**Задание 1. Реализация деревни эльфов**

**Что нужно сделать**

Лесные эльфы расположили свою деревню в лесу, прямо на деревьях. Нужно расселить эльфов по веткам деревьев, а затем подсчитать общее количество соседей определённого эльфа.

Всего в лесу пять деревьев, каждое из которых генерируется с помощью случайных чисел, заданных в указанных далее диапазонах. На каждом дереве есть 3–5 основных больших ветвей. На каждой большой ветке расположены ещё 2-3 средние ветки. Дома эльфов построены на больших и на средних ветках. Более мелкие ветви не рассматриваются в этой задаче.

В начале программы пользователь размещает в каждом доме по одному эльфу. Делается это с помощью последовательного перебора всех имеющихся домов и запроса имени заселяемого эльфа через стандартный ввод. Если было введено None в качестве имени, то дом пропускается и не заселяется никем.

После этого требуется найти определённого эльфа по имени. Имя искомого эльфа вводится через стандартный ввод. Для найденного эльфа нужно вывести общее количество эльфов, живущих вместе с ним на одной большой ветви. Это и будет искомое количество соседей.

**Советы и рекомендации**

И для представления деревьев, и для представления больших и средних ветвей следует использовать один и тот же класс (например Branch) с полем, которое будет обозначать родительскую ветвь, и полем, обозначающим дочерние ветви. У самого дерева, в свою очередь, родительская ветвь будет всегда нулевой, то есть nullptr.

Функцию поиска эльфа в дереве можно реализовать с помощью рекурсивного метода в классе Branch, который сначала поищет эльфа в себе, а потом уже на дочерних ветвях.

Чтобы лучше представлять структуру дерева и понимать, как получать его вершину, взгляните на данный метод:

Branch\* getTopBranch()  
{  
    *//Если это дерево*  
*if (parent == nullptr) return nullptr;*  
*//Если это средняя ветка*  
*if (parent->parent == nullptr) return parent;*  
*//Если это маленькая ветка*  
*return parent->getTopBranch();*  
*}*

**Что оценивается**

Корректность работы программы и элегантность принимаемых решений.