

# Динамическое программирование

КУРС  
АЛГОРИТМЫ  
И СТРУКТУРЫ  
ДАННЫХ

СПИКЕР  
Немков Максим Юрьевич

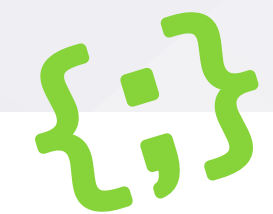
# Содержание темы

→ Динамическое программирование: основные принципы и задачи

→ Реализация динамического программирования в Python

→ Решение задач оптимизации с помощью динамического программирования

# Динамическое программирование DP



это метод решения сложных задач путём разбиения их на более простые подзадачи, результаты которых сохраняются и используются для решения исходной задачи. Этот подход особенно полезен для оптимизации задач, которые имеют перекрывающиеся подзадачи и оптимальные подструктуры



# Реализация динамического программирования в Python

В Python динамическое программирование можно реализовать различными способами, включая использование **массивов, списков или словарей** для хранения промежуточных результатов

*Рассмотрим примеры реализации  
для двух классических задач*



# Решение задач оптимизации с помощью динамического программирования

Задачи оптимизации часто могут быть решены с использованием динамического программирования для нахождения оптимального решения из всех возможных вариантов

## Основные примеры включают

Задачи на графах

Задачи оптимизации ресурсов

Задачи по последовательностям



# Решение задач оптимизации с помощью динамического программирования

## Пример 1

задача о наименьших суммарных путях в графе (Shortest Path Problem)

```
import sys
def shortest_path(graph, start):
    n = len(graph)
    dist = [float('inf')] * n
    dist[start] = 0
    visited = [False] * n

    for _ in range(n):
        u = min((dist[v], v) for v in range(n) if not visited[v])[1]
        visited[u] = True

        for v in range(n):
            if graph[u][v] and not visited[v]:
                dist[v] = min(dist[v], dist[u] + graph[u][v])

    return dist
```


```
# Пример использования:
graph = [
    [0, 2, 0, 1],
    [2, 0, 3, 2],
    [0, 3, 0, 4],
    [1, 2, 4, 0]
]
start = 0
print(shortest_path(graph,
start)) # Вывод: [0, 2, 5, 1]
```

# Решение задач оптимизации с помощью динамического программирования

## Пример 2

задача о вычислении чисел Фибоначчи

```
def fibonacci(n):  
    if n ≤ 1:  
        return n  
    dp = [0] * (n + 1)  
    dp[1] = 1  
  
    for i in range(2, n + 1):  
        dp[i] = dp[i - 1] + dp[i - 2]  
  
    return dp[n]
```

A yellow icon representing the Python programming language, consisting of a curly brace and a semicolon.

```
# Пример использования:  
print(fibonacci(10))  
# Вывод: 55
```

# ПОДВЕДЕМ ИТОГИ



Изучили динамическое программирование



Реализовали динамическое программирование на Python



Решили задачи с помощью динамического программирования