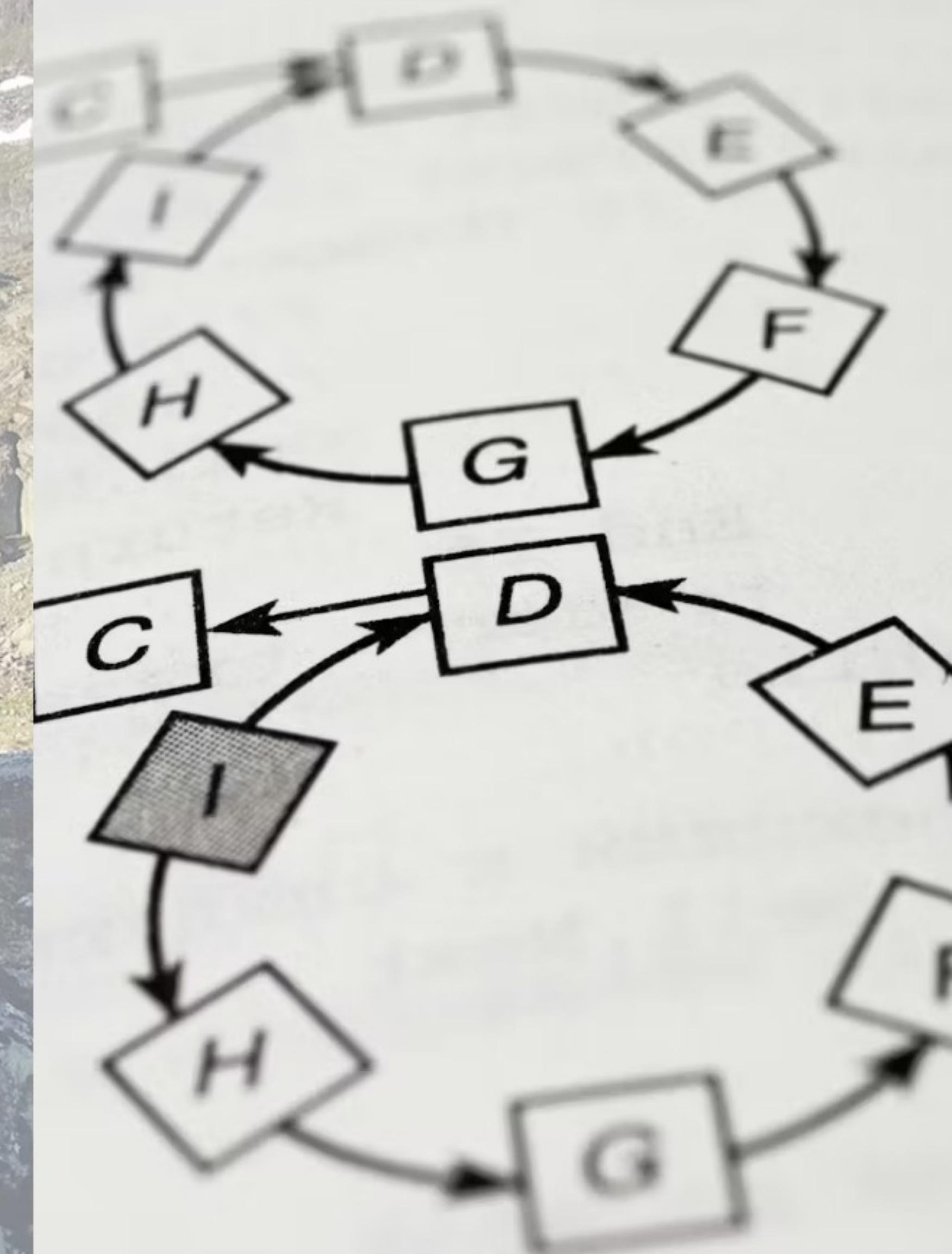
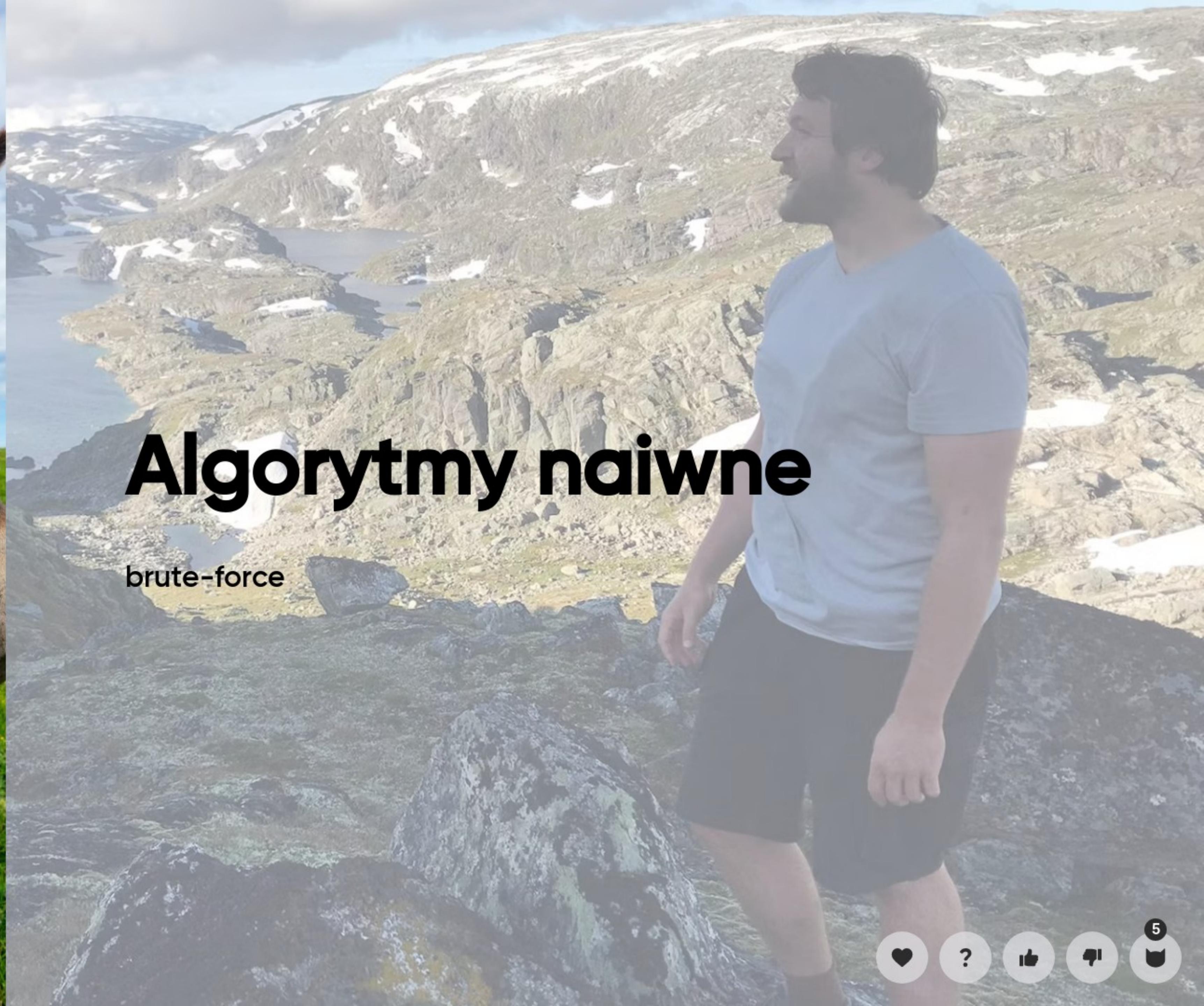
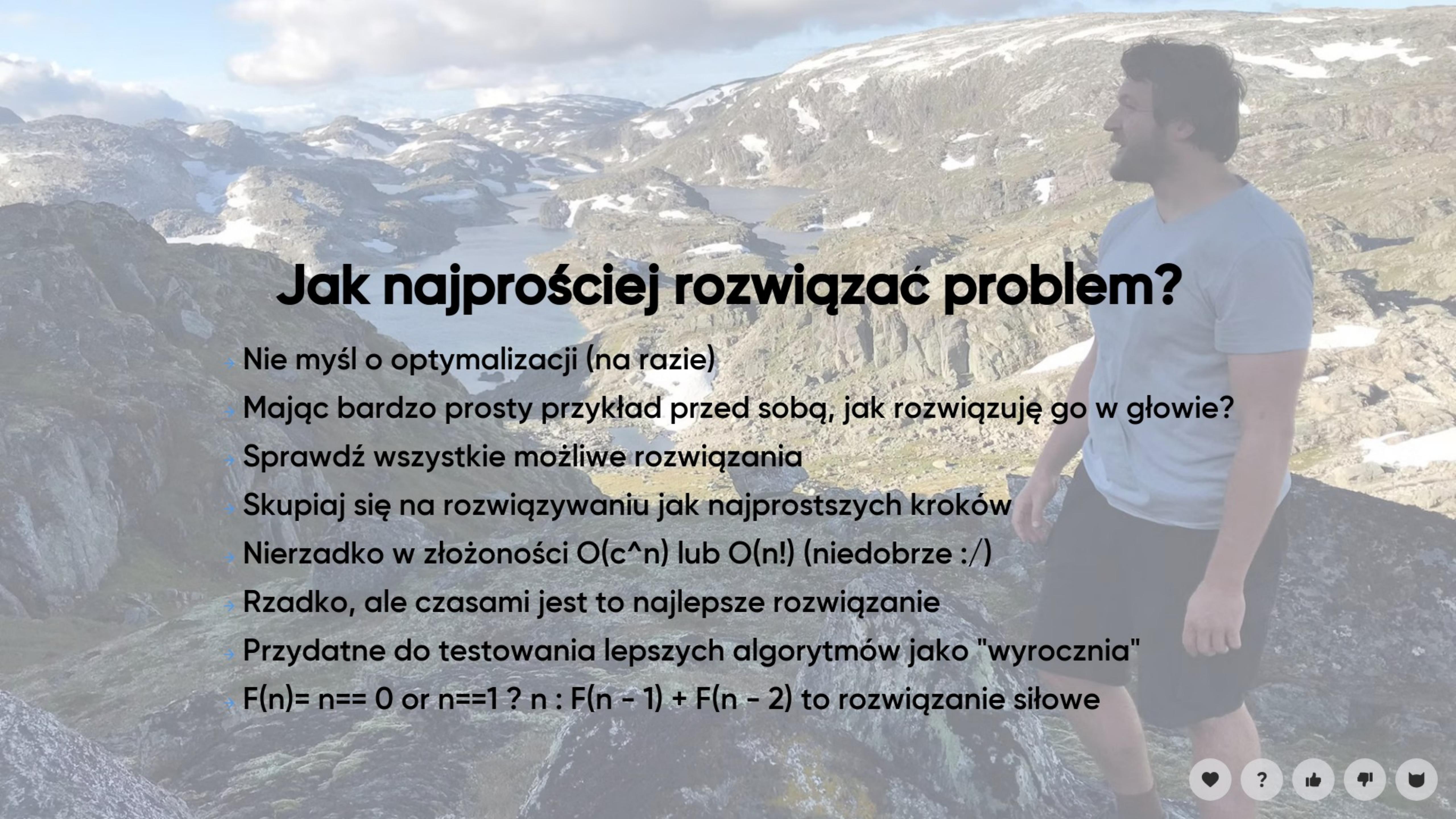




Algorytmy i Struktury Danych







Jak najprościej rozwiązać problem?

- Nie myśl o optymalizacji (na razie)
- Mając bardzo prosty przykład przed sobą, jak rozwiązuje go w głowie?
- Sprawdź wszystkie możliwe rozwiązania
- Skupiaj się na rozwiązywaniu jak najprostszych kroków
- Nierzadko w złożoności $O(c^n)$ lub $O(n!)$ (niedobrze :/)
- Rzadko, ale czasami jest to najlepsze rozwiązanie
- Przydatne do testowania lepszych algorytmów jako "wyrocznia"
- $F(n) = n == 0 \text{ or } n == 1 ? n : F(n - 1) + F(n - 2)$ to rozwiążanie siłowe



Problem sortowania

- Mamy tablicę elementów, jak uzyskać tablicę tych samych elementów od ułożonych od najmniejszego do największego?
- Dawno rozwiązywany i raczej nikt z was nie będzie się nim zajmował w praktyce
- Wciąż, jest świetny do pokazywania przykładów algorytmów

Sortowanie na siłę (bogosort)

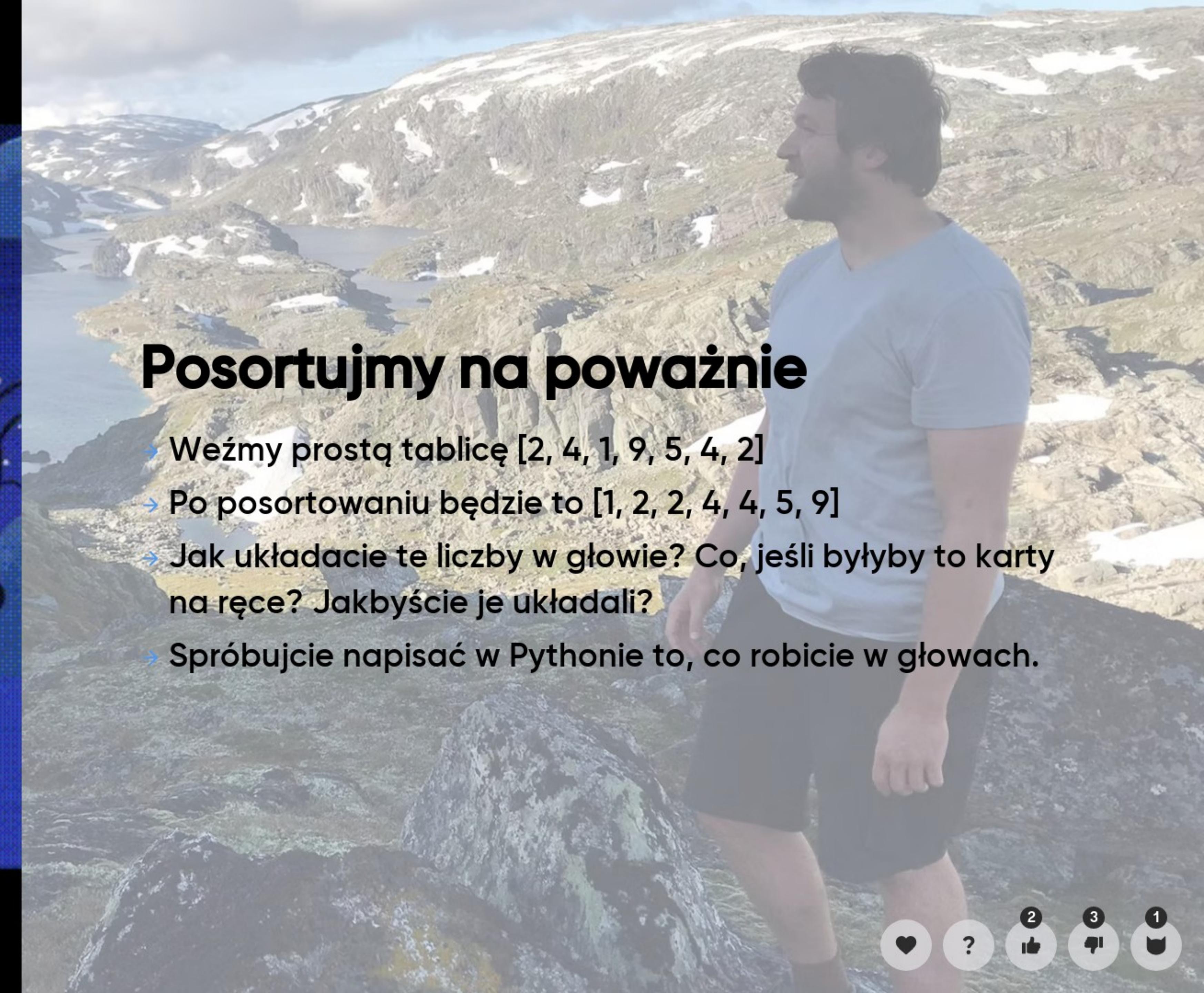
1. Sprawdź, czy tablica jest posortowana
2. Jeśli tak, to zakończ algorytm
3. Jeśli nie, przetasuj tablicę i powtórz algorytm

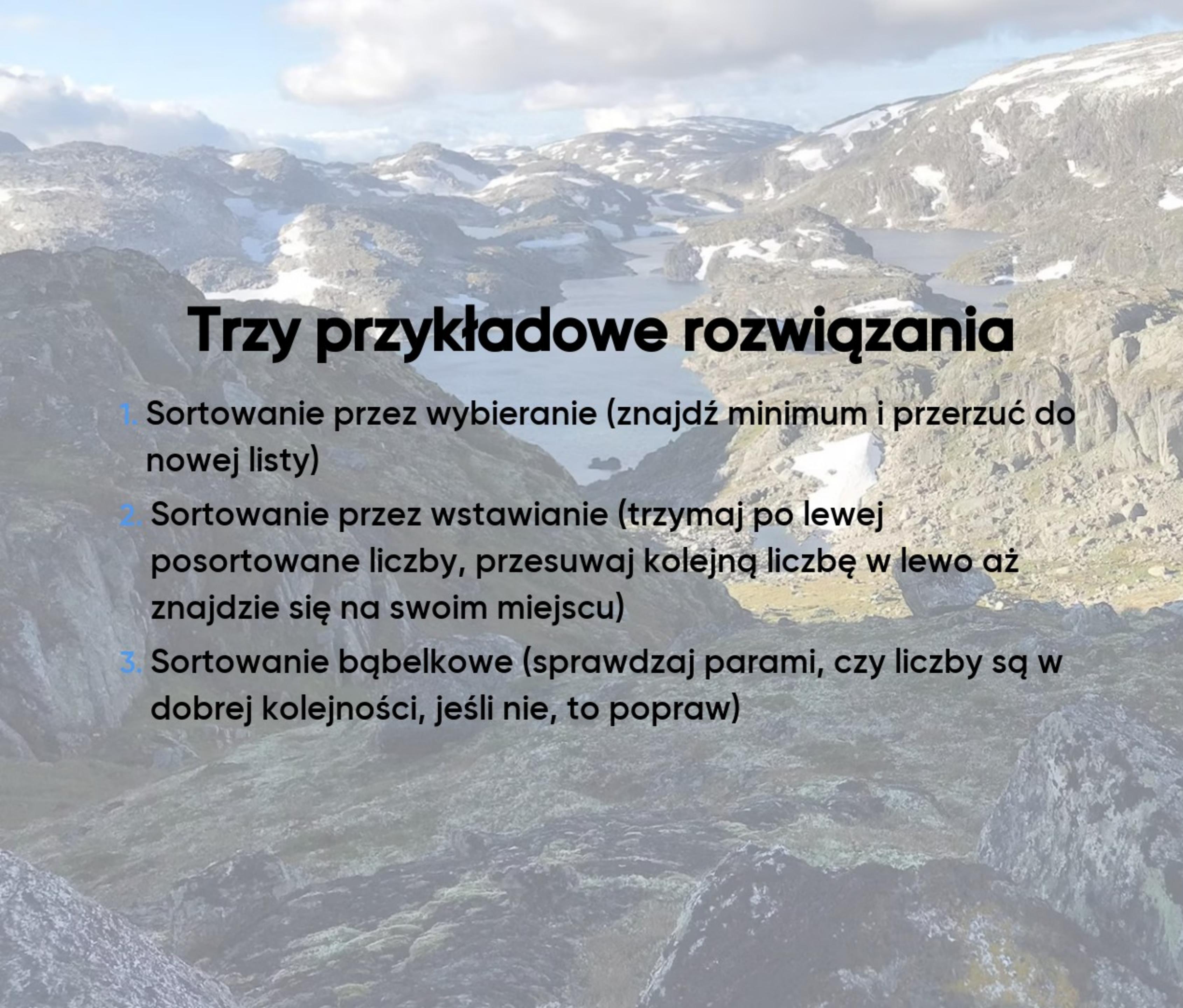
```
1 def is_sorted(data):  
2     return all([a <= b for a,b in zip(data, data[1:])])  
3  
4 def bogosort(data):  
5     while not is_sorted(data):  
6         shuffle(data)  
7     return data
```



Jaka jest złożoność bogosorta?





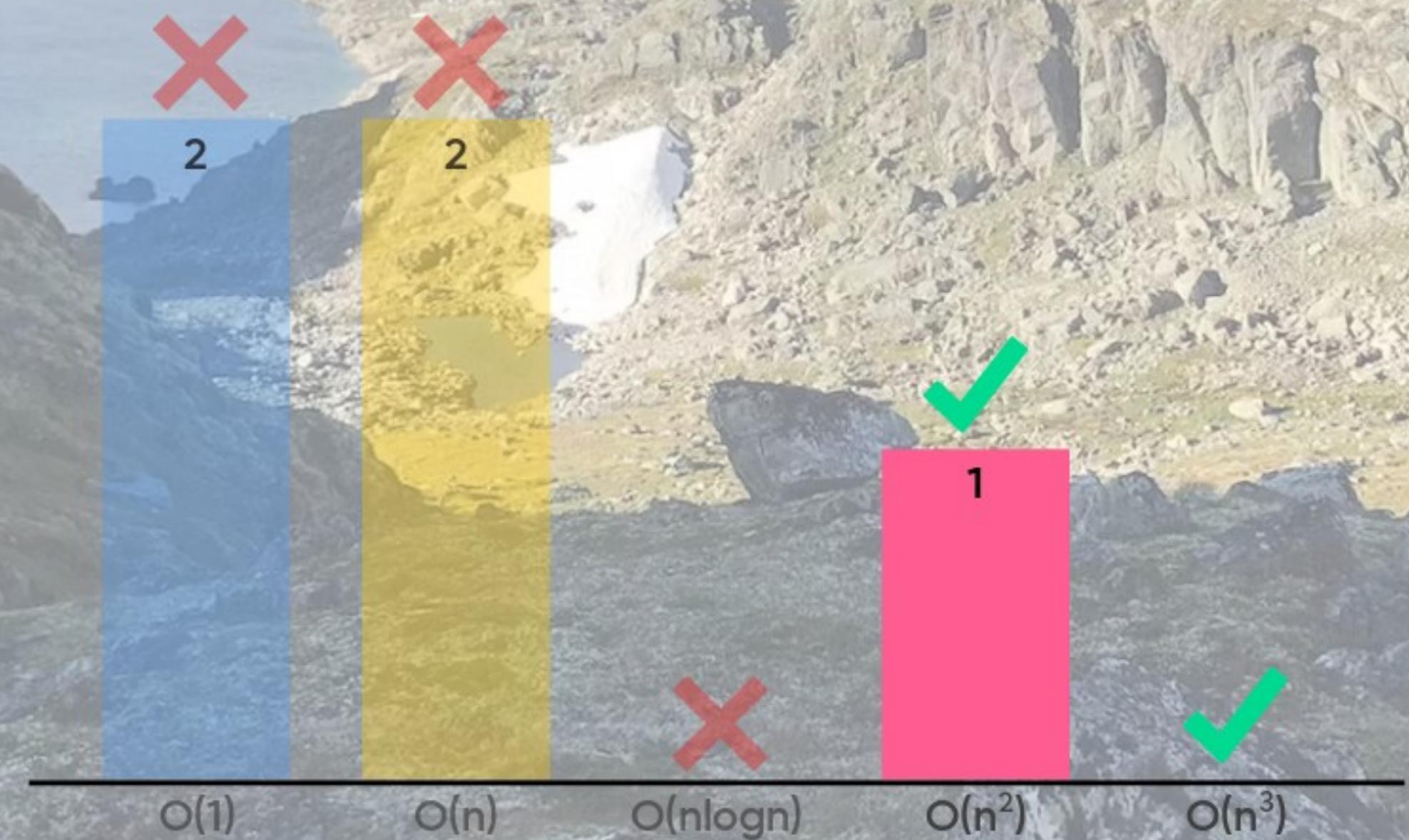


Trzy przykładowe rozwiązania

1. Sortowanie przez wybieranie (znajdź minimum i przerzuć do nowej listy)
2. Sortowanie przez wstawianie (trzymaj po lewej posortowane liczby, przesuwaj kolejną liczbę w lewo aż znajdzie się na swoim miejscu)
3. Sortowanie bąbelkowe (sprawdzaj parami, czy liczby są w dobrej kolejności, jeśli nie, to popraw)



Jaka jest złożoność tych rozwiązań dla tablic długości n ?



Jak się czujecie z tym tematem?

