Задача 5

3 октября 2025 г.

Условие:

Дан треугольник (в виде списка списков, где $triangle[i]$ — это строка с $i+1$ элементом). Найди минимальную сумму пути от вершины треугольника до его основания. На каждом шаге ты можешь переместиться на соседнее число в строке ниже. Если ты находишься на индексе i в текущей строке, ты можешь перейти на индекс i или индекс $i+1$ в следующей строке. Предложи решение с алгоритмической сложностью, не превышающей $O(n^2)$.

Решение:

Условие можно переформулировать в постановке задачи динамического программирования.

Формализация состояний:

Пусть состояние (i, j) обозначает ячейку в строке i и столбце j, где $i = 0, 1, 2, n-1$; $j = 0, 1, ..., i$.

Функция Беллмана:

Пусть $S(i, j)$ – минимальная сумма пути от ячейки (i, j) до любой ячейки основания. Тогда уравнение Беллмана имеет вид:

$$S(i,j) = \begin{cases} triangle[i][j] & \text{, если } i = n-1 \text{(основание)} \\ triangle[i][j] + \min\{S(i+1,j), S(i+1,j+1)\} & \text{, иначе} \end{cases}.$$

Минимальному расстоянию от вершины треугольника до основания будет соответствовать значение $S(0, 0)$.

Алгоритм:

Алгоритм начинает проход по всем ячейкам треугольника для поиска минимального пути от каждой вершины до основания, начиная с предпоследней строки. Результаты записываются в исходный список, тем самым изменяя

Алгоритм 1 Листинг функции на языке Python 3

```
def minimum_total(triangle: list[list[int]])->int:
    rowsCount = len(triangle)
    # идём снизу вверх по строкам, начиная с предпоследней строки
    for i in range(rowsCount - 2, -1, -1):
        for j in range(len(triangle[i])):
            # вычисляем значение функции Беллмана S(i,j)
            triangle[i][j] += min(triangle[i+1][j],
                                  triangle[i+1][j+1])
    return triangle[0][0]
```

значение входного параметра функции, переданного по ссылке. По завершению работы внешнего цикла возвращается значение самого первого элемента списка, который представляет собой $S(0,0)$, что соответствует описанию алгоритма, приведенного выше.

Сложность:

Приведем оценку асимптотической сложности алгоритма в зависимости от размера входного списка n. Оценка асимптотической сложности фактически сводится к расчету количества всех итераций двойного цикла, поскольку время работы функции внутри цикла для всех итераций можно оценить некоторой положительной константой, а также добавление константных членов вне цикла не повлияет на асимптотическое поведение. Запишем количество итераций цикла:

$$\sum_{i=0}^{n-1}\sum_{j=0}^{i} 1 = \sum_{i=0}^{n-1}(i+1) = \frac{n\cdot(n+1)}{2} = \Theta(n^2), \text{ при } n \to \infty.$$

Точная асимптотическая оценка данного алгоритма равна $\Theta(n^2)$.