Pitagora trijnieki

Pitagora trijnieks – tie ir trīs naturālie skaitļi $a < b < c$, kuram ir izpildīts nosacījums

$$a^2 + b^2 = c^2$$

Uzdevums: saskaitīt Pitagora trijniekus, kuras vērtības ir mazākas par uzdoto `max`.

1. versija

```
int countPythagoreanTriples(const int max) {
    int cnt = 0;
    for (int a = 1; a <= max; a++) {
        for (int b = 1; b <= max; b++) {
            for (int c = 1; c <= max; c++) {
                if (a * a + b * b == c * c && a < b && b < c) {
                    cnt++;
                }
            }
        }
    }
    return cnt;
}
```

max = **5000** → result = **5681**, time = **4 min**

Pitagora trijnieki

2. versija

```
int countPythagoreanTriples(const int max) {
    int cnt = 0;
    for (int a = 1; a <= max; a++) {
        for (int b = a + 1; b <= max; b++) {
            for (int c = b + 1; c <= max; c++) {
                if (a * a + b * b == c * c) {
                    cnt++;
                }
            }
        }
    }
    return cnt;
}
```

max = **5000** → result = **5681**, time = **40 sec**

Pitagora trijnieki

3. versija

```
int countPythagoreanTriples(const int max) {
    int cnt = 0;
    for (int a = 1; a <= max; a++) {
        for (int b = a + 1; b * b <= max * max - a * a; b++) {
            for (int c = b + 1; c * c <= a * a + b * b; c++) {
                if (a * a + b * b == c * c) {
                    cnt++;
                }
            }
        }
    }
    return cnt;
}
```

max = **5000** → result = **5681**, time = **8 sec**

Pitagora trijnieki

4. versija

```
int countPythagoreanTriples(const int max) {
    int cnt = 0;
    for (int a = 1; a <= max; a++) {
        for (int b = a + 1; b * b <= max * max - a * a; b++) {
            int c = round(sqrt(a * a + b * b));
            if (a * a + b * b == c * c) {
                cnt++;
            }
        }
    }
    return cnt;
}
```

max = **5000** → result = **5681**, time = **0.5 sec**

Mājas uzdevums: precīza izmaksa

Pircējam ir monētas ar trijām vērtībām 2, 5, 9. Vajag izmaksāt summu `sum` bez atlikuma.

Uzdevumi:

1. Saskaitīt dažādas izmaksas variantu daudzumu.
2. Atrast optimālo izmaksas veidu ar minimālu kopīgo monētu skaitu.

Piemēram:

```
sum = 3018 → 50870 variantus, optimālais 1×2+2×5+334×9

sum = 5759 → 184768 variantus, optimālais 1×2+3×5+638×9
```