Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение
высшего образования
«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт перспективной инженерии
Департамент цифровых, робототехнических систем и электроники

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №1
дисциплины
«Объектно-ориентированное программирование»
Вариант 13

Выполнил:
Мотовилов Вадим Борисович
3 курс, группа ИВТ-б-о-22-1,
09.03.01 «Информатика и вычислительная техника»,
направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем», очная форма обучения

_____
(подпись)

Проверил:
Воронкин Р.А.

_____
(подпись)

Отчет защищен с оценкой _____ Дата защиты_____

Ставрополь, 2024 г.

Тема: Элементы объектно-ориентированного программирования в языке Python

Цель: приобретение навыков по работе с классами и объектами при написании программ с помощью языка программирования Python версии 3.x.

Порядок выполнения работы:

1. Создал новый репозиторий, клонировал его, в нем создал ветку developer и перешел на нее.
2. Выполнил индивидуальное задание №1:

13. Поле first — дробное положительное число, катет a прямоугольного треугольника; поле second — дробное положительное число, катет b прямоугольного треугольника. Реализовать метод `hypotenuse()` — вычисление гипотенузы.

Код индивидуального задания №1:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

class Triangle:
    def __init__(self, first, second):
        if not isinstance(first, (int, float)) or first <= 0:
            raise ValueError("Катет 'a' должен быть положительным числом.")
        if not isinstance(second, (int, float)) or second <= 0:
            raise ValueError("Катет 'b' должен быть положительным числом.")
        self.first = first
        self.second = second

    def read(self):
        while True:
            try:
                self.first = float(input("Введите катет 'a': "))
                if self.first <= 0:
                    raise ValueError("Катет 'a' должен быть положительным числом.")
                break
            except ValueError as e:
                print(e)
        while True:
            try:
                self.second = float(input("Введите катет 'b': "))
                if self.second <= 0:
                    raise ValueError("Катет 'b' должен быть положительным числом.")
                break
            except ValueError as e:
                print(e)

    def display(self):
        print(f"Катет 'a': {self.first}")
        print(f"Катет 'b': {self.second}")

    def hypotenuse(self):
        return (self.first**2 + self.second**2)**0.5
```

```
def make_Triangle(first, second):
    try:
        return Triangle(first, second)
    except ValueError as e:
        print(e)
        return None

if __name__ == "__main__":

    triangle2 = Triangle(1, 1)
    triangle2.read()

    print(f"Гипотенуза: {triangle2.hypotenuse()}")
```

3. Выполнил индивидуальное задание №2:

13. Создать класс Goods (товар). В классе должны быть представлены поля: наименование товара, дата оформления, цена товара, количество единиц товара, номер накладной, по которой товар поступил на склад. Реализовать методы изменения цены товара, изменения количества товара (увеличения и уменьшения), вычисления стоимости товара.

Код индивидуального задания №2:

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-

class Goods:

    def __init__(self, name="", date="", price=0.0, quantity=0, invoice_number=""):
        self.name = name
        self.date = date
        self.price = price
        self.quantity = quantity
        self.invoice_number = invoice_number

    def read(self):
        self.name = input("Введите наименование товара: ")
        self.date = input("Введите дату оформления: ")
        self.price = float(input("Введите цену товара: "))
        self.quantity = int(input("Введите количество товара: "))
        self.invoice_number = input("Введите номер накладной: ")

    def display(self):
        print("Наименование товара:", self.name)
        print("Дата оформления:", self.date)
        print("Цена товара:", self.price)
        print("Количество товара:", self.quantity)
        print("Номер накладной:", self.invoice_number)

    def change_price(self, new_price):
        self.price = new_price

    def change_quantity(self, delta):
        self.quantity += delta

    def calculate_total_cost(self):
        return self.price * self.quantity

if __name__ == "__main__":
    product = Goods()

    product.read()
```

```
product.display()

new_price = float(input("Введите новую цену товара: "))
product.change_price(new_price)

delta = int(input("Введите изменение количества товара: "))
product.change_quantity(delta)

total_cost = product.calculate_total_cost()
print("Стоимость товара:", total_cost)
```

Ответы на контрольные вопросы:

1. Как осуществляется объявление класса в языке Python?

Классы объявляются при помощи ключевого слова «class» и имени класса:

```
# class syntax
class MyClass:
    var = ...  # некоторая переменная

    def do_smt(self):
    # какой-то метод
```

Рисунок 1. Пример объявления класса

2. Чем атрибуты класса отличаются от атрибутов экземпляра?

Атрибуты класса принадлежат самому классу и общие для всех его экземпляров, тогда как атрибуты экземпляра уникальны для каждого объекта (экземпляра) класса.

3. Каково назначение методов класса?

Методы класса предназначены для выполнения операций, связанных с классом или его экземплярами. Они могут изменять состояние объекта, выполнять вычисления или взаимодействовать с другими объектами.

4. Для чего предназначен метод __init__() класса?

Метод __init__() является конструктором класса и вызывается при создании нового экземпляра. Он используется для инициализации атрибутов объекта и выполнения любых необходимых начальных настроек.

5. Каково назначение self?

self − это ссылка на текущий экземпляр класса. Она используется для доступа к атрибутам и методам объекта внутри класса. Это позволяет различать атрибуты экземпляра от локальных переменных.

6. Как добавить атрибуты в класс?

Атрибуты можно добавлять в класс как в теле класса (атрибуты класса), так и в методе __init__() (атрибуты экземпляра).

7. Как осуществляется управление доступом к методам и атрибутам в языке Python?

В Python управление доступом осуществляется с помощью соглашений об именах. Атрибуты и методы, начинающиеся с одного подчеркивания (`_`), считаются защищенными, а начинающиеся с двух подчеркиваний (`__`) − приватными. Однако это не является строгим ограничением, а лишь соглашением.

8. Каково назначение функции isinstance?

Функция isinstance() используется для проверки, является ли объект экземпляром определенного класса или его подкласса. Это позволяет безопасно проверять типы объектов перед выполнением операций с ними.

Вывод: в результате выполнения работы были приобретены навыки по работе с классами и объектами при написании программ с помощью языка программирования Python версии 3.x.