# 1 Задание 1

## 1.1 Задача 1

### а)

Алгоритм выведет все простые числа от 2 до n, если в k-ой ячейке был 0, то номер в этой ячейке не делился ни на какие числа, меньшие его. Значит это простое число.

**Ответ:** Простые числа от 2 до n.

### б)

N раз алгоритм проходит с k-ой позиции до конца с шагом 1, где $1 \leq k \leq n$, сл-но асимптотика будет f(n) = $O(n^2)$.

**Ответ:** f(n)= $Om(n^2)$.

### в)

В прошлом пункте было показано, что f(n)= $O(n^2)$

**Ответ:** Да, является.

## 1.2 Задача 2

$g(n) = \Theta(f(n)) \Leftrightarrow \exists C_1, C_2 > 0, N \in \mathbb{N} : \forall n > N \quad C_1 \cdot f(n) \leqslant g(n) \leqslant C_2 \cdot f(n)$

### а)

$c < 1$
Сумма бесконечно малой геометрической прогрессии: g(n) = $\frac{1}{1-c}$, т.е. $g(n) = \theta(1)$

### b)

$c = 1$
$g(n) = n + 1$, возьмём $C_1 = 1$ и $C_2 = 2$, т.е. $g(n) = \Theta(n)$

### c)

$c > 1$
Сумма геом. прогрессии: $g(n) = \frac{c^n-1}{c-1} = \frac{1}{c-1}c^n - \frac{1}{c-1} = \Theta(c^n)$, т.е. $g(n) = \Theta(c^n)$

## 1.3 Задача 3

### а)

$g(n) = O(f(n)) \leftrightarrow \exists C > 0, N \in \mathbb{N} : \forall n > N \ g(n) \leq C \cdot f(n)$
$\forall N \geq 10 \rightarrow log(n) \geq 1 \rightarrow nlog(n) \geq n \rightarrow n = O(nlog(n))$

**Ответ:** Да, верно.

### б)

$g(n) = \Theta(f(n)) \leftrightarrow \exists C_1, C_2 > 0, N \in \mathbb{N} : \forall n > N \ C_1 \cdot f(n) \leq g(n) \leq C_2 \cdot f(n)$
Возьмём отрицание от условия нижней границы: $\forall c > 0 \forall \Rightarrow \exists n \geq N : nlog(n) < c \cdot n^{1+\varepsilon}$ или же $log(n) < c \cdot n^{\varepsilon}$. Теперь возьмём производные от обоих частей, получим: $\frac{1}{ln(10)n} < c \cdot \varepsilon n^{\varepsilon - 1}$, $1 < c \cdot \varepsilon n^{\varepsilon}$. Т.к. $\varepsilon > 0$, то при достаточно больших n это условие будет выполнено. А сл-но производная правой части будет меньше, чем производная левой части. Т.к. обе функции устремляются к бесконечности, то при больших n левая часть будет меньше, чем правая часть. А значит такое n по условиям задачи найдется. Т.е. $\nexists \varepsilon > 0 : nlog(n) = \Omega(n^{1+\varepsilon})$

**Ответ:** Нет, неверно.

## 1.4 Задача 4

### 1.а)

Возьмём f(n) $= n \cdot log(n)$, а $g(n) = 1$. Тогда $h(n) = n \cdot log(n) = \Theta(n \cdot log(n))$

**Ответ:** Да, может.

### 1.б)

$h(n) = \frac{f(n)}{g(n)} \leq \frac{c_1 \cdot n^2}{c_2 \cdot 1}$ Получается, что $h(n) = O(n^2)$, т.е. $h(n) \neq \Omega(n^3) \rightarrow h(n) \neq \Theta(n^3)$

**Ответ:** Нет, не может.

### 2.)

Как показано ранее $h(n) = O(n^2)$ (например при $f(n) = n^2, g(n) = 1$). Т.к. у нас нет нижней оценки на f(n), то она может быть сколько угодной малой, а значит сокльно угодно малой может быть и h(n), а значит у нас нет нижней оценки на h(n). (Если считать, что можно быстрее, чем за O(1), то она конечно есть и равна O(1)).

**Ответ:** $h(n) = O(n^2)$, нижней оценки на h(n) нет.

## 1.5 Задача 5

От внешнего цикла получаем $k_1 = log(n)$, от внутреннего цикла $0 < i < log(n); k_2 = log(n)$, и ещё от двух внутренних циклов $0 \leq j \leq n/2, k_3 = n/2$ и $1 \leq j \leq n, k_4 = log(n)$.

Итого g(n) $= \Theta(k_1 \cdot k_2 \cdot (k_3 + k_4)) = \Theta(log(n) \cdot log(n) \cdot (n/2 + log(n))) = \Theta(n \cdot log^2(n))$

**Ответ:** $g(n) = \Theta(n \cdot log^2(n))$

## 1.6 Задача 7

Переменные: A, B, C - массивы, соотвественно счетчики x, y, z массивов A, B, C и $N_a, N_b, N_c$ - их размеры. k - количество различных элементов. 1. Будем считать, что в массивах A, B, C первые элементы идут в порядке неубывания, т.е. $A[0] \leq B[0] \leq C[0]$, если это не так перенумеруем.

2. На этом шаге сравниваем элемент A[0] с B[0] и C[0].

2.1 Если A[0] не совпало ни с кем, то увеличиваем x на один.

2.2 Если совпало A[0] с B[0] и не совпало A[0] с C[0], то $x+ = 1$, $y+ = 1$ и $k+ = 2$. (Если A[0] совпадет с C[0], то аналогично, но уже увеличиваем z).

2.3 Если совпали все три, то $x+ = 1, y+ = 1, z+ = 1, k+ = 3$

3. На следующем шаге шаге опять выбираем минимальный элемент из трёх и проделываем эту операцию.

4. Продолжаем до того момента, когда не закончится необработанные элементы в массиве.

**Корректность:** По причине того, что число элементов конечно, то из них всегда будет наименьший элемент, значит мы посчитаем все не повторяющиеся элементы. Почему же мы не посчитаем больше? Всё очень просто, у нас элементы внутри каждого массива различны, поэтому если мы его посчитаем в одном массиве, то в этом же массиве он больше никогда не встретиться, а количество элементов мы увеличивем согласованно. Т.е. сколько встретилось на определенном шаге, столько и получим. Причем на каждом шаге, мы считаем, что $A[x] \leq B[y] \leq C[z]$, это реализуемо технически т.к. после каждого мы можем переназывать переменные местами, технически это обойдется за константное время, поэтому с этим тоже нет проблем.

**Оценка:** $n = N_a + N_b + N_c$ 1) По памяти: $N_a + N_b + N_c + 7 = \Theta(n)$ - линейна 2) По времени: $C_1 \cdot N_a \cdot O(1) + C_2 \cdot N_b \cdot O(1) + C_3 \cdot N_c \cdot O(1) = \Theta(n)$ - линейна.

## 1.7 Задача 9

Будем записывать элементы последовательности в стек и если элемент вершины стека не равен элементу, который мы хотим положить на вершину, то удаляем элемент из вершины стека, а тот элемент, который должны записать, не записываем. Тогда в стеке останется только элемент, который встречается в последовательности больше половины раз или же несколько таких элементов тоже может быть.

**Корректность:** Мы не можем удалить все элементы, которые встречаются больше половины раз, так как мы обрабатываем парами, в которых элементы не равны. То есть хотя бы 2 элемента останутся (мы удаляем элементы, не равные искомому, вместе с искомым).

**Оценка:**

по времени: $O(2n) = O(n)$, т.к. каждый элемент можем записать в стек и удалить из стека(хотя такого никогда не будет). Заметим, что O(n) возможно, когда все элементы равны. Т.е. O(n).

по памяти: O(n) - всего элементов n, их все можно записать в массив, когда все элементы равны между собой.