Класс Creature (абстрактный) + наследники Person и Animal

```
class Creature {
    public:
        virtual string type() = 0;
        virtual bool IsPerson() = 0; // 0 - animal, 1 - person
        void out() {
            cout << "    Type: " << type() << endl;
        };
};
```

```
class Person :public Creature {
    char gender;
public:
    Person();
    bool IsPerson() { return 1; };
    string type();
};

class Animal :public Creature {
    char kind;
public:
    Animal();
    bool IsPerson() { return 0; };
    string type();
};
```

Класс Flat

```
class Flat {
    int n_residents; // number of residents
    Creature** residents; // list of residents
```

`Flat() {` – конструктор по умолчанию

`Flat(int _n_residents)` – конструктор присваивания

`Flat(const Flat& flat)` – конструктор копирования (опционально)

`int out_n_residents()` – ф-ция возвращения числа жильцов

`Creature** out_residents()` – ф-ция возвращения списка жильцов

`void review()` – ф-ция демонстрации списка жильцов

Класс House

```
class House {
    int n_floors;
    int n_flats;
    int n_residents;
    Flat** flats;
```

`House()` – конструктор по умолчанию

`House(int _n_floors, int _n_flats, int _n_residents)` – конструктор присваивания

`void review()` – демонстрация списка квартир с жильцами по этажам

`void FG_counter()` – ф-ция подсчёта и вывода количества женщин и девочек на каждом этаже

`int BG_with_D_counter()` – ф-ция подсчёта мальчиков и девочек, живущих в одной квартире с собакой (кол-во собак в квартире на результат не влияет)