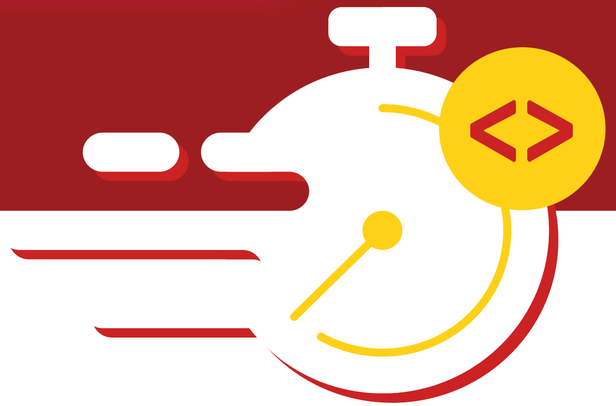
Линейное динамическое программирование. Задача о кузнечике

Урок 3.2.1



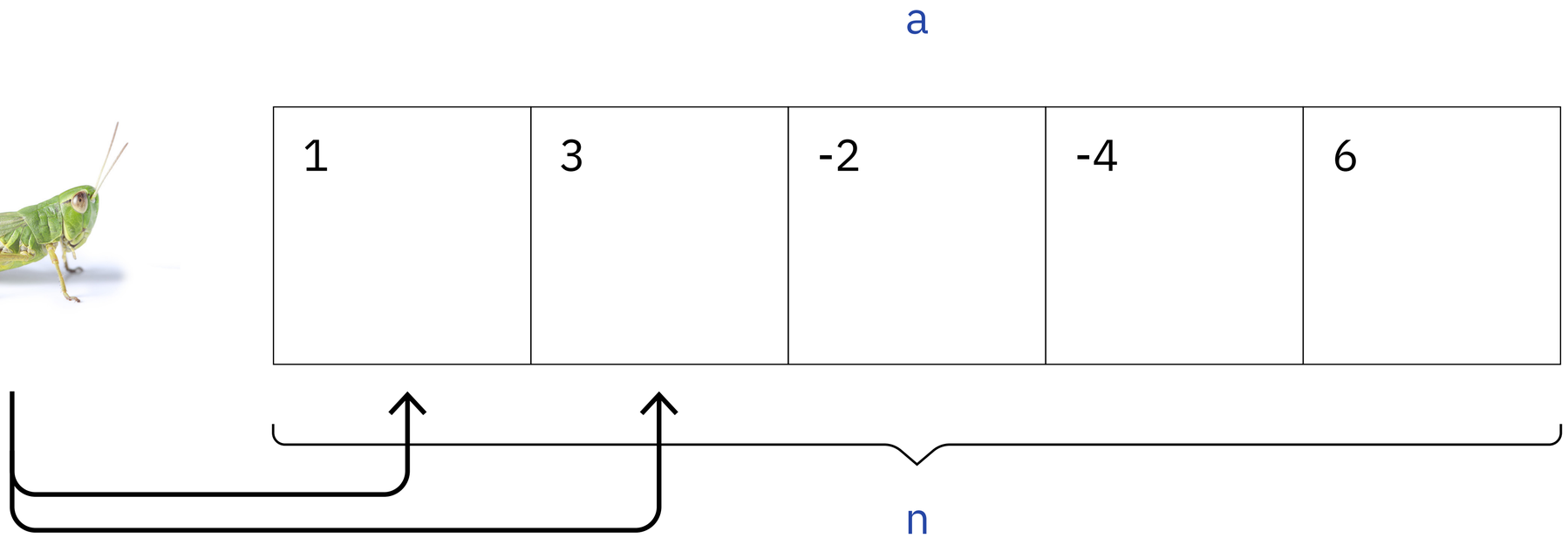
МФТИ
Московский физико-технический институт

При поддержке Фонда президентских грантов

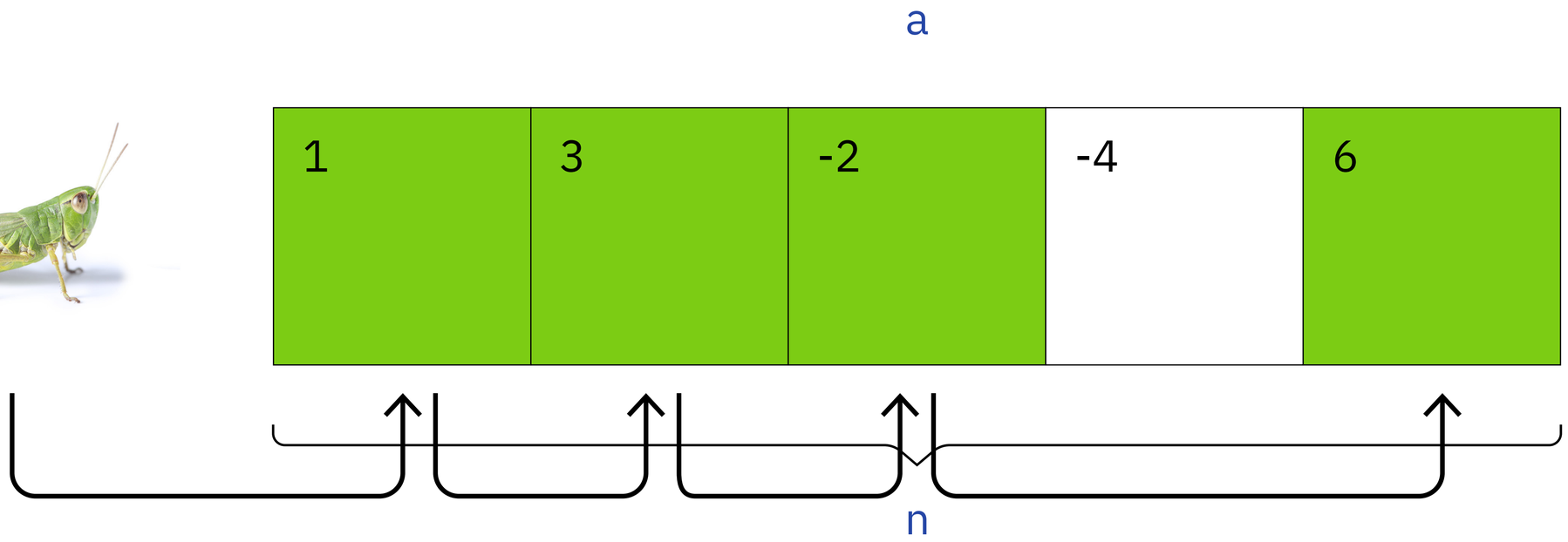
RUCODE festival

Фонд развития Физтех-школ

Постановка задачи_



Постановка задачи_

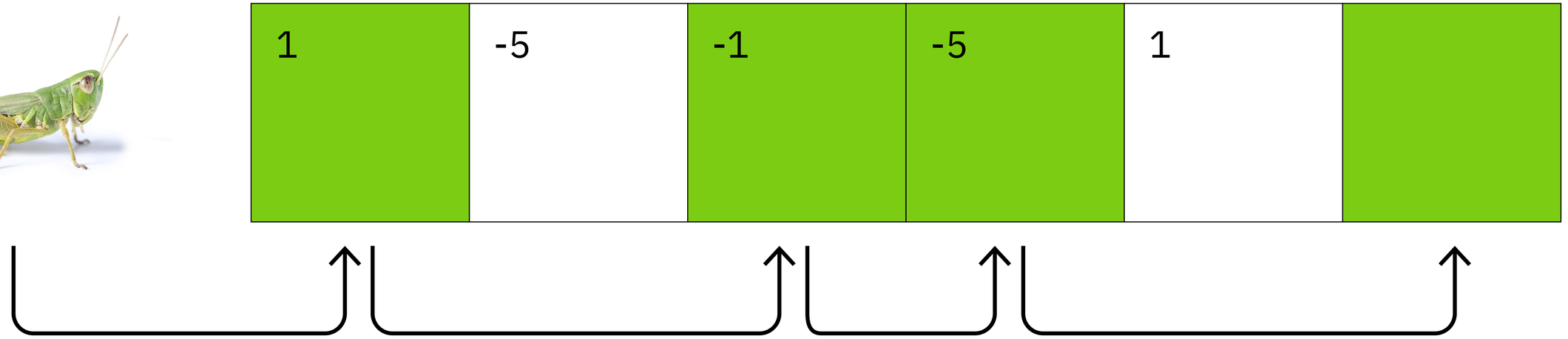


Жадный алгоритм_

1. Брать все положительное

2. Пропускать все отрицательное

Жадный алгоритм_



Решение_

1. Состояние
2. База
3. Формула
4. Порядок
5. Ответ

Решение_

1. **Состояние:** dp_i — максимальное количество конфет, которое можно собрать, добравшись до i-й клетки
2. **База**
3. **Формула**
4. **Порядок**
5. **Ответ**

Решение_

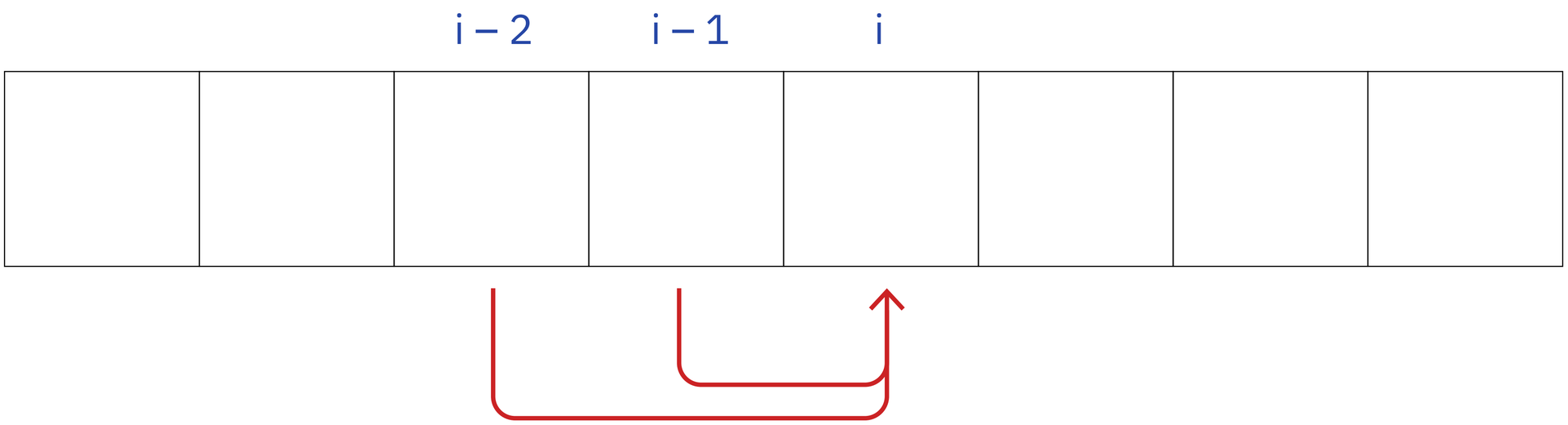
1. **Состояние:** dp_i — максимальное количество конфет, которое можно собрать, добравшись до i-й клетки
2. **База**
3. **Формула**
4. **Порядок**
5. **Ответ:** хранится в dp_n

Постановка задачи_

1	3	-2	-4	6

1	4	2	0	8

Решение_



$$dp_i = \max(dp_{i-1}, dp_{i-2}) + a_i$$

Решение_

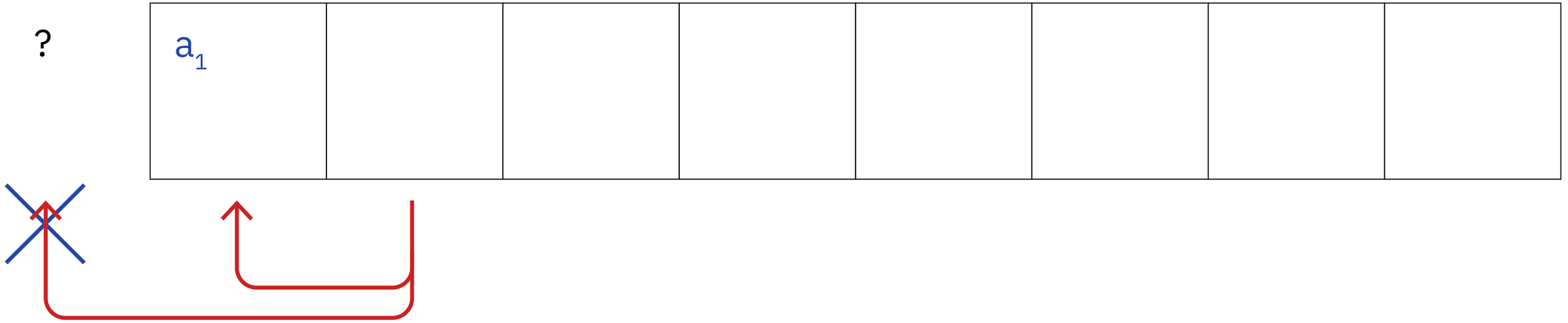
1. **Состояние:** dp_i — максимальное количество конфет, которое можно собрать, добравшись до i-й клетки
2. **База**
3. **Формула:** $dp_i = \max(dp_{i-1}, dp_{i-2}) + a_i$
4. **Порядок**
5. **Ответ:** хранится в dp_n

Решение_

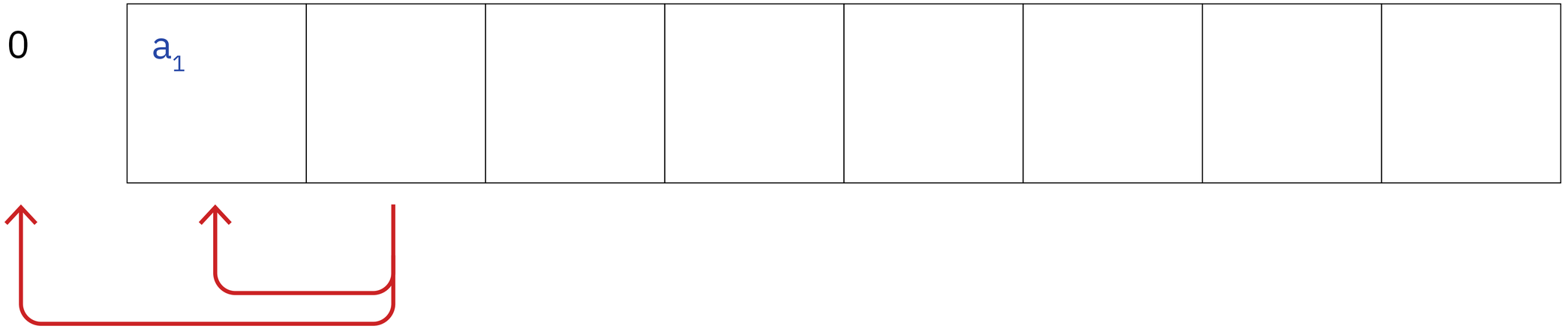
1. **Состояние:** dp_i — максимальное количество конфет, которое можно собрать, добравшись до i-й клетки
2. **База**
3. **Формула:** $dp_i = \max(dp_{i-1}, dp_{i-2}) + a_i$
4. **Порядок:** по возрастанию i
5. **Ответ:** хранится в dp_n

Решение_

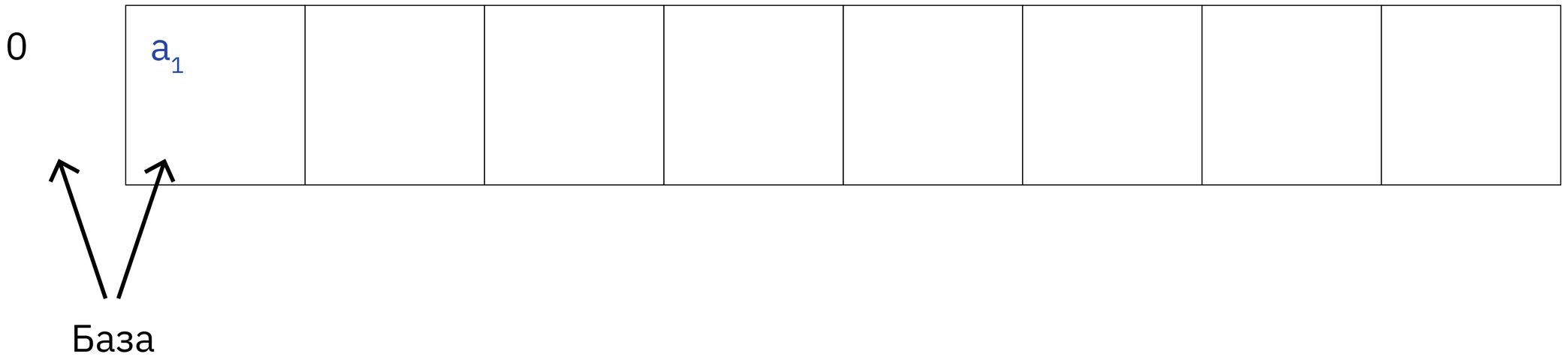
?



Решение_



Решение_



Решение_

1. **Состояние:** dp_i — максимальное количество конфет, которое можно собрать, добравшись до i-й клетки
2. **База:** $dp_1 = a_1$; $dp_0 = 0$
3. **Формула:** $dp_i = \max(dp_{i-1}, dp_{i-2}) + a_i$
4. **Порядок:** по возрастанию i
5. **Ответ:** хранится в dp_n

Реализация_

```cpp
int n;
int a[n + 1];
int dp[n + 1];

for (int i = 1; i <= n; ++i) {
    cin >> a[i];
} // Считываем массив a

dp[0] = 0;
dp[1] = a[1]; // База

for (int i = 2; i <= n; ++i) {
    dp[i] = min(dp[i - 1], dp[i - 2]) + a[i];
    // Наша формула
}

cout << dp[n] << endl; // Ответ
```

Решение_

1. **Состояние:** dp_i — максимальное количество конфет, которое можно собрать, добравшись до i-й клетки
2. **База:** $dp_1 = a_1$; $dp_0 = 0$
3. **Формула:** $dp_i = \max(dp_{i-1}, dp_{i-2}) + a_i$
4. **Порядок:** по возрастанию i
5. **Ответ:** хранится в dp_n

Следующее занятие — продвинутый кузнечик

МФТИ
Московский физико-технический институт

ПРИ ПОДДЕРЖКЕ
ФОНДА
ПРЕЗИДЕНТСКИХ
ГРАНТОВ

RUCODE
festival

Фонд развития
Физтех-школ