# Задача 1.

Негадвоичное представление числа *n* – это представление его в системе счисления с *основанием* -2 (минус два) с использованием цифр 0 и 1. Например, негадвоичное представление числа 7 – это 11011. Каждое целое число имеет единственное такое представление, причем знака минус в нем не бывает. Вам даны негадвоичные представления двух чисел: 1011 и 1101. Найдите негадвоичное представление их произведения.

## Ответ: 1101111

## Решение

Построим базис данной системы счисления:

Тогда число будет вычисляться следующим образом, каждый разряд будет домножаться на соответственное число из базиса (самый правый разряд – это разряд единиц). На примере:

Аналогичным образом переведем два числа из задачи:

Произведение:

Теперь необходимо найти соответствие этому числу в негадвоичной системе счисления. Заметим, что каждое следующее число в базисе увеличивается в 2 раза, при этом самый старший разряд по модулю больше всех остальных вместе взятых. Значит если старший разряд положительный, то число будет положительным, а если отрицательный – то отрицательным. Число 27 положительное, значит первый элемент базиса больше 27. Ближайший такой элемент – это 64 ( не подходит, так как оно отрицательное, а 16 не подходит, так как сумма всех положительных начиная с 16 – это 21, что меньше 27). Дальше разряды подбираются аналогично только для числа . Итоговый ответ:

# Задача 2.

Массив A состоит из 6 чисел:  
a[1], a[2], a[3], a[4], a[5], a[6] = {5, 3, 4, 6, 7, 2}.  
Если массив A будем сортировать в убывающем порядке пузырьковой сортировкой, то после второго прохода сортировки слева направо какое значение будет стоять на третьем слева месте массива А?

## Ответ: 7

## Решение

Сортировка – это алгоритм, который упорядочивает элементы в массиве в определённом порядке (в данной задаче по убыванию).

Алгоритм сортировки пузырьком устроен следующим образом: алгоритм состоит из повторяющихся проходов по сортируемому массиву. За каждый проход элементы последовательно сравниваются попарно и, если порядок в паре неверный, выполняется обмен элементов. Проходы по массиву повторяются раз или до тех пор, пока на очередном проходе не окажется, что обмены больше не нужны, что означает — массив отсортирован. При каждом проходе алгоритма по внутреннему циклу, очередной наименьший элемент массива ставится на своё место в конце массива рядом с предыдущим «наименьшим элементом», а наибольший элемент перемещается на одну позицию к началу массива («всплывает» до нужной позиции, как пузырёк в воде. Отсюда и название алгоритма).

Код на языке python выглядит следующим образом:

**a = [5, 3, 4, 6, 7, 2]**

**for i in range(len(a)):**

**for j in range(1, len(a) - i):**

**if a[j] > a[j - 1]:**

**a[j], a[j - 1] = a[j - 1], a[j]**

**print(\*a)**

После второго выполнения внешнего цикла массив будет выглядеть так:

5 6 7 4 3 2

# Задача 3.

*A* – натуральное число, такое что НОК(100, *A*) = 600 и НОК(100, *A* + 1) = 100.  
Найдите значение НОК(100, *A* + 2). *Примечание*. НОК – это наименьшее общее кратное.

## Ответ: 1300

## Решение

Данную задачу можно решить, как программой, так и математически. Для нахождения НОК есть следующая формула:

где НОД – это наибольший общий делитель. Для нахождения наибольшего общего делителя можно воспользоваться алгоритмом Евклида, который основан на том, что

где – это остаток от деления a на b.

Заметим, что НОК >= A, тогда А будет <= 100. Напишем программу:

def gcd(a, b):

while b != 0:

a, b = b, a % b

return a

def lcm(a, b):

return (a \* b) // gcd(a, b)

for a in range(0, 101):

if lcm(100, a) == 600 and lcm(100, a + 1) == 100:

print(a, lcm(100, a + 2))

Программа найдёт *а* = 24 и НОК = 1300.

Математический способ. Заметим, что НОК(100, a + 1) = 100, значит 100 делится на *а*+1, значит *a*+1 комбинация из делителей 100: 5\*5\*2\*2, для 600 делители 5\*5\*3\*2\*2\*2 определяют комбинацию для *a*. Перебрав возможные комбинации делителей *a*+1 можно понять, что *а* – это число 24, и ответ 1300.

# Задача 4.

В китайских математических текстах, созданных ещё до Рождества Христова, встречается алгоритм, названный “вычти друг из друга”. Ниже приведено описание этого алгоритма на языке Basic для входных переменных *a* и *b*:

c = 0  
Do While a Mod 2 = 0 And b mod 2 = 0  
a = a / 2  
b = b / 2  
c = c + 1  
Loop  
Do While a <> b  
If a < b Then  
t = a  
a = b  
b = t  
End If  
r = a – b  
a = b  
b = r  
Loop  
Print r\*2^c

Что выдаст этот алгоритм, если входные данные a = 294 и b = 504?

## Ответ: 42

## Решение

Можно либо заметить, что это НОД упомянутых чисел, который можно просто вычислить, или воспользоваться приведенной программой, либо переписать код на известном вам языке программирования, узнав, что операция ^ на Visual Basic обозначает возведение в степень.

Переписанный на Python код:

a = 294

b = 504

c = 0

while a % 2 == 0 and b % 2 == 0:

a //= 2

b //= 2

c += 1

while a != b:

if a < b:

a, b = b, a

a, b = b, a - b

print(b \* 2 \*\* c)

# Задача 5.

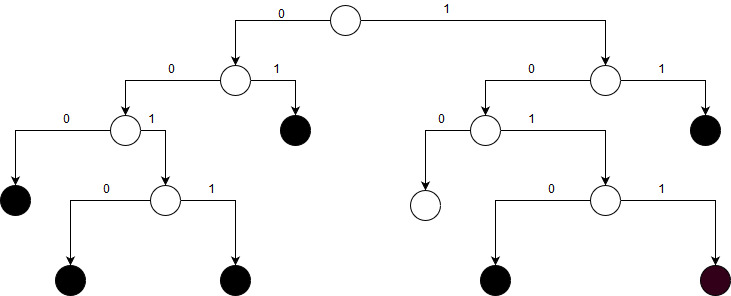
10 символов сообщения кодируются двоичными кодами, удовлетворяющими условию Фано, которое означает, что ни одно кодовое слово не является началом никакого другого кодового слова. Нам известны коды 7 символов: 000, 11, 01, 0010, 0011, 1010, 1011. Чему равна суммарная длина трех остальных символов, если известно, что она минимально возможная.

## Ответ: 14.

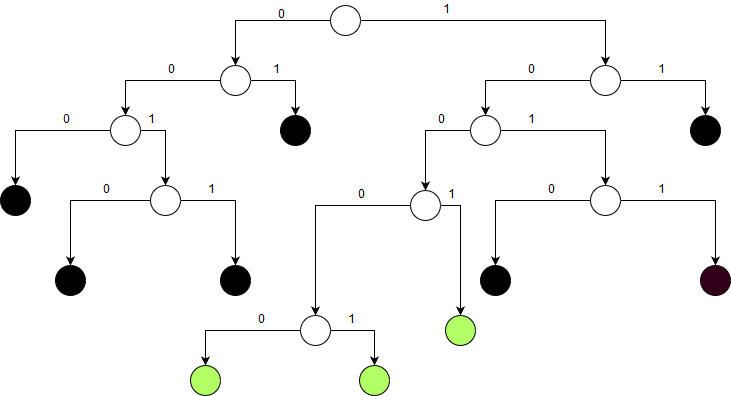
## Решение

Условие Фано: «Никакое кодовое слово не может быть началом другого кодового слова».

Удобнее всего решать такие задачи, рисуя дерево кодов и отмечая в его листьях буквы. При этом из листьев уже не выходят новые ребра. Закрашенные листья – это буквы, которые уже даны.



У нас остаётся всего одно свободное место, а нам требуется ещё 3 буквы. Добавить их можно, например, так:



Длины новых путей 5, 5, и 4.