1 Задание 3

1.1 Задача 1

Понятно, что на каждой ветке будет C_1 операций, и каждая ветка будет порождать ветку от $\frac{n}{4}$, когда $n \leq 2020$, то операций будет C_2 (т.к. во втором else цикл от 0 но n, но n не превышает 2020). $T(n) = 3 \cdot T(\frac{n}{4}) + C$, возьмём за $C = max(C_1, C_2)$, и применим Мастер теорему.

$a = 3, b = 4, d = 0, log_4 3, d = O(n^{log_4 3})$, тогда это первый случай мастер теоремы и $T(n) = \Theta(n^{log_4 3})$

Ответ: $T(n) = \Theta(n^{log_4 3})$

2 Задача 2

Кажется, что это алгоритм вычисления НОД для всех чисел массива. Все числа массива хоть и уменьшаются, но они все остаются положительными, т.к. из большего вычитается меньшее и к тому же они все различны. При этом понятно, что все числа в конце концов будут одинаковы, т.к. иначе мы могли из большего вычесть меньшее число.

Предположим, что $D =$ НОД – для всех чисел, логично, что до этого шага были числа $A = a \cdot D, B = b \cdot D$, пусть $A > B$ для определенности. После же этого шага будет $A = (a - b) \cdot D, B = b \cdot D$. Видно, что числа не могут после каждого шага стать меньше, чем D

Пусть выполнение алгоритма завершено и все числа не будут равны D, тогда $D \neq$ НОД, приходим к противоречию.

Ответ: Наибольший общий делитель всех чисел массива.

3 Задача 3

$(a + b)^2 = a^2 + 2 \cdot ab + b^2 \longrightarrow ab = \frac{(a+b)^2 - a^2 - b^2}{2}$

Отсюда видно, что сложение двух чисел за линейку, потом еще возведение в квадрат по предположению тоже за линейку, потом еще две операции сложения тоже за линейку, и деление на два мне было сказано, что тоже за линию выполняется. Итого получаем, что произведение двух чисел будет производиться за линию, при предположении того, что возводить в квадрат можно за линию.

Доказано

4 Задача 5

Вспоним о том, что $(\sum_{i=1}^{n} a_i)^2 = a_1^2 + 2 \cdot a_1 \cdot a_2 + ... + a_n^2$, тогда можем получить следующее выражение:

$$\sum_{i \neq j}^{n} (a_i \cdot a_j) = \frac{(\sum_{i=1}^{n} a_i)^2 - \sum_{i=1}^{n} a_i^2}{2}$$

Получим, что :
Сложность первого слагаемого будет $O(n)$, т.к. всего n операций сложения и одно возведение в степень.
Сложность второго слагаемого будет $O(n)$, т.к. всего n операций умножения и n-1 операция сложения.
И еще одна операция деления, т.е. в итоге получаем $O(n)$

Ответ: Получили $O(n)$ от количества операций.

5 Задача 6

5.1 а)

$T(n) = 36 \cdot T(\frac{n}{6}) + n^2$,
$a = 36, b = 6, f(n) = n^2, d = log_b a = 2, f(n) = n^2 = \Theta(n^2) = \Theta(n^d)$
Это будет второй случай мастер теоремы: $T(n) = \Theta(n^2 \cdot logn)$

Ответ: $T(n) = \Theta(n^2 logn)$

5.2 б)

$T(n) = 3T(\frac{n}{3}) + n^2$,
$a = 3, b = 3, f(n) = n^2, d = log_b a = 1, f(n) = n^2 = \Omega(n^{1+\varepsilon})$.
$\exists c : 0 < c < 1, a \cdot f(\frac{n}{b}) \leq c \cdot f(n)$
$3 \cdot \frac{n^2}{9} = \frac{n^2}{3} \leq c \cdot n^2 \longrightarrow c = \frac{1}{2}$, такое c существует и равно $\frac{1}{2}$, сл-но выполняется третий случай Мастер теоремы.

Ответ: $T(n) = \Theta(n^2)$

5.3 в)

$T(n) = 4 \cdot T(\frac{n}{2}) + \frac{n}{logn}$,
$a = 4, b = 2, d = log_b a = 2, f(n) = \frac{n}{logn}, f(n) = O(n^{2-\varepsilon})$.

Пусть $\varepsilon = \frac{1}{2}$ и это будет выполняться, т.к. любой логарифм
Это будет первый случай мастер теоремы.

Ответ: $T(n) = \Theta(n^2)$

6 Задача 7

Воспользуемся сортировкой массива с помощью слияния, когда же будем сливать или же соединять наши два остортированных массива A_1 и A_2 и будем брать элемент из массива A_2, то будем прибавлять к счетчику инверсий количество элементов, которые будут стоять в A_1 правее этого элемента.

Корректность: Предположим, что алгоритм не будет корректным, тогда найдутся инверсии, которые мы не посчитали, то есть на каком-то шаге m и k, $m < k$ и $A[m] > A[k]$, значит $A[k]\ inA_1$ и $A[m] \in A_2$, т.е. мы их будем учитывать при следующем слиянии. Получили противоречие.

Асимптотика: $T(n) = 2 \cdot T(\frac{n}{2}) + C \cdot n$,
$a = 2, b = 2, d = log_b a = 1, f(n) = C\ cdot = \Theta(n^d)$
Это будет второй случай мастер теоремы.

Ответ: $T(n) = \Theta(n \cdot logn)$

7 Задача 8

Доказать, что если $T_1(n) = a \cdot T_1(\frac{n}{b}) + f(n)$, $T_2(n) = a \cdot T_2(\frac{n}{b}) + g(n)$, и $f(n) = \Theta(g(n))$, то $T_1(n) = \Theta(T_2(n))$

$$T_1(n) = \sum_{i=0}^{log_b n} a^i \cdot f(\frac{n}{b^i}) + C_1 \cdot a^{log_b n}$$

$$T_2(n) = \sum_{i=0}^{log_b n} a^i \cdot g(\frac{n}{b^i}) + C_2 \cdot a^{log_b n}$$

Учитывая, что $f(n) = \Theta(g(n))$, т.е. $a^i \cdot f(\frac{n}{b^i}) = \Theta(a^i g(\frac{n}{b^i}))$,
сл-но $T_1(n) = \Theta(T_2(n))$

Доказано

8 Задача 9

8.1 а)

$T(n) = 3 \cdot T(\frac{n}{4}) + T(\frac{n}{6}) + n$ Рассмотрев, несколько строчек рекурсий, можем заметить, что на k-ой строке количество операций будет равно $\frac{11}{12}\cdot^k \cdot n$

$$\sum_{k=0}^{logn} (\frac{11}{12})^k \cdot n = n \cdot (1 - \frac{11}{12}^{logn})$$

Получаем, что при $\frac{n}{2} \leq T(n) \leq 12 \cdot T(n) \longrightarrow T(n) = \Theta(n)$

Ответ: $T(n) = \Theta(n)$

8.2 б)

$T(n) = T(\alpha \cdot n) + T((1 - \alpha) \cdot n) + C \cdot n$, $(0 < \alpha 1)$.

Заметим, что на каждом шаге будет $C \cdot n$ операций. Высота же дерева получится $h = max\{log_\alpha n, log_{1-\alpha} n)\}$, видно, что $h = C_2 \cdot logn$, т.к. $log(1 - \alpha)$ и $log(\alpha)$ отличаются друг от на константу. И т.к у нас одинаковое количество операций на каждой ветке, то это не повлияет на итоговый результат. (То есть это сумма ниже будет ограничена снизу и сверху двумя разными константами, умноженными на $n \cdot logn$, поэтому и получается верная асимптотическая оценка).

$$\sum_{i=0}^{C_2 \cdot logn} C \cdot n = C \cdot n \cdot ((logn + 1) \cdot C_2)$$

Ответ: $T(n) = \Theta(n \cdot logn)$

8.3 в)

$T(n) = T(\frac{n}{2}) + 2 \cdot T(\frac{n}{4}) + C \cdot n$

Точно также заметим, что на каждой ветке у нас $C \cdot n$ операций и что высота дерева будет $log_2 n \leq h \leq log_4 n$. То есть как и в примеры выше, где тоже на каждой ветке было $\Theta(n)$ операций, мы можем сумму снизу ограничить двумя константными, умноженными на $n \cdot logn$.

$$\sum_{i=0}^{logn \cdot c_1} C \cdot n = C \cdot n(c_1(logn + 1))$$

Ответ: $T(n) = \Theta(n \cdot logn)$

8.4 г)

$T(n) = 27 \cdot T(\frac{n}{3}) + \frac{n^3}{log^2 n}$
Методом долгого вглядывания обнаружим, что $T(n) = \Theta(...)$

9 Задача 10

Будем сначала вычислять n!, потом с помощью алгоритма быстрого возведения в степень вычислим $n!^{p-2} mod p$, и заполним ячейку $invfac[n]$. После ээтого вычислим $invfac[k]$, где $k \in [1, n-1]$, следующим образом: $invfac[k] = invfac[k+1] \cdot k (mod p)$.

Корректность: Поскольку n $<$ p, то n и p будут взаимно просты, следовательно можем воспользоваться малой теоремой Ферма, то есть $n!^{p-1} \equiv 1 (mod p) \longrightarrow n!^{p-2} \equiv n!^{-1} \equiv (mod p)$. Для n-ого элемента получили, что значение было вычислено корректно. $n! \cdot n!^{-1} \equiv (n-1)! \cdot (n \cdot n!^{-1})(mod p \longrightarrow (n-1)!^{-1} \equiv n!^{-1} \cdot n(mod p))$

Асимптотика: Для того, чтобы посчитать факториал нам нужно $O(n)$, чтобы быстро возвести степень $log_2(p)$, каждый оставшийся элементв вычисляется за одну арифметическую операцию, т.е. за $O(n)$. Получаем, что сложность всего алгоритма $O(n + log p)$