

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ  
им. Н.Э. Баумана

Факультет “Информатика и системы управления”  
Кафедра “Системы обработки информации и управления”



Дисциплина “Парадигмы и конструкции языков программирования”  
Отчет по рубежному контролю №1  
Вариант Г10

**Выполнил:**

Студент группы ИУ5-35Б

Евдокимов М.С.

**Проверил:**

Гапанюк Ю. Е.

подпись, дата

\_\_\_\_\_/\_\_\_\_\_

Москва 2025

## Задание

В рамках рубежного контроля №1 по курсу «Парадигмы и конструкции языков программирования» требовалось разработать программу на языке Python.

### Цель работы:

1. Создать два класса данных, соответствующие предметной области «Браузер» (Класс 1) и «Компьютер» (Класс 2) (Вариант №10).
2. Реализовать связь «один-ко-многим» между классами.
3. Создать дополнительный класс для реализации связи «многие-ко-многим».
4. Создать списки объектов классов, содержащие тестовые данные (3-5 записей).
5. Реализовать три запроса в соответствии с **Вариантом Г**, адаптировав их под выбранную предметную область.
6. При реализации запросов использовать функциональные возможности Python (list/dict comprehensions, функции высших порядков).

### Запросы:

1. «Браузер» и «Компьютер» связаны соотношением «один-ко-многим». Вывести список всех компьютеров, у которых модель начинается с буквы «А», и список браузеров, установленных на них.
2. «Браузер» и «Компьютер» связаны соотношением «один-ко-многим». Вывести список компьютеров с максимальным использованием памяти браузеров на каждом компьютере, отсортированный по максимальному использованию памяти.
3. «Браузер» и «Компьютер» связаны соотношением «многие-ко-многим». Вывести список всех связанных браузеров и компьютеров, отсортированный по компьютерам.

# Листинг программы

```
# вариант 10 Г
from operator import itemgetter

class Browser:
    def __init__(self, id, name, memory_usage, computer_id):
        self.id = id
        self.name = name
        self.memory_usage = memory_usage
        self.computer_id = computer_id

class Computer:
    def __init__(self, id, model):
        self.id = id
        self.model = model

class BrowserComputer:
    def __init__(self, computer_id, browser_id):
        self.computer_id = computer_id
        self.browser_id = browser_id
```

```
# Компьютеры
computers = [
    Computer(1, 'Asus ROG'),
    Computer(2, 'Apple MacBook Pro'),
    Computer(3, 'Dell XPS'),
    Computer(11, 'Acer Predator'),
    Computer(22, 'Apple iMac'),
    Computer(33, 'Asus ZenBook'),
]
```

```
# Браузеры
browsers = [
    Browser(1, 'Chrome', 512, 1),
    Browser(2, 'Firefox', 256, 2),
    Browser(3, 'Safari', 384, 3),
    Browser(4, 'Edge', 320, 3),
    Browser(5, 'Opera', 192, 3),
]
```

```
# Связи многие-ко-многим
browsers_computers = [
    BrowserComputer(1, 1),
    BrowserComputer(2, 2),
    BrowserComputer(3, 3),
    BrowserComputer(3, 4),
    BrowserComputer(3, 5),
    BrowserComputer(11, 1),
    BrowserComputer(22, 2),
    BrowserComputer(33, 3),
    BrowserComputer(33, 4),
]
```

```

    BrowserComputer(33, 5),
]

def main():
    # Соединение данных один-ко-многим
    one_to_many = [(b.name, b.memory_usage, c.model)
                    for c in computers
                    for b in browsers
                    if b.computer_id == c.id]

    # Соединение данных многие-ко-многим
    many_to_many_temp = [(c.model, bc.computer_id, bc.browser_id)
                          for c in computers
                          for bc in browsers_computers
                          if c.id == bc.computer_id]

    many_to_many = [(b.name, b.memory_usage, comp_model)
                    for comp_model, comp_id, browser_id in many_to_many_temp
                    for b in browsers if b.id == browser_id]

    print('Задание Г1')
    print('Список всех компьютеров, у которых модель начинается с буквы "А", и список браузеров на них:')

    comps_with_a = list(filter(lambda i: i[2].startswith('А'), many_to_many))

    # Группируем по компьютерам
    res_1 = {}
    for browser_name, memory, comp_model in comps_with_a:
        if comp_model not in res_1:
            res_1[comp_model] = []
        # Добавляем браузер, если его еще нет в списке для этого компьютера
        if browser_name not in res_1[comp_model]:
            res_1[comp_model].append(browser_name)

    for comp, browsers_list in sorted(res_1.items()):
        print(f'{comp}: {browsers_list}')

    print("\nЗадание Г2")
    print('Список компьютеров с максимальным использованием памяти браузеров на каждом компьютере, отсортированный по максимальному использованию памяти:')

    res_2_unsorted = []

    # Перебираем все компьютеры
    for c in computers:
        c_browsers = list(filter(lambda i: i[2] == c.model, many_to_many))

        if len(c_browsers) > 0:
            c_memory = [memory for _, memory, _ in c_browsers]
            c_memory_max = max(c_memory)
            res_2_unsorted.append((c.model, c_memory_max))
        else:

```

```

res_2_unsorted.append((c.model, 0))

# Сортировка по максимальному использованию памяти (по убыванию)
res_2 = sorted(res_2_unsorted, key=itemgetter(1), reverse=True)

for comp, max_memory in res_2:
    print(f'{comp}: {max_memory} МБ')

print("\nЗадание Г3")
print('Список всех связанных браузеров и компьютеров, отсортированный по компьютерам:')

# Сортируем по названию компьютера
res_3 = sorted(many_to_many, key=itemgetter(2))

grouped_result = {}
for browser_name, memory, comp_model in res_3:
    if comp_model not in grouped_result:
        grouped_result[comp_model] = []
    grouped_result[comp_model].append(f'{browser_name} ({memory} МБ)')

for comp_model, browsers_list in sorted(grouped_result.items()):
    print(f'{comp_model}: {browsers_list}')

if __name__ == '__main__':
    main()

```

# Скриншоты работы программы

## Задание Г1

Список всех компьютеров, у которых модель начинается с буквы "A", и список браузеров на них:

Acer Predator: ['Chrome']

Apple MacBook Pro: ['Firefox']

Apple iMac: ['Firefox']

Asus ROG: ['Chrome']

Asus ZenBook: ['Safari', 'Edge', 'Opera']

## Задание Г2

Список компьютеров с максимальным использованием памяти браузеров на каждом компьютере, отсортированный по максимальному использованию памяти:

Asus ROG: 512 МБ

Acer Predator: 512 МБ

Dell XPS: 384 МБ

Asus ZenBook: 384 МБ

Apple MacBook Pro: 256 МБ

Apple iMac: 256 МБ

## Задание Г3

Список всех связанных браузеров и компьютеров, отсортированный по компьютерам:

Acer Predator: ['Chrome (512 МБ)']

Apple MacBook Pro: ['Firefox (256 МБ)']

Apple iMac: ['Firefox (256 МБ)']

Asus ROG: ['Chrome (512 МБ)']

Asus ZenBook: ['Safari (384 МБ)', 'Edge (320 МБ)', 'Opera (192 МБ)']

Dell XPS: ['Safari (384 МБ)', 'Edge (320 МБ)', 'Opera (192 МБ)']

Process finished with exit code 0