Problem Menu Back
Statement
Submissions

Your score not attempted

Spoilers
Show difficulty Show tags

solved by 29 / 36

dynamic programming: combinatorics

math: number theory

Top users by time

#	User	Time
1	Retr0Foxx	21 ms
2	bagasangga	23 ms
3	frennn	34 ms
4	mhasan01	42 ms
5	YukiSaki	43 ms

Top users by memory

#	User	Memory
1	Retr0Foxx	1096 KB
2	hitsuuj	1452 KB
3	Raphela	1656 KB
4	YukiSaki	2124 KB
5	ftri	2316 KB

Submission #4744450

Arkavidia 9.0 - Penyisihan CP / M. Main GCD

Accepted • LilyFlower • Go • January 9, 2026 at 18:10:07

▸ Sample Test Data Results

▸ Test Data Results

```go
package main

import (
    "bufio"
    "fmt"
    "math/bits"
    "os"
)

var (
    in      = bufio.NewReader(os.Stdin)
    out     = bufio.NewWriter(os.Stdout)
    println = Println
    scan    = Scan
    print   = Print
)

func gcd(a, b int64) int64 {
    if a == 0 || b == 0 {
        return a | b
    }
    shift := uint(bits.TrailingZeros64(uint64(a | b)))
    a >>= bits.TrailingZeros64(uint64(a))
    for {
        b >>= bits.TrailingZeros64(uint64(b))
        if a > b {
            a, b = b, a
        }
        b -= a
        if b == 0 {
            break
        }
    }
    return a << shift
}

func modPow(base, exp, mod int64) int64 {
    result := int64(1)
    base %= mod
    for exp > 0 {
        if exp&1 == 1 {
            result = (result * base) % mod
        }
        base = (base * base) % mod
        exp >>= 1
    }
    return result
}

func Run() {
    var n int64
    scan(&n)

    A := make([]int64, n)
    maxVal := int64(0)
    freq := make([]int64, 200001)

    for i := range A {
        scan(&A[i])
        freq[A[i]]++
        if A[i] > maxVal {
            maxVal = A[i]
        }
    }

    mod := int64(998244353)

    cnt := make([]int64, maxVal+1)
    for d := int64(1); d <= maxVal; d++ {
        for multiple := d; multiple <= maxVal; multiple += d {
            cnt[d] += freq[multiple]
        }
    }

    pairs := make([]int64, maxVal+1)
    for d := maxVal; d >= 1; d-- {
        if cnt[d] >= 2 {
            pairs[d] = (cnt[d] * (cnt[d] - 1) / 2) % (mod - 1)
            for multiple := 2 * d; multiple <= maxVal; multiple += d {
                pairs[d] = (pairs[d] - pairs[multiple] + (mod - 1)) % (mod - 1)
            }
        }
    }

    x := int64(1)
    for d := int64(1); d <= maxVal; d++ {
        if pairs[d] > 0 {
            x = (x * modPow(d, pairs[d], mod)) % mod
        }
    }

    println(x)
}

func Solve() {
    Run()
}

func Print(a ...any) {
    fmt.Fprint(out, a...)
}
```

```
102
103 func Println(a ...any) {
104     fmt.Fprintln(out, a...)
105 }
106
107 func Scan(a ...any) {
108     fmt.Fscan(in, a...)
109 }
110
111 func main() {
112     defer out.Flush()
113
114     Solve()
115 }
```