TASK REPORT
ALGORITHMS AND DATA STRUCTURE
WEEK 3

Created by:

Onic Agustino

L200234275

X

INFORMATICS ENGINEERING

FACULTY OF COMMUNICATION AND INFORMATICS

UNIVERSITAS MUHAMMADIYAH SURAKARTA

2024/2025

TASK

1. Exercise 1

```python
import math

class Point:
    def __init__(self, x=0, y=0):
        """Initialize the point with coordinates x and y."""
        self.x = x
        self.y = y

    def set_location(self, x, y):
        """Set the location of the point."""
        self.x = x
        self.y = y

    def distance_from_origin(self):
        """Calculate the distance from the origin (0, 0)."""
        return math.sqrt(self.x ** 2 + self.y ** 2)

    def distance(self, other_point):
        """Calculate the distance from another point."""
        return math.sqrt((self.x - other_point.x) ** 2 + (self.y - other_point.y) ** 2)

# Example usage
point1 = Point()
point1.set_location(3, 4)
print("Distance from origin:", point1.distance_from_origin())

point2 = Point(6, 8)
print("Distance from point1 to point2:", point1.distance(point2))

# Alternative way to call methods using the class name
Point.set_location(point1, 3, 4)
print("Distance from origin (alternative call):", Point.distance_from_origin(point1))
print("Distance from point1 to point2 (alternative call):", Point.distance(point1,
point2))

print('\n--- Oleh L200234275 ---')
```

Picture 1.1 the code.

```
PS D:\Semester 4\Algorithm_and_structuredata> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/Algorithm_and_structuredata/Week3/coba1.py"
Distance from origin: 5.0
Distance from point1 to point2: 5.0
Distance from origin (alternative call): 5.0
Distance from point1 to point2 (alternative call): 5.0

--- Oleh L200234275 ---
```

Picture 1.2 the output.

2. Exercise 2

```python
class Vehicle:
    def __init__(self, name, color, year_production, capacity, mileage):
        """Initialize the vehicle with name, color, year of production, capacity, and
mileage."""
        self.name = name
        self.color = color
        self.year_production = year_production
        self.capacity = capacity
        self.mileage = mileage

    def set_capacity(self, capacity):
        """Set the capacity of the vehicle."""
        self.capacity = capacity

class Truck(Vehicle):
    def __init__(self, name, color, year_production, capacity, mileage):
        """Initialize the truck with the same attributes as the vehicle."""
        super().__init__(name, color, year_production, capacity, mileage)

# Example usage
vehicle1 = Vehicle("Car", "Red", 2020, 5, 15000)
print(f"Vehicle: {vehicle1.name}, Color: {vehicle1.color}, Year:
{vehicle1.year_production}, Capacity: {vehicle1.capacity}, Mileage: {vehicle1.mileage}")

truck1 = Truck("Truck", "Blue", 2018, 3, 30000)
print(f"Truck: {truck1.name}, Color: {truck1.color}, Year: {truck1.year_production},
Capacity: {truck1.capacity}, Mileage: {truck1.mileage}")

# Set new capacity for the truck
truck1.set_capacity(4)
print(f"Updated Truck Capacity: {truck1.capacity}")

print('\n--- Oleh L200234275 ---')
```

Picture 2.1 the code.

```
PS D:\Semester 4\Algorithm_and_structuredata> & C:/Users/Acer/AppData/Local/Programs/Python/Python311/python.exe "d:/Semester 4/Algorithm_and_structuredata/Week3/coba2.py"
Vehicle: Car, Color: Red, Year: 2020, Capacity: 5, Mileage: 15000
Truck: Truck, Color: Blue, Year: 2018, Capacity: 3, Mileage: 30000
Updated Truck Capacity: 4

--- Oleh L200234275 ---
```

Picture 2.2 the output.