# horizontal lineData Structures

Homework Assignment 1 - Python and OOP

**Tasks on this Worksheet**

* Problem 1 - Introduce Yourself (OOP) - 25 Points
* Problem 2 - Vehicle Class - 25 Points
* Problem 3 - Custom Generator - 25 Points
* Problem 4 - Flip the Bit - 25 Points

**Notes and Requirements**

* Your submission must be your effort. You can not copy other students' code.
* This worksheet is graded on performance; Implementations must be correct.
* You are encouraged to visit our office hours to ask coding questions.
* Only the latest (most recent) submission is graded.
* Late submissions are not considered for grading.
* You can not use any third-party libraries.

**All assignments on this worksheet are manually graded.**

## Problem 1 - Introduce Yourself (OOP) - 25 Points

Create a class called ***Person*** that represents a person with the following attributes:

* *first\_name* (string)
* *last\_name* (string)
* *hobbies* (list of strings)
* *age* (integer) (private)

In addition to that, implement the following methods:

* *introduce()* - introducing the person, e.g., *“Hi, my name is John Doe. I like playing guitar.”*
* *add\_hobbies(lis)* - adds a list of hobbies without introducing duplicate hobbies.

**Requirements**

* You need to implement all required attributes and methods.
* Object-oriented design principles need to be applied correctly.
* You need to assign meaningful parameter and variable names.
* You need to preserve the order of hobbies added.
* Instantiate at least three persons.
* You can not use Python sets.

**Hint**

* You can implement additional methods and attributes to familiarise yourself with OOP.

**Example Implementation**

p1 = Person("John", "Doe", 20, ["playing guitar"])

print(p1.introduce())

*# Output: Hi, my name is John Doe. I like playing guitar.*

p2 = Person("Peter", "Wang", 24, ["driving cars", "jogging"])

print(p2.introduce())

*# Output: Hi, my name is Peter Wang. I like driving cars and jogging.*

p2.add\_hobbies(["jogging", "diving"])

print(p2.introduce())

*# Output: Hi, my name is Peter Wang. I like driving cars, jogging, and diving.*

## Problem 2 - Vehicle Class - 25 Points

Create a base class called ***Vehicle*** with the following attributes:

* make (string)
* model (string)
* year (integer)

In addition to that, implement the following method:

* *get\_description()* that returns a formatted string like *“2021 Toyota Corolla.”*

Create a subclass ***Car*** that inherits from ***Vehicle*** and adds:

* *doors (integer)*

Create another subclass ***Truck*** that inherits from ***Vehicle*** and adds:

* *payload\_capacity (float)*

**Important**

* Object-oriented design principles need to be applied correctly.
* You need to assign meaningful parameter and variable names.

**Hint**

* Try to draw a diagram to sketch the relationship between the classes.

**Example Implementation**

car = Car("Toyota", "Corolla", 2021, 4)

print(car.get\_description())

*# Output: 2021 Toyota Corolla, 4-door*

truck = Truck("Ford", "F-150", 2020, 1.5)

print(truck.get\_description())

*# Output: 2020 Ford F-150, Payload capacity: 1.5 tons*

## Problem 3 - Custom Generator - 25 Points

Write a generator function called special\_generator(n) that yields numbers from 1 to n (inclusive) with the following modifications:

* If a number is divisible by 4, Yield the string "Quad" instead of the number.
* If a number is divisible by 6, Yield the string "Hex" instead of the number.
* If a number is divisible by both (i.e., divisible by 12), Yield the string "QuadHex.
* Otherwise, Yield the number itself.

**Important**

* You can not use any libraries.
* The input can be any positive integer.

**Example Implementation**

gen = special\_generator(15)

print(list(gen))

*# Output: [1,2,3,'Quad',5,'Hex',7,'Quad',9,10,11,'QuadHex',13,14,15]*

## Problem 4 – Flip the Bit - 25 Points

Write a function *flip\_bit(num, bit\_position)* that flips (*inverts*) the bit at the specified position in a given number. Assume bit\_position starts from 0 (the rightmost bit is position 0).

**Requirements**

* You can only use bitwise operations.
* You are not allowed to cast (*change the type of*) parameters.
* You are not allowed to use Python functions such as *bin()*.

**Example Implementation**

print(flip\_bit(8, 1)) *# Output: 10 (Binary: 1000 -> 1010)*

print(flip\_bit(10, 1)) *# Output: 8 (Binary: 1010 -> 1000)*