Практическое занятие № 16
Тема: составление программ с использованием ООП.
Цель: закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ с ООП в IDE PyCharm Community.
Постановка задачи:
1. Создайте класс «Круг», который имеет атрибут радиуса и методы для вычисления площади, длины окружности и диаметра.
2. Создание базового класса "Транспортное средство" и его наследование для создания классов "Автомобиль" и "Мотоцикл". В классе "Транспортное средство" будут общие свойства, такие как максимальная скорость и количество колес, а классынаследники будут иметь свои уникальные свойства и методы.
3. Для задачи из блока 1 создать две функции, save_def и load_def, которые позволяют сохранять информацию из экземпляров класса (3 шт.) в файл и загружать ее обратно. Использовать модуль pickle для сериализации и десериализации объектов Python в бинарном формате.
Тип алгоритма: линейный.
Текст программы:
1.
```
# Создайте класс «Круг», который имеет атрибут радиуса и методы для вычисления
# площади, длины окружности и диаметра.
from math import pi
class Circle:

    def __init__(self, radius):
        self.radius = radius

    def area(self):
        S = pi * self.radius**2
        print("Площадь равна: ", S)

    def length(self):
        C = 2 * pi * self.radius
        print("Длина окружности равна: ", C)

    def diametr(self):
        D = 2 * self.radius
        print("Диаметр равен: ", D)

circle = Circle(5)
circle.area()
circle.length()
circle.diametr()
print(circle.__dict__)
```

Протокол работы программы:

```
Площадь равна:  78.53981633974483
Длина окружности равна:  31.41592653589793
Диаметр равен:  10
{'radius': 5}

Process finished with exit code 0
```

2.
```
# Создание базового класса "Транспортное средство" и его наследование для создания
# классов "Автомобиль" и "Мотоцикл". В классе "Транспортное средство" будут
# общие свойства, такие как максимальная скорость и количество колес, а классы-наследники
```

```
# будут иметь свои уникальные свойства и методы.

class Transport:
    def __init__(self, max_speed, num_wheels):
        self.max_speed = max_speed
        self.num_wheels = num_wheels

class Car(Transport):
    def __init__(self, max_speed, num_wheels, path, time):
        super().__init__(max_speed, num_wheels)
        self.path = path
        self.time = time

    def calculate_speed(self):
        if self.time > 0:
            return self.path / self.time
        else:
            return 0

    def describe(self):
        speed = self.calculate_speed()
        return f"Автомобиль: максимальная скорость: {self.max_speed} км/ч, количество колёс: {self.num_wheels},
путь: {self.path} км, время:: {self.time} ч, расчетная скорость: {speed} км/ч."

class Motorcycle(Transport):
    def __init__(self, max_speed, num_wheels, speed, time):
        super().__init__(max_speed, num_wheels)
        self.speed = speed
        self.time = time

    def calculate_path(self):
        return self.speed * self.time

    def describe(self):
        path = self.calculate_path()
        return f"Мотоцикл: максимальная скорость: {self.max_speed} км/ч, количество колёс:{self.num_wheels},
скорость: {self.speed} км/ч, время: {self.time} ч, расчетный путь: {path} км."

car = Car(180, 4, 400, 4)
print(car.describe())

motorcycle = Motorcycle(200, 2, 85, 3)
print(motorcycle.describe())
```

Протокол работы программы:

Автомобиль: максимальная скорость: 180 км/ч, количество колёс: 4, путь: 400 км, время:: 4 ч, расчетная скорость: 100.0 км/ч.
Мотоцикл: максимальная скорость: 200 км/ч, количество колёс:2, скорость: 85 км/ч, время: 3 ч, расчетный путь: 255 км.

Process finished with exit code 0

3.
```
# Для задачи из блока 1 создать две функции, save_def и load_def, которые позволяют
# сохранять информацию из экземпляров класса (3 шт.) в файл и загружать ее обратно.
# Использовать модуль pickle для сериализации и десериализации объектов Python в
# бинарном формате.

import pickle
```

```
from math import pi
class Circle:
    def __init__(self, radius):
        self.radius = radius

    def area(self):
        return pi * self.radius ** 2

    def length(self):
        return 2 * pi * self.radius

    def diameter(self):
        return 2 * self.radius

def save_def(circles, filename):
    with open(filename, 'wb') as file:
        pickle.dump(circles, file)

def load_def(filename):
    with open(filename, 'rb') as file:
        return pickle.load(file)

circle1 = Circle(5)
circle2 = Circle(7)
circle3 = Circle(10)

print("Круг 1:")
print("Площадь:", circle1.area())
print("Длина окружности:", circle1.length())
print("Диаметр:", circle1.diameter())

print("\nКруг 2:")
print("Площадь:", circle2.area())
print("Длина окружности:", circle2.length())
print("Диаметр:", circle2.diameter())

print("\nКруг 3:")
print("Площадь:", circle3.area())
print("Длина окружности:", circle3.length())
print("Диаметр:", circle3.diameter())

circles = [circle1, circle2, circle3]
save_def(circles, "circles.bin")

loaded_circles = load_def("circles.bin")
print("\nЗагруженные круги:")
for circle in loaded_circles:
    print("Площадь:", circle.area())
    print("Длина окружности:", circle.length())
    print("Диаметр:", circle.diameter())
```

Протокол работы программы:

Круг 1:
Площадь: 78.53981633974483
Длина окружности: 31.41592653589793
Диаметр: 10

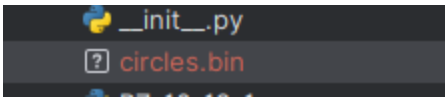
Круг 2:
Площадь: 153.93804002589985
Длина окружности: 43.982297150257104
Диаметр: 14

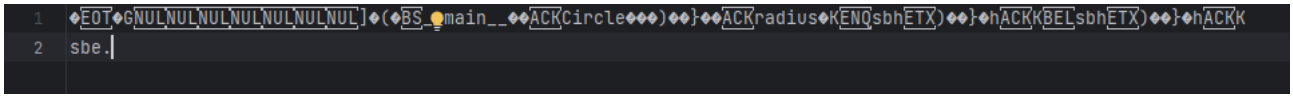
Круг 3:
Площадь: 314.1592653589793
Длина окружности: 62.83185307179586
Диаметр: 20

Загруженные круги:
Площадь: 78.53981633974483
Длина окружности: 31.41592653589793
Диаметр: 10
Площадь: 153.93804002589985
Длина окружности: 43.982297150257104
Диаметр: 14
Площадь: 314.1592653589793
Длина окружности: 62.83185307179586
Диаметр: 20

Process finished with exit code 0

Файл был создан:





Вывод: в процессе выполнения практического занятия выработала навыки составление программ с использованием ООП в IDE PyCharm Community.
Выполнены разработка кода, отладка, тестирование, оптимизация программного кода.
Готовые программные коды выложены на GitHub.