

Задание 1

Доказать, что

$$\sum_{i=0}^{\log n} \log\left(\frac{n}{2^i}\right) = \Theta(\log^2 n)$$

$$\begin{aligned} \sum_{i=0}^{\log n} \log\left(\frac{n}{2^i}\right) &= \sum_{i=0}^{\log n} \log(n) - i = (\log(n) + 1) \cdot \log(n) - \sum_{i=0}^{\log n} i = \\ &= (\log(n) + 1) \cdot \log(n) - \log(n) \cdot \frac{\log(n) + 1}{2} \\ &= \frac{(\log(n) + 1) \cdot \log(n)}{2} = \frac{1}{2} \log^2(n) + \frac{1}{2} \log(n) \end{aligned}$$

Для $n > 4$ верно:

$$\begin{aligned} \frac{1}{2} \log^2(n) &\leq \frac{1}{2} \log^2(n) + \frac{1}{2} \log(n) \leq 2 \log^2(n) \\ \frac{1}{2} \log^2(n) + \frac{1}{2} \log(n) &\in \Theta(\log^2(n)) \end{aligned}$$

Задание 2

Написать код алгоритма scan в fork-join модели с $O(n)$ work и $O(\log n)$ span.

```

1 func Scan(a []int, f func(int, int) int) []int {
2     n := len(a)
3     sum := make([]int, 2*n-1)
4     result := make([]int, n)
5
6     ScanUp(a, sum, 0, 0, n, f)
7     ScanDown(a, sum, result, 0, 0, n, 0, f)
8
9     return result
10 }
11
12 func ScanUp(
13     a, sum []int,
14     node, l, r int,
15     f func(int, int) int,
16 ) int {
17     if r-l == 1 {
18         sum[node] = a[l]
19         return sum[node]
20     }
21     m := (l + r) / 2
22     var left, right int
23     Fork2Join(
24         func() { left = ScanUp(a, sum, 2*node+1, l, m, f) },
25         func() { right = ScanUp(a, sum, 2*node+2, m, r, f) },
26     )

```

```

27     sum[node] = f(left, right)
28     return sum[node]
29 }
30
31 func ScanDown(
32     a, sum,
33     res []int,
34     node, l, r, fromLeft int,
35     f func(int, int) int,
36 ) {
37     if r-l == 1 {
38         res[l] = fromLeft + a[l]
39         return
40     }
41
42     m := (l + r) / 2
43
44     leftSum := sum[2*node+1]
45
46     Fork2Join(
47         func() { ScanDown(a, sum, res, 2*node+1, l, m, fromLeft, f) },
48         func() { ScanDown(a, sum, res, 2*node+2, m, r, fromLeft+leftSum, f) },
49     )
50 }

```

Задание 3

Вам нужно написать функцию $\text{eps}(a, S)$, которая получает положительных чисел a и возвращает минимальное j , такое что сумма элементов $a[1] + \dots + a[j] \geq S$, за $O(j)$ work и $O(\log^2 j)$ span. Предподсчётом пользоваться нельзя.

```

1 func EPS(a []int, S int) int {
2     l := 0
3     r := len(a)
4     result := r
5
6     for l <= r {
7         m := (l + r) / 2
8         sum := Reduce(0, m, a)
9
10        if sum > S {
11            result = m
12            r = m
13        } else {
14            l = m + 1
15        }
16    }
17    return result
18 }

```

Задание 4

Опишите алгоритм нахождения всех простых до N за $O(N \log \log N)$ work и $O(\text{polylog } N)$ span. (Желательно за $O(\log N \cdot \log \log N)$ span). Тут можно пользоваться `parallel_for`, `map`, `scan` и `filter`. Псевдокод даже необязательно.

Идея в том чтобы вычислять решето Эратосфена и с помощью `parallel_for` вычеркивать кратные числа, для этого создается булевый массив в котором помечаются числа которые не могут быть простыми:

1. Если $N=2$ возвращаем 2
 2. Ищем рекурсивно простые до \sqrt{N}
 3. Вычеркиваем с `parallel_for` все простые от \sqrt{N} до N
 4. Фильтруем и возвращаем простые
- Span

$$\text{глубина рекурсии } N^{\frac{1}{\text{Depth}}} = 2 \Rightarrow \text{Depth} = \log \log N$$

пометка составных и фильтрация простых требует $O(\log N) \Rightarrow \text{span} = O(\log N \cdot \log \log N)$

- Work

$$\text{work} = O(N \log \log N) - \text{обычная сложность решета эратосфена}$$