Домашнее задание.

Маратулы Темирболат

**Замечания по выполнению домашнего задания.**

1. **Вверху этого документа напишите свою Фамилию и Имя.**
2. Исходные коды Ваших программ сохраняйте прямо здесь, в тексте задания, строго под текстом каждого задания.
3. В дополнение к исходным кодам вставляйте в документ скриншот результата выполнения программ.
4. Не надо документ архивировать и вкладывать скриншоты в архив. Скриншоты сразу вставляете в документ, как картинку.
5. **Скриншоты удобно делать программой https://app.prntscr.com/ru/privacy.html**
6. Переменные начинаются с маленькими буквами, слова разделяются символом «\_» нижняя черта.
7. Константы пишутся заглавными буквами, слова разделяются символом «\_» нижняя черта.
8. **Не забывайте КРАСИВО оформлять интерфейс программ, за плохой, ЛЕНИВЫЙ интерфейс я тоже СНИЖАЮ ОЦЕНКУ!**

Создать классы для программы «Электрическая техника», В программе предусмотреть классы для следующих видов бытовой техники:

1. Компьютер (Процессор, материнская память, мощность блока питания, объём оперативной памяти, объём диска);
2. Фотоаппарат (разрешение матрицы, объём оперативной памяти, объём диска (flash-карты));
3. Видеокамера (разрешение матрицы, частота кадров, тип стабилизации, объём оперативной памяти, объём диска (flash-карты))
4. Смартфон.

Продумайте взаимоотношения классов и создайте иерархию классов.

Создайте объекты указанных классов и напишите простейшую программу для их использования. Определить «магический» метод \_\_str\_\_ для каждого класса.

class MemoryEquipment:

    def \_\_init\_\_(self,ram\_size,disk\_space):

        self.\_\_ram\_size = ram\_size

        self.\_\_disk\_space = disk\_space

    def get\_ram\_size(self):

        return self.\_\_ram\_size

    def set\_ram\_size(self,ram\_size):

        self.\_\_ram\_size = ram\_size

    def get\_disk\_space(self):

        return self.\_\_disk\_space

    def set\_disk\_space(self,disk\_space):

        self.\_\_disk\_space = disk\_space

    def \_\_str\_\_(self):

        return 'Размер оперативной памяти: {} мб. Объем диска: {} гб.\

'.format(self.\_\_ram\_size,self.\_\_disk\_space)

class ComputerEquipment:

    def \_\_init\_\_(self,processor,maternal\_memory,power\_supply\_power):

        self.\_\_processor = processor

        self.\_\_material\_memory = maternal\_memory

        self.\_\_power\_supply\_power = power\_supply\_power

    def get\_processor(self):

        return self.\_\_processor

    def set\_processor(self,processor):

        self.\_\_processor = processor

    def get\_maternal\_memory(self):

        return self.\_\_material\_memory

    def set\_maternal\_memory(self,maternal\_memory):

        self.\_\_material\_memory = maternal\_memory

    def get\_power\_supply(self):

        return self.\_\_power\_supply\_power

    def set\_power\_supply(self,power\_supply):

        self.\_\_power\_supply\_power = power\_supply

    def \_\_str\_\_(self):

        return 'Процессор: {}. Объем Материнской памяти: {} мб. \

Мощность блока питания: {} Вт.\

'.format(self.\_\_processor,self.\_\_material\_memory,self.\_\_power\_supply\_power)

class CameraEquipment:

    def \_\_init\_\_(self,matrix\_resolution):

        self.\_\_matrix\_resolution = matrix\_resolution

    def get\_matrix\_resolution(self):

        return self.\_\_matrix\_resolution

    def set\_matrix\_resolution(self,matrix\_resolution):

        self.\_\_matrix\_resolution = matrix\_resolution

    def \_\_str\_\_(self):

        return 'Разрешение матрицы: {} пикселей.\

'.format(self.\_\_matrix\_resolution)

class VideoEquipment(CameraEquipment):

    def \_\_init\_\_(self,matrix\_resolution,frame\_frequency,stabilization\_type):

        CameraEquipment.\_\_init\_\_(self,matrix\_resolution)

        self.\_\_frame\_frequency = frame\_frequency

        self.\_\_stabilization\_type = stabilization\_type

    def get\_frame\_frequency(self):

        return self.\_\_frame\_frequency

    def set\_frame\_frequency(self,frame\_frequenct):

        self.\_\_frame\_frequency = frame\_frequenct

    def get\_stabilization\_type(self):

        return self.\_\_stabilization\_type

    def set\_stabilization\_type(self,stabilization\_type):

        self.\_\_stabilization\_type = stabilization\_type

    def \_\_str\_\_(self):

        return CameraEquipment.\_\_str\_\_(self) + ' Частота кадров: {} Гц. \

Тип стабилизации: {}.'.format(self.\_\_frame\_frequency,self.\_\_stabilization\_type)

class Computer(ComputerEquipment,MemoryEquipment):

    def \_\_init\_\_(self,processor,maternal\_memory,power\_supply\_power,ram\_size,disk\_space):

        ComputerEquipment.\_\_init\_\_(self,processor,maternal\_memory,power\_supply\_power)

        MemoryEquipment.\_\_init\_\_(self,ram\_size,disk\_space)

    def \_\_str\_\_(self):

        return 'Компьютер: ' + ComputerEquipment.\_\_str\_\_(self) + ' ' + MemoryEquipment.\_\_str\_\_(self)

class Camera(CameraEquipment,MemoryEquipment):

    def \_\_init\_\_(self,matrix\_resolution,ram\_size,disk\_space):

        CameraEquipment.\_\_init\_\_(self,matrix\_resolution)

        MemoryEquipment.\_\_init\_\_(self,ram\_size,disk\_space)

    def \_\_str\_\_(self):

        return 'Камера: ' + CameraEquipment.\_\_str\_\_(self) + ' ' + MemoryEquipment.\_\_str\_\_(self)

class VideoCamera(VideoEquipment,MemoryEquipment):

    def \_\_init\_\_(self,matrix\_resolution,frame\_frequency,stabilization\_type,ram\_size,disk\_space):

        VideoEquipment.\_\_init\_\_(self,matrix\_resolution,frame\_frequency,stabilization\_type)

        MemoryEquipment.\_\_init\_\_(self,ram\_size,disk\_space)

    def \_\_str\_\_(self):

        return 'Видеокамера: ' + VideoEquipment.\_\_str\_\_(self) + ' ' + MemoryEquipment.\_\_str\_\_(self)

class Smartphone(ComputerEquipment,VideoEquipment,MemoryEquipment):

    def \_\_init\_\_(self,processor,maternal\_memory,power\_supply\_power,matrix\_resolution,frame\_frequency,stabilization\_type,ram\_size,disk\_space):

        ComputerEquipment.\_\_init\_\_(self,processor,maternal\_memory,power\_supply\_power)

        VideoEquipment.\_\_init\_\_(self,matrix\_resolution,frame\_frequency,stabilization\_type)

        MemoryEquipment.\_\_init\_\_(self,ram\_size,disk\_space)

    def \_\_str\_\_(self):

        return 'Смартфон: ' + ComputerEquipment.\_\_str\_\_(self) + ' ' + VideoEquipment.\_\_str\_\_(self) + ' ' + MemoryEquipment.\_\_str\_\_(self)

ONE\_STEP = 1

class View:

    def input\_processor(self):

        return input('Введите ваш процессор: ')

    def input\_maternal\_memory(self):

        return int(input('Введите объем материнской памяти: '))

    def input\_power\_supply\_power(self):

        return int(input('Введите Мощность блока питания: '))

    def input\_ram\_size(self):

        return int(input('Введите объем оперативной памяти: '))

    def input\_disk\_space(self):

        return int(input('Введите объем диска: '))

    def input\_matrix\_resolution(self):

        return input('Введите разрешение матрицы: ')

    def input\_frame\_frequency(self):

        return int(input('Введите частоту кадров: '))

    def input\_stabilization\_type(self):

        return input('Введите тип стабилизации: ')

    def input\_user\_choice(self):

        return input('Ваш выбор: ')

    def view\_device\_info(self,device):

        print(device)

    def show\_items(self,items):

        print()

        item\_number = 1

        for item in items:

            print(item\_number,item)

            item\_number += ONE\_STEP

        print()

    def view\_all\_added\_devices(self,all\_devices):

        print()

        item\_number = 1

        for device in all\_devices:

            print(item\_number,device)

            item\_number += ONE\_STEP

        print()

    def show\_wrong\_user\_input\_error(self):

        print('Извините, но вы ввели недопустимый символ!')

    def show\_good\_buy\_message(self):

        print('Спасибо за использование программы! До свидания!')

    def show\_successful\_added\_created\_device(self):

        print('Созданный вами девайс успешно добавлен!')

ADD\_NEW\_COMPUTER\_OPTION = '1'

ADD\_NEW\_CAMERA\_OPTION = '2'

ADD\_NEW\_VIDEOCAMERA\_OPTION = '3'

ADD\_NEW\_SMARTPHONE\_OPTION = '4'

VIEW\_ALL\_ADDED\_DEVICES\_OPTION = '5'

EXIT\_OPTION = '6'

class Controller:

    menu\_items = ['Добавить новый Компьютер',

                  'Добавить новый Фотоаппарат',

                  'Добавить новую Видеокамеру',

                  'Добавить новый Смартфон',

                  'Просмотреть все добавленные девайсы',

                  'Выход']

    devices = []

    view = View()

    def add\_new\_computer(self):

        processor = self.view.input\_processor()

        maternal\_memory = self.view.input\_maternal\_memory()

        power\_supply\_power = self.view.input\_power\_supply\_power()

        ram\_size = self.view.input\_ram\_size()

        disk\_space = self.view.input\_disk\_space()

        computer = Computer(processor,maternal\_memory,power\_supply\_power,ram\_size,disk\_space)

        self.devices.append(computer)

    def add\_new\_camera(self):

        matrix\_resolution = self.view.input\_matrix\_resolution()

        ram\_size = self.view.input\_ram\_size()

        disk\_space = self.view.input\_disk\_space()

        camera = Camera(matrix\_resolution,ram\_size,disk\_space)

        self.devices.append(camera)

    def add\_new\_videocamera(self):

        matrix\_resolition = self.view.input\_matrix\_resolution()

        frame\_frequency = self.view.input\_frame\_frequency()

        stabilization\_type = self.view.input\_stabilization\_type()

        ram\_size = self.view.input\_ram\_size()

        disk\_space = self.view.input\_disk\_space()

        videocamera = VideoCamera(matrix\_resolition,frame\_frequency,stabilization\_type,ram\_size,disk\_space)

        self.devices.append(videocamera)

    def add\_new\_smartphone(self):

        processor = self.view.input\_processor()

        maternal\_memory = self.view.input\_maternal\_memory()

        power\_supply\_power = self.view.input\_power\_supply\_power()

        matrix\_resolution = self.view.input\_matrix\_resolution()

        frame\_frequency = self.view.input\_frame\_frequency()

        stabilization\_type = self.view.input\_stabilization\_type()

        ram\_size = self.view.input\_ram\_size()

        disk\_space = self.view.input\_disk\_space()

        smartphone = Smartphone(processor,maternal\_memory,power\_supply\_power,matrix\_resolution,frame\_frequency,stabilization\_type,ram\_size,disk\_space)

        self.devices.append(smartphone)

    def main\_menu(self):

        while True:

            self.view.show\_items(self.menu\_items)

            user\_choice = self.view.input\_user\_choice()

            if user\_choice == ADD\_NEW\_COMPUTER\_OPTION:

                self.add\_new\_computer()

            elif user\_choice == ADD\_NEW\_CAMERA\_OPTION:

                self.add\_new\_camera()

            elif user\_choice == ADD\_NEW\_VIDEOCAMERA\_OPTION:

                self.add\_new\_videocamera()

            elif user\_choice == ADD\_NEW\_SMARTPHONE\_OPTION:

                self.add\_new\_smartphone()

            elif user\_choice == VIEW\_ALL\_ADDED\_DEVICES\_OPTION:

                self.view.view\_all\_added\_devices(self.devices)

            elif user\_choice == EXIT\_OPTION:

                self.view.show\_good\_buy\_message()

                break

            else:

                self.view.show\_wrong\_user\_input\_error()

controller = Controller()

controller.main\_menu()





