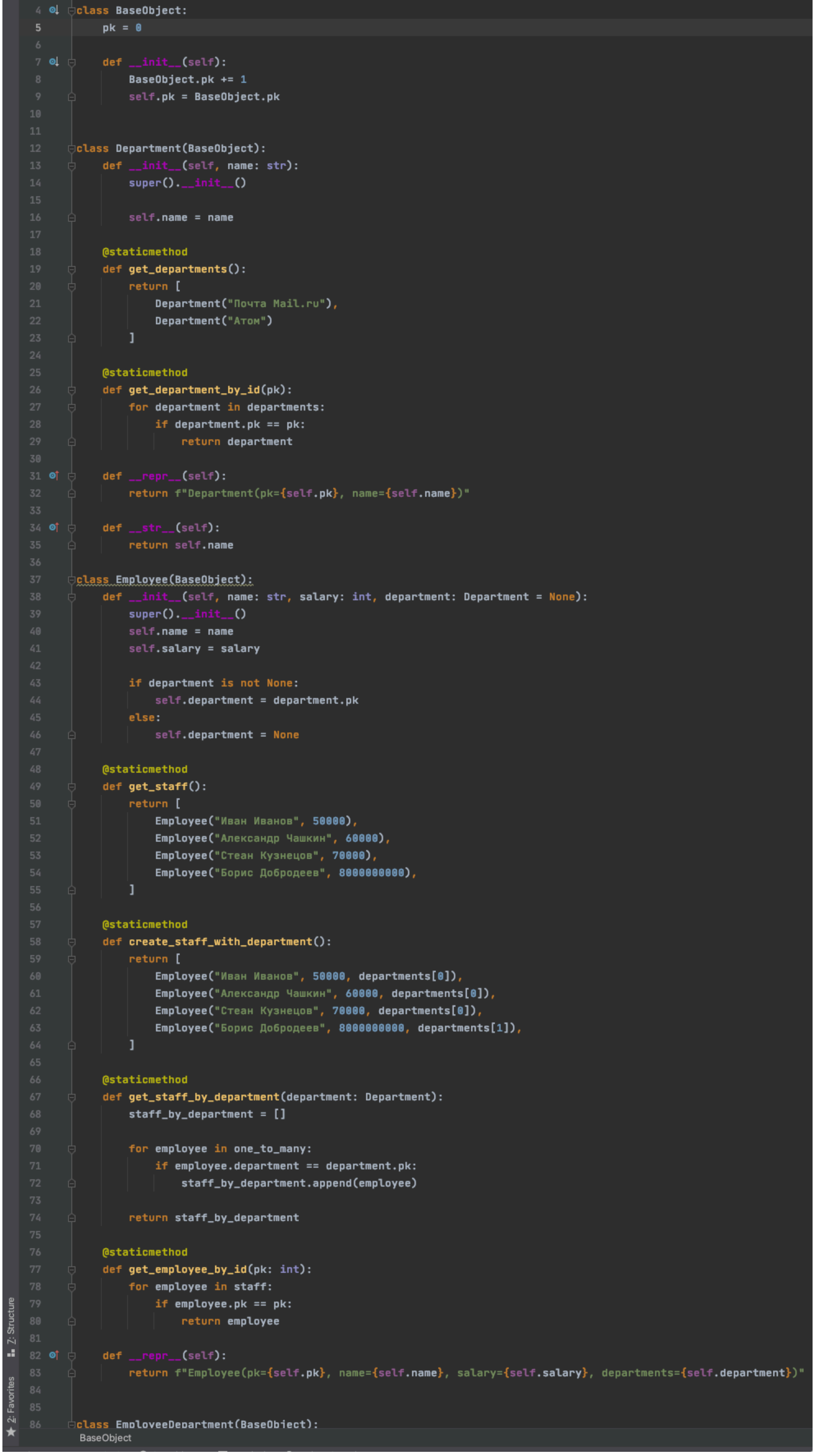
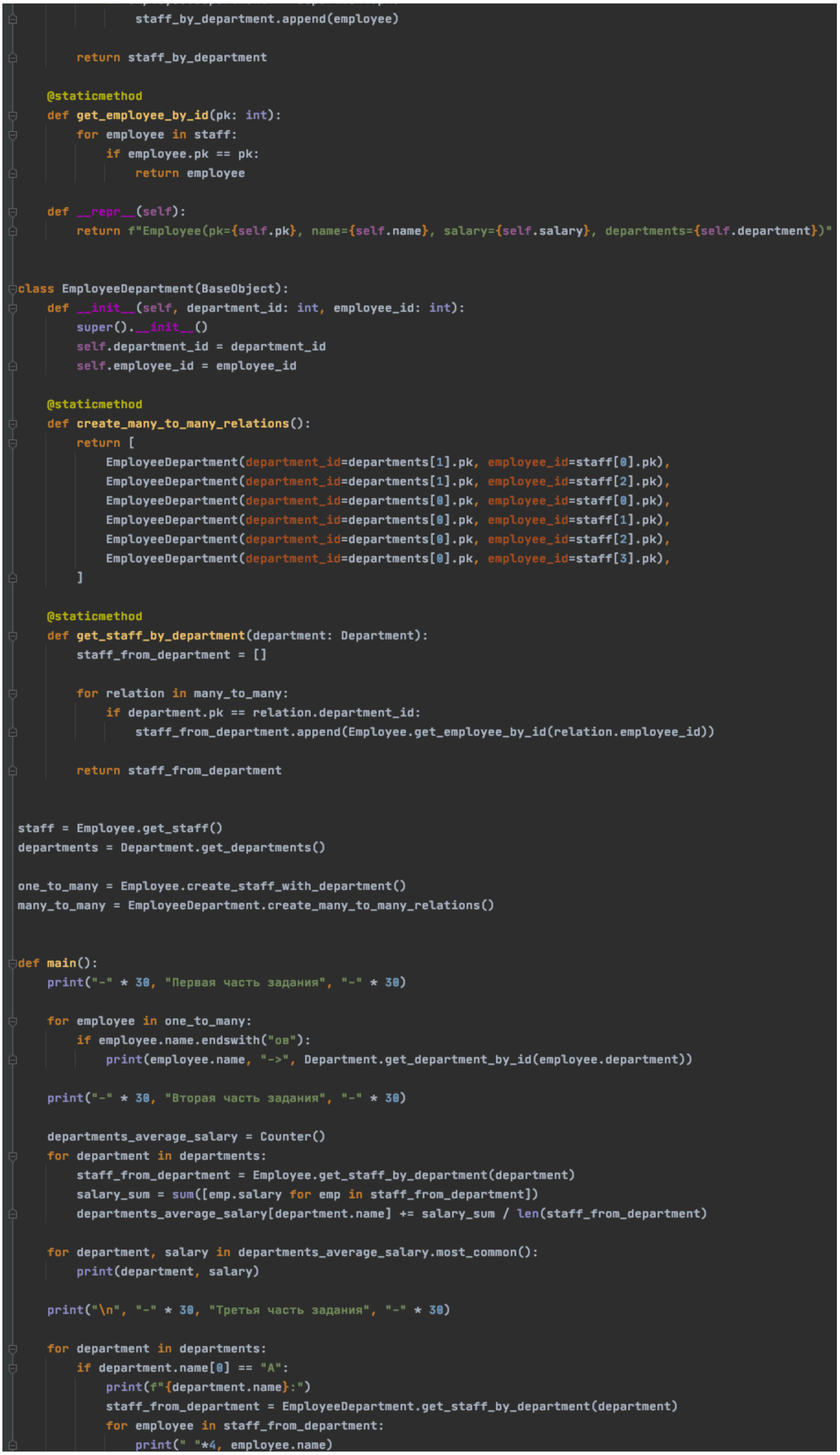
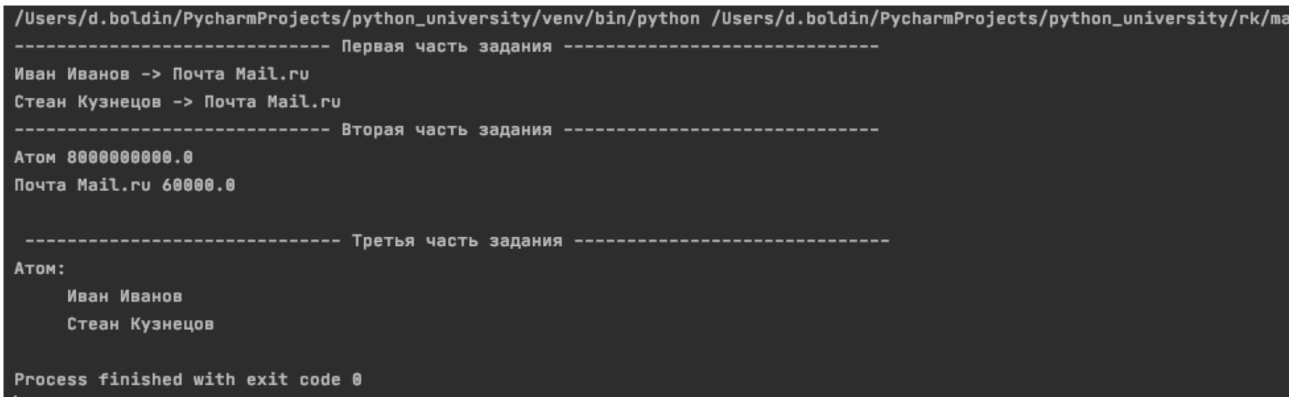


```python
class BaseObject:
    pk = 0

    def __init__(self):
        BaseObject.pk += 1
        self.pk = BaseObject.pk


class Department(BaseObject):
    def __init__(self, name: str):
        super().__init__()

        self.name = name

    @staticmethod
    def get_departments():
        return [
            Department("Почта Mail.ru"),
            Department("Атом")
        ]

    @staticmethod
    def get_department_by_id(pk):
        for department in departments:
            if department.pk == pk:
                return department

    def __repr__(self):
        return f"Department(pk={self.pk}, name={self.name})"

    def __str__(self):
        return self.name

class Employee(BaseObject):
    def __init__(self, name: str, salary: int, department: Department = None):
        super().__init__()
        self.name = name
        self.salary = salary

        if department is not None:
            self.department = department.pk
        else:
            self.department = None

    @staticmethod
    def get_staff():
        return [
            Employee("Иван Иванов", 50000),
            Employee("Александр Чашкин", 60000),
            Employee("Стеан Кузнецов", 70000),
            Employee("Борис Добродеев", 8000000000),
        ]

    @staticmethod
    def create_staff_with_department():
        return [
            Employee("Иван Иванов", 50000, departments[0]),
            Employee("Александр Чашкин", 60000, departments[0]),
            Employee("Стеан Кузнецов", 70000, departments[0]),
            Employee("Борис Добродеев", 8000000000, departments[1]),
        ]

    @staticmethod
    def get_staff_by_department(department: Department):
        staff_by_department = []

        for employee in one_to_many:
            if employee.department == department.pk:
                staff_by_department.append(employee)

        return staff_by_department

    @staticmethod
    def get_employee_by_id(pk: int):
        for employee in staff:
            if employee.pk == pk:
                return employee

    def __repr__(self):
        return f"Employee(pk={self.pk}, name={self.name}, salary={self.salary}, departments={self.department})"


class EmployeeDepartment(BaseObject):
```



```python
            staff_by_department.append(employee)

        return staff_by_department

    @staticmethod
    def get_employee_by_id(pk: int):
        for employee in staff:
            if employee.pk == pk:
                return employee

    def __repr__(self):
        return f"Employee(pk={self.pk}, name={self.name}, salary={self.salary}, departments={self.department})"


class EmployeeDepartment(BaseObject):
    def __init__(self, department_id: int, employee_id: int):
        super().__init__()
        self.department_id = department_id
        self.employee_id = employee_id

    @staticmethod
    def create_many_to_many_relations():
        return [
            EmployeeDepartment(department_id=departments[1].pk, employee_id=staff[0].pk),
            EmployeeDepartment(department_id=departments[1].pk, employee_id=staff[2].pk),
            EmployeeDepartment(department_id=departments[0].pk, employee_id=staff[0].pk),
            EmployeeDepartment(department_id=departments[0].pk, employee_id=staff[1].pk),
            EmployeeDepartment(department_id=departments[0].pk, employee_id=staff[2].pk),
            EmployeeDepartment(department_id=departments[0].pk, employee_id=staff[3].pk),
        ]

    @staticmethod
    def get_staff_by_department(department: Department):
        staff_from_department = []

        for relation in many_to_many:
            if department.pk == relation.department_id:
                staff_from_department.append(Employee.get_employee_by_id(relation.employee_id))

        return staff_from_department


staff = Employee.get_staff()
departments = Department.get_departments()

one_to_many = Employee.create_staff_with_department()
many_to_many = EmployeeDepartment.create_many_to_many_relations()


def main():
    print("-" * 30, "Первая часть задания", "-" * 30)

    for employee in one_to_many:
        if employee.name.endswith("ов"):
            print(employee.name, "->", Department.get_department_by_id(employee.department))

    print("-" * 30, "Вторая часть задания", "-" * 30)

    departments_average_salary = Counter()
    for department in departments:
        staff_from_department = Employee.get_staff_by_department(department)
        salary_sum = sum([emp.salary for emp in staff_from_department])
        departments_average_salary[department.name] += salary_sum / len(staff_from_department)

    for department, salary in departments_average_salary.most_common():
        print(department, salary)

    print("\n", "-" * 30, "Третья часть задания", "-" * 30)

    for department in departments:
        if department.name[0] == "А":
            print(f"{department.name}:")
            staff_from_department = EmployeeDepartment.get_staff_by_department(department)
            for employee in staff_from_department:
                print(" "*4, employee.name)
```



```
/Users/d.boldin/PycharmProjects/python_university/venv/bin/python /Users/d.boldin/PycharmProjects/python_university/rk/ma
------------------------------ Первая часть задания ------------------------------
Иван Иванов -> Почта Mail.ru
Стеан Кузнецов -> Почта Mail.ru
------------------------------ Вторая часть задания ------------------------------
Атом 8000000000.0
Почта Mail.ru 60000.0

 ------------------------------ Третья часть задания ------------------------------
Атом:
     Иван Иванов
     Стеан Кузнецов

Process finished with exit code 0
```