Zaawansowane środowiska programistyczne
Ćwiczenie 2

Napisać następujące klasy/interfejsy:

1. *Person*. Klasa abstrakcyjna. Ma pola chronione *name_surname* (łańcuch znaków – obiekt klasy String) przechowujące imię i nazwisko, *personal_ID* (klasy *Pesel* z ćwiczenia 1) przechowujące numer PESEL, *knowledge* (typu short) oznaczające poziom wiedzy i *wealth* (typu long) oznaczające wartość majątku. Klasa ma abstrakcyjną funkcję składową *introduce_yourself()*.

2. *Student*. Klasa dziedzicząca po klasie *Person*. Ma prywatne pole *student_number* (typu int) przechowujące numer indeksu i pola publiczne *gpa* (typu double) przechowujące średnią ocen oraz *year_of_study* (typu short) przechowujące rok studiów. Konstruktor ustawia wartość tych pól oraz pól klasy bazowej na podane w argumentach wartości, przy czym pole *wealth* jest inicjowane wartością 0. Funkcja *introduce_yourself()* wyświetla na konsoli wszystkie informacje o studencie. Ponadto klasa ma funkcję *learn()* zwiększającą wartość pola *knowledge* o podaną w argumencie wartość oraz funkcję *party()* zmniejszającą wartość pola *knowledge* o 1.

3. *Employee*. Interfejs. Ma publiczną statyczną zmienną finalną *minimum_salary* (typu int) oraz publiczną abstrakcyjną metodę *work()* przyjmująca jako argument liczbę dni pracy.

4. *Professor*. Klasa dziedzicząca po klasie *Person* i implementująca interfejs *Employee*. Ma publiczne pole *publications* (typu int) przechowujące liczbę publikacji. Konstruktor ustawia liczbę publikacji na podaną w argumencie wartość (w przypadku nie podania wartości domyślnie 50) oraz ustawia wartość pól klasy bazowej. Funkcja *introduce_yourself()* wyświetla na konsoli wszystkie informacje o profesorze. Funkcja work() zwiększa proporcjonalnie pole *knowledge* profesora o 1 za każdy pełny tydzień pracy, pole *publications* o 5 za każdy pełny rok pracy i pole *wealth* o trzykrotność *minimum_salary* za każdy pełny miesiąc pracy. Ponadto klasa ma funkcję *examine()* przyjmującą jako argument referencję do obiektu klasy *Student* i modyfikującą średnią ocen tego studenta w zależności od wartości pól *knowledge* i *year_of_study* np. zgodnie z zależnością:

$$GPA = \frac{GPA \cdot year\_of\_study + \frac{knowledge}{year\_of\_study}}{year\_of\_study + 1}$$

   oraz zwiększającą rok studiów o 1 gdy średnia ocen wynosi co najmniej 3,0.

Przetestować na dowolnym przykładzie wykorzystując obiekty poszczególnych klas potomnych i wywołując na nich zaimplementowane metody.

Dodać tam, gdzie to konieczne, sprawdzanie warunków:
$year\_of\_study \in \{1,2,3,4,5\}$
$gpa \in <2,0;\ 5,0>$
$knowledge \geq 0$
$publications \geq 0$
$wealth \geq 0$