance = math.sqrt((x2 - x1)**2 + (y2 - y1)**2)
    return distance

# Main program
try:
    # Coordinates of points A and B
    x1, y1 = 7, 8
    x2, y2 = 2, 3

    # Calculate the distance between A and B
    distance = calculate_distance(x1, y1, x2, y2)

    # Print the result
    print(f"The distance between points A({x1}, {y1}) and B({x2}, {y2}) is: {distance:.2f} units")
except ValueError:
    print("Please enter valid numerical values for coordinates.")
```

→ The distance between points A(7, 8) and B(2, 3) is: 7.07 units

▼ Exercise 13

```
def calculate_simple_interest(principal, rate, time):  
    """Calculate simple interest."""  
    interest = principal * rate * time  
    return interest  
  
# Main program  
try:  
    # Given values  
    principal = 10000 # initial investment in dollars  
    annual_rate = 0.05 # annual interest rate (5%)  
    time_in_years = 3 # time in years
```