Студент группы ИС-23 Климов Я.В.

Практическое занятие №16

Тема: Составление программ с ООП в IDE PyCharm Community.

Цель: Закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ с ООП в IDE PyCharm Community.

Задача №1

Постановка задачи.

Создайте класс "Товар" с атрибутами "название", "цена" и "количество". Напишите метод, который выводит информацию о товаре в формате "Название: название, Цена: цена, Количество: кол-во"

Текст программы:

```
# Создайте класс "Товар" с атрибутами "название",
"цена" и "количество". Напишите
# метод, который выводит информацию о товаре в
формате "Название: название,
# Цена: цена, Количество: кол-во"

class Product: # Определение класса Product
    def __init__(self, name, price, quantity):
        self.name = name
        self.price = price
        self.quantity = quantity

    def display_info(self):
        print(f"Название: {self.name},"
              f" Цена: {self.price},"
              f" Количество: {self.quantity}")
```

```
product = Product("Apple", 1500, 5) # Создание
объекта класса Product

product.display_info() # Вызов метода display_info
```

Протокол работы программы:

```
Название: Apple, Цена: 1500, Количество: 5

Process finished with exit code 0
```

Задача №2

Постановка задачи.

Создайте базовый класс "Фигура" со свойствами "ширина" и "высота". От этого класса унаследуйте классы "Прямоугольник" и "Квадрат". Для класса "Квадрат" переопределите методы, связанные с вычислением площади и периметра.

Текст программы:

```
# Создайте базовый класс "Фигура" со свойствами
"ширина" и "высота".
# От этого класса унаследуйте классы "Прямоугольник"
и "Квадрат". Для класса "Квадрат"
# переопределите методы, связанные с вычислением
площади и периметра.

class Figure:
    def __init__(self, width, height):
        self.width = width # Установка ширины фигуры
```

```
        self.height = height # Установка высоты фигуры

    def get_area(self):
        return self.width * self.height # Вычисление
площади

    def get_perimeter(self):
        return 2 * (self.width + self.height) #
Вычисление периметра

class Rectangle(Figure):
    pass

class Square(Figure):
    def __init__(self, side_length):
        super().__init__(side_length, side_length)

    def get_area(self):
        return self.width ** 2 #  Вычисляет площадь
квадрата

    def get_perimeter(self):
        return 4 * self.width # Вычисляет периметр
квадрата

rectangle = Rectangle(5, 10) # Создание
прямоугольника
square = Square(7) # Создание квадрата

print("Прямоугольник:")
print(f"Площадь: {rectangle.get_area()}")
print(f"Периметр: {rectangle.get_perimeter()}")

print("\nКвадрат:")
print(f"Площадь: {square.get_area()}")
print(f"Периметр: {square.get_perimeter()}")
```

Протокол работы программы:

Прямоугольник:
Площадь: 50
Периметр: 30

Квадрат:
Площадь: 49
Периметр: 28

Process finished with exit code 0

Задача №3

Постановка задачи.

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.

Текст программы:

```
# 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

class Product:
    def __init__(self, name, price, quantity):
        self.name = name # Установка имени продукта
```

```
        self.price = price # Установка цены продукта
        self.quantity = quantity # Установка
количества продукта

    def display_info(self):
        print(f"Название: {self.name},"
              f" Цена: {self.price},"
              f" Количество: {self.quantity}")

# Создание объектов класса Product
product1 = Product("Apple", 1500, 5)
product2 = Product("Banana", 1000, 10)
product3 = Product("Orange", 1200, 8)

def save_def(products, filename): # Функция для
сохранения списка продуктов в файл
    with open(filename, 'wb') as file: # Открытие
файла для записи
        pickle.dump(products, file)

def load_def(filename):  # Функция для загрузки
списка продуктов из файла
    with open(filename, 'rb') as file:  # Открытие
файла для чтения
        return pickle.load(file)

save_def([product1, product2, product3],
'products.pkl') # Сохранение списка продуктов в файл
'products.pkl'

loaded_products = load_def('products.pkl')

for product in loaded_products:
    product.display_info() # Вывод информации о
загруженных продуктах
```

Протокол работы программы:

Название: Apple, Цена: 1500, Количество: 5
Название: Banana, Цена: 1000, Количество: 10
Название: Orange, Цена: 1200, Количество: 8

Process finished with exit code 0

Вывод: В процессе выполнения практического занятия выработал навыки Составление программ с ООП в IDE PyCharm Community. Были использованы языковые конструкции `class,def,return ,pass,import ,\n ,with,as,for,in .`
Выполнены разработка кода, отладка, тестирование, оптимизация программного кода.
Готовые программные коды выложены на GitHub.