Студент группы ИС-22 Дьяченко Л.А.

Практическое занятие № 14

Задача №1

Тема: составление программ в функциональном стиле в IDE PyCharm Community.

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

Постановка задачи:

Создайте класс "Человек" с атрибутами "имя", "возраст" и "пол". Напишите метод,

который выводит информацию о человеке в формате "Имя: имя, Возраст: возраст,

Пол: пол".

Текст программы:

class Person:

    def \_\_init\_\_(self, name, age, gender):

        self.name = name

        self.age = age

        self.gender = gender

    def info(self):

        return f"name: {self.name}, age: {self.age}, gender: {self.gender}"

# пример исgenderьзования

man = Person("Иван", 30, "мужчина")

print(man.info())  # выводит "name: Иван, age: 30, gender: мужчина"

Протокол работы программы:

name: Иван, age: 30, gender: мужчина

Вывод: Я закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ в IDE PyCharm Community.

PyCharm Community.

Задача №2

Тема: составление программ в функциональном стиле в IDE PyCharm Community.

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

Постановка задачи:

Создание базового класса "Животное" и его наследование для создания классов

"Собака" и "Кошка". В классе "Животное" будут общие методы, такие как "дышать"

и "питаться", а классы-наследники будут иметь свои уникальные методы и свойства,

такие как "гавкать" и "мурлыкать".

Текст программы:

class Animal:

    def \_\_init\_\_(self, name):

        self.name = name

    def breathe(self):

        print(f"{self.name} дышит")

    def eat(self):

        print(f"{self.name} ест")

class Dog(Animal):

    def \_\_init\_\_(self, name, breed):

        super().\_\_init\_\_(name)

        self.breed = breed # Порода

    def bark(self):

        print(f"{self.name} лает")

class Cat(Animal):

    def \_\_init\_\_(self, name, fur\_color):

        super().\_\_init\_\_(name)

        self.fur\_color = fur\_color # Цвет меха

    def purr(self):

        print(f"{self.name} мурлычет")

# example usage

dog = Dog("Bobby", "German Shepherd")

dog.breathe()  # outputs "Bobby дышит"

dog.eat()  # outputs "Bobby ест"

dog.bark()  # outputs "Bobby лает"

cat = Cat("Murry", "black")

cat.breathe()  # outputs "Murry дышит"

cat.eat()  # outputs "Murry ест"

cat.purr()  # outputs "Murry дышит"

Протокол работы программы:

Bobby дышит

Bobby ест

Bobby лает

Murry дышит

Murry ест

Murry мурлычет

Вывод: Я закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ в IDE PyCharm Community.

PyCharm Community.

Задача №3

Тема: составление программ в функциональном стиле в IDE PyCharm Community.

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

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:

import pickle

class Person:

    def \_\_init\_\_(self, name, age, gender):

        self.name = name

        self.age = age

        self.gender = gender

    def info(self):

        return f"name: {self.name}, age: {self.age}, gender: {self.gender}"

def save\_def(filename, persons):

    """Сохраняет список объектов Person в файл."""

    with open(filename, 'wb') as file:

        pickle.dump(persons, file)

def load\_def(filename):

    """Загружает список объектов Person из файла."""

    with open(filename, 'rb') as file:

        persons = pickle.load(file)

    return persons

# Пример использования

# Создаем три экземпляра класса Person

person1 = Person("Иван", 30, "мужчина")

person2 = Person("Мария", 25, "женщина")

person3 = Person("Алексей", 40, "мужчина")

# Сохраняем объекты в файл

persons\_list = [person1, person2, person3]

save\_def('persons.pkl', persons\_list)

fd

# Загружаем объекты из файла

loaded\_persons = load\_def('persons.pkl')

# Выводим информацию о загруженных людях

for person in loaded\_persons:

    print(person.info())

Протокол работы программы:

name: Иван, age: 30, gender: мужчина

name: Мария, age: 25, gender: женщина

name: Алексей, age: 40, gender: мужчина

Вывод: Я закрепил усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ в IDE PyCharm Community.

PyCharm Community.