**Билет 4**

**1. Правила простого наследования классов. Примеры.**

**Простое наследование** — это механизм в объектно-ориентированном программировании, при котором один класс (наследник) получает все свойства и методы другого класса (родителя). Это позволяет избежать дублирования кода и упростить разработку.

**Основные правила:**

1. Используется ключевое слово class и двоеточие : для указания родительского класса.
2. Наследник получает все методы и атрибуты родителя, но может их переопределять.
3. Можно добавлять новые методы и атрибуты в класс-наследник.
4. Конструктор родителя можно вызвать с помощью super().
5. Если родительский класс содержит private-атрибуты (начинаются с \_\_), они не передаются наследнику напрямую.

Пример кода с пояснениями:  
# Базовый класс "Человек"

class Human**:**

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

self**.**name **=** name # Имя человека

self**.**age **=** age # Возраст человека

def introduce**(**self**):**

**return** f"Меня зовут {self.name}, мне {self.age} лет."

# Дочерний класс "Работник", который наследуется от "Человек"

class Employee**(**Human**):**

def \_\_init\_\_**(**self**,** name**,** age**,** job\_title**):**

super**().**\_\_init\_\_**(**name**,** age**)** # Вызов конструктора родителя

self**.**job\_title **=** job\_title # Должность работника

def introduce**(**self**):**

**return** f"Меня зовут {self.name}, мне {self.age} лет, я работаю {self.job\_title}."

# Создание объектов

person **=** Human**(**"Иван"**,** 30**)**

employee **=** Employee**(**"Алексей"**,** 40**,** "Программист"**)**

# Вывод информации

print**(**person**.**introduce**())** # Выведет**:** Меня зовут Иван**,** мне 30 лет**.**

print**(**employee**.**introduce**())** # Выведет**:** Меня зовут Алексей**,** мне 40 лет**,** я работаю Программистом**.**

Используя объектно-ориентированный подход (допускается STL) написать программу с использованием ООП позволяющую выполнять следующие действия:   
а) ввод с консоли информации о сотруднике (ФИО, дата рождения, дата начала работы в организации)   
б) запись в файл.   
в) Вывод из файла списка сотрудников, отработавших заданное пользователем количество лет и возраст которых не превышает указанное пользователем ограничение.

Код:  
import json # Для работы с файлом в формате JSON

from datetime import datetime # Для работы с датами

# Класс "Сотрудник"

class Employee**:**

def \_\_init\_\_**(**self**,** full\_name**,** birth\_date**,** start\_date**):**

"""

Конструктор класса Employee

**:**param full\_name**:** ФИО сотрудника

**:**param birth\_date**:** Дата рождения **(**строка в формате "дд.мм.гггг"**)**

**:**param start\_date**:** Дата начала работы **(**строка в формате "дд.мм.гггг"**)**

"""

self**.**full\_name **=** full\_name

self**.**birth\_date **=** birth\_date

self**.**start\_date **=** start\_date

def to\_dict**(**self**):**

"""Конвертирует объект в словарь для записи в файл"""

**return** **{**

"full\_name"**:** self**.**full\_name**,**

"birth\_date"**:** self**.**birth\_date**,**

"start\_date"**:** self**.**start\_date

**}**

@staticmethod

def from\_dict**(**data**):**

"""Создаёт объект Employee из словаря"""

**return** Employee**(**data**[**"full\_name"**],** data**[**"birth\_date"**],** data**[**"start\_date"**])**

# Функция ввода данных о сотруднике

def input\_employee**():**

full\_name **=** input**(**"Введите ФИО сотрудника: "**)**

birth\_date **=** input**(**"Введите дату рождения (дд.мм.гггг): "**)**

start\_date **=** input**(**"Введите дату начала работы (дд.мм.гггг): "**)**

**return** Employee**(**full\_name**,** birth\_date**,** start\_date**)**

# Функция записи списка сотрудников в файл

def save\_to\_file**(**employees**,** filename**=**"employees.json"**):**

with open**(**filename**,** "w"**,** encoding**=**"utf-8"**)** as file**:**

json**.**dump**([**emp**.**to\_dict**()** **for** emp in employees**],** file**,** ensure\_ascii**=**False**,** indent**=**4**)**

# Функция загрузки списка сотрудников из файла

def load\_from\_file**(**filename**=**"employees.json"**):**

**try:**

with open**(**filename**,** "r"**,** encoding**=**"utf-8"**)** as file**:**

data **=** json**.**load**(**file**)**

**return** **[**Employee**.**from\_dict**(**emp**)** **for** emp in data**]**

except FileNotFoundError**:**

**return** **[]** # Если файл не найден**,** возвращаем пустой список

# Функция фильтрации сотрудников по стажу и возрасту

def filter\_employees**(**employees**,** years\_worked**,** max\_age**):**

current\_year **=** datetime**.**now**().**year # Получаем текущий год

filtered **=** **[]**

**for** emp in employees**:**

birth\_year **=** int**(**emp**.**birth\_date**.**split**(**"."**)[-**1**])**

start\_year **=** int**(**emp**.**start\_date**.**split**(**"."**)[-**1**])**

age **=** current\_year **-** birth\_year # Рассчитываем возраст

work\_experience **=** current\_year **-** start\_year # Рассчитываем стаж

**if** work\_experience **>=** years\_worked **and** age **<=** max\_age**:**

filtered**.**append**(**emp**)**

**return** filtered

# Основная программа

**if** \_\_name\_\_ **==** "\_\_main\_\_"**:**

employees **=** load\_from\_file**()**

**while** True**:**

print**(**"\n1. Добавить сотрудника"**)**

print**(**"2. Показать сотрудников по критериям"**)**

print**(**"3. Выход"**)**

choice **=** input**(**"Выберите действие: "**)**

**if** choice **==** "1"**:**

new\_employee **=** input\_employee**()**

employees**.**append**(**new\_employee**)**

save\_to\_file**(**employees**)**

elif choice **==** "2"**:**

years **=** int**(**input**(**"Введите минимальный стаж работы: "**))**

max\_age **=** int**(**input**(**"Введите максимальный возраст: "**))**

filtered\_list **=** filter\_employees**(**employees**,** years**,** max\_age**)**

**if** filtered\_list**:**

print**(**"\nПодходящие сотрудники:"**)**

**for** emp in filtered\_list**:**

print**(**f"{emp.full\_name}, Возраст: {datetime.now().year - int(emp.birth\_date.split('.')[-1])}, Стаж: {datetime.now().year - int(emp.start\_date.split('.')[-1])}"**)**

**else:**

print**(**"\nНет сотрудников, удовлетворяющих условиям."**)**

elif choice **==** "3"**:**

**break**

**else:**

print**(**"Ошибка ввода. Попробуйте снова."**)**