PHYS2160 Introductory Computational Physics
2021/22 Exercise 2

1. (a) Write a Python class `Sphere` for representing spheres. The class has a data attribute `radius` indicating the radius of the sphere. It also provides the following methods:
   (i) `getRadius(self)` for returning the radius of the sphere,
   (ii) `surfaceArea(self)` for returning the surface area of the sphere,
   (iii) `volume(self)` for returning the volume of the sphere.

   (b) Write a Python class `Quadratic` for representing the quadratic function $f(x) = ax^2 + bx + c$. The class has three data attributes for the coefficients a, b, and c called `a`, `b`, and `c`. It also has the following methods:
   (i) `value(self, x)` for computing a value of f at a point x,
   (ii) `table(self, n, L, R)` for printing a table of x and f values at n values of x in the interval $[L, R]$,
   (iii) `roots(self)` for computing the roots of $f(x) = 0$.

2. (a) Write a Python class `Card` for representing a playing card. The class has two data attributes `rank` and `suit` where `rank` is an `int` in the range 1-13 indicating the ranks Ace to King and `suit` is a single character `"d"`, `"c"`, `"h"`, or `"s"` indicating the suit (diamonds, clubs, hearts, or spades). It also has the following methods:
   (i) `getRank(self)` for returning the rank of the card,
   (ii) `getSuit(self)` for returning the suit of the card,
   (iii) `value(self)` for returning the Blackjack value of the card where numeral cards 2 to 10 count at their face values, face cards (Jacks, Queens, Kings) count as 10, and Aces count as 1,
   (iv) `__str__(self)` for returning a string that names the card such as “”Ace of Spades”.
   (v) `__repr__(self)` for returning a string such that `eval` applied to the string recreates the instance.

   (b) Write a Python class `RationalNumber` for performing arithmetics with fractions. The class has two data attributes `num` and `den` which are the numerator and denominator of the fraction expressed in reduced form, e. g. 2/4 is stored as the fraction 1/2. It also provides the following methods:
   (i) `__add__(self, other)` for returning the sum of two rational numbers in reduced form,
   (ii) `__sub__(self, other)` for returning the difference of two rational numbers in reduced form,
   (iii) `__mul__(self, other)` for returning the product of two rational numbers in reduced form,
   (iv) `__truediv__(self, other)` for returning the quotient of two rational numbers in reduced form,

(v) `display(self)` for printing the fraction in the form `a/b` where `a` is the numerator and `b` is the denominator.

3. Write a Python class `Employee` for storing the data of the employee in a company. The class has four data attributes for the last name, first name, staff number, and salary of the employee called `lastname`, `firstname`, `staffnum` and `salary` where the first two are of type `string` and the last two are of type `int`. It also has a class attribute called `count` for counting the number of `Employee` objects created. Moreover, it provides the method `display(self)` for printing the information of the employee as well as the static method `getcount()` for returning the total number of employees.

4. (a) Write a Python class `Vec3D` for representing vectors in three-dimensional space by inheriting from class `Vec2D` in section 3.2.2 of the notes. In this derived class, you should override the methods defined in the base class appropriately. Moreover, add a new method `cross(self, other)` to this class for returning the cross product of two vectors in three-dimensional space.

   (b) Write a Python class `Point` for representing geometric points. This class has two data attributes `x` and `y` for the x- and y-coordinates of the point which are of type `int`. It also provides the method `__str__(self)` for returning a string representation of the point in the form `(x, y)`. Next, write a Python class `Circle` for representing circles by inheriting from class `Point`. This class has two data attributes `(x, y)` and `radius` where `(x, y)` is a `Point` object indicating the center of the circle and `radius` is a `float` indicating the radius of the circle. Moreover, it adds a method `area` for returning the area of the circle and overrides the method `__str__(self)` for returning a string representation of the circle in the form of `Center = (x, y), Radius = radius`. Finally, write a Python class `Cylinder` for representing cylinders by inheriting from class `Circle`. This class adds a data attribute `height` which is a `float` indicating the height of the cylinder and a method `volume` for returning the volume of the cylinder. (For this class, `(x, y)` refers to the base center of the cylinder.) It also overrides the method `area` for returning the surface area of the cylinder and the method `__str__(self)` for returning a string representation of the cylinder in the form of `Center = (x, y), Radius = radius, Height = height`.