А) Написать программу кодирования информации на основе неравенства Крафта.

Входными данными являются длины кодовых слов (в порядке убывания) **

1. Загружаем в программу текстовый файл или вводим сообщение с клавиатуры.
2. Программа считает и выводит вероятности (частоту) появления Pi каждого символа в сообщении по формуле Pi = (кол-во вхождений одного символа в текст/ кол-во всех символов в тексте).
3. Программа считает и выводит длины кодовых слов Li по формуле



**Округляя их до целых чисел.**

1. Программа проверяет длины кодовых слов на соответствие нер-ву Крафта.

### Неравенство Крафта

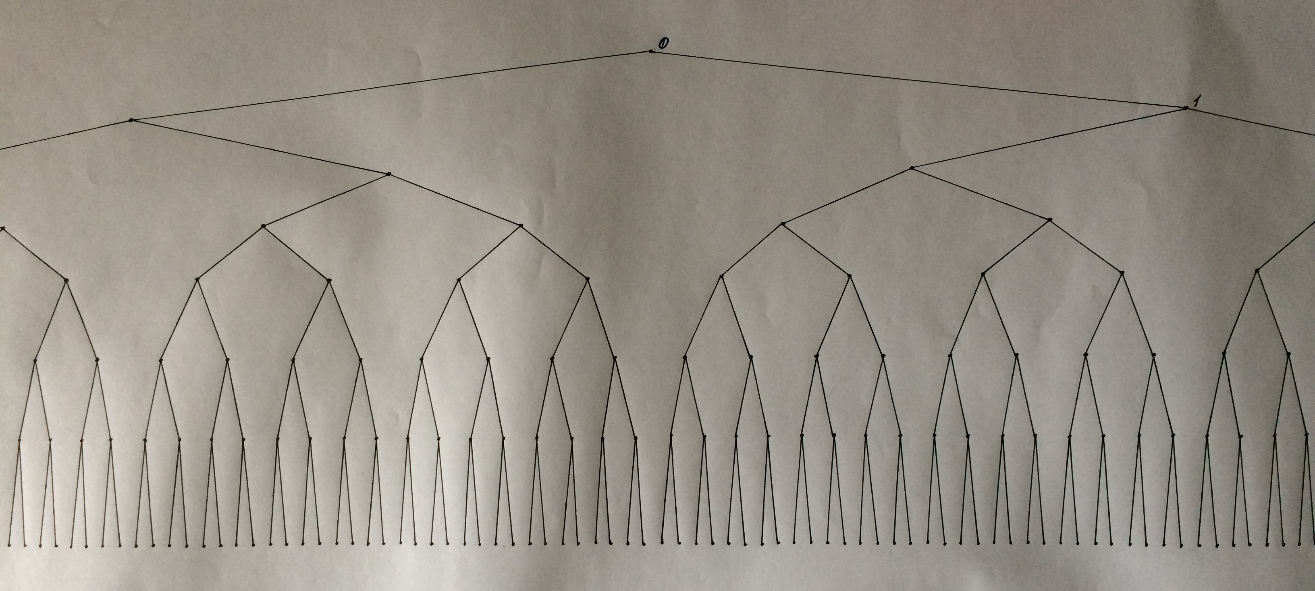
В случае префиксных кодов на пути, соединяющем основание дерева с изображающим узлом, не может быть промежуточных изображающих узлов.

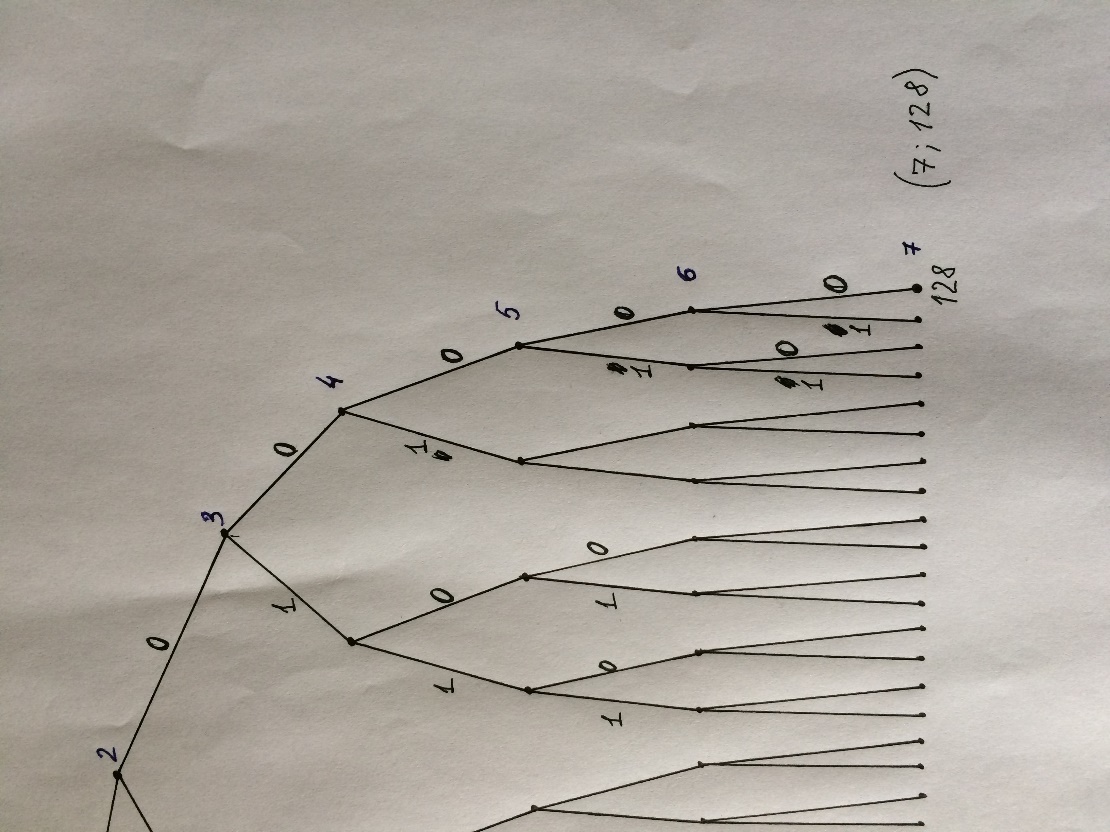
Поскольку с помощью дерева можно изобразить все кодовые слова с длиной, меньшей или равной l, то оно называется полным деревом порядка l с алфавитом объема *mу*.

**Теорема.** *Если целые числа  - длины кодовых слов удовлетворяют неравенству*

**

*то существует код, обладающий свойством префикса с алфавитом объема mу, длины кодовых слов в котором равны, этим числам. Обратно, длины кодовых слов любого кода, обладаю­щего свойством, префикса, удовлетворяют указанному неравенству.*

**

**

Условимся, что все правые ветви всегда нули, левые ветви - единицы.

1. Программа строит **графически** двоичное дерево, где уровни дерева – это длины кодовых слов l, то есть если максимальная длина кодового слова 7, то будет 7 уровней, как на рисунке выше.

(дерево хранится в памяти в виде матрицы или двумерной таблицы точек c двумя координатами, где i- уровень дерева, j- порядковый номер точки на уровне.)

1. Начиная с самого длинного кодового слова программа **рандомно** выбирает любую точку на соответствующем уровне l (**графически выделяем ее красным цветом),** и запоминает ее координаты (7 – уровень дерева, 128 – порядковый номер по счёту на этом уровне), см.рисунок.
2. Далее программа находит кодовое слово по алгоритму:

- порядковый номер точки 128 делим на 2, (если в остатке 0, то в код записывается 0, иначе 1). Проходим до корня дерева.

128/2=64 – запишем в код 0

64/2=32 – запишем в код 0

32/2= 16 – запишем в код 0

16/2= 8 – запишем в код 0

8/2 =4 – запишем в код 0

4/2= 2 – запишем в код 0

2/2=1 – запишем в код 0

- Получаем кодовое слово точки 0000000. (Но нужно читать его в обратном порядке, двигаясь по ветвям от корня дерева).

4) **Проходя от выбранной точки до корня дерева, программа должна запоминать весь путь (графически выделяем красным все пути до конечных точек)!   
И на следующем шестом уровне можно выбирать рандомно точки, которые не были задействованы в предыдущем пути, то есть, одно кодовое слово не может являться приставкой другого кодового слова!**

1. Итак, запомнив все кодовые слова программа должна вывести на экран весь код, соединив все кодовые слова в единую последовательность.

**Б)** Написать программу декодирования.  
1) На вход программы декодирования подается код (последовательность кодовых слов), **полученный на выходе предыдущей программы.**

2) Программа по коду (последовательности 0 и 1) проходит по ветвям **уже имеющегося** дерева от корня до конечных красных точек, которые уже были отмечены на дереве ранее (**показать графически**).

3) далее вывести на экран длины кодовых слов ** и соответствующие им вероятности P появления символов и сами символы в следующем виде:

Символ – Pi - Li

4) Вывести на экран сообщение, которое получили в результате декодирования.