**Хэширование** Есть задача сравнения объектов:

$$U = \{objects\},\ u, v \in U:\ u \neq v?$$

Введем функцию $h:\ U \to \mathbb{Z}_m$, такую что $\forall\ u, v \in U:\ u \sim v \to h(v) = h(u)$. Обычно $m \ll |U|$.
Зачем юзать хэши, а не наивное сравнение?

- Бывает много сравнений, и мы не хотим дублировать вычисления
- Безопасность
- Для некоторых хешей верно, что $h(f(v, u)) = g(h(v), h(u))$. То есть можно вычислить хэш от некоего объекта, пользуясь уже посчитанными хэшами для других объектов.
- Иногда мы сравниваем объекты не на равенство, а на изоморфность.
- Сравнивать объекты бывает дорого

Требования к хэшу:

- вычислим за линейное время
- детерменирован
- семейство хэш-функций $\mathbb{H}$ (и можем взять сколько угодно оттуда)
- равномерность $\forall_{v \neq u} v,\ u:\ p_{h \in \mathbb{H}}(h(u) = h(v)) \simeq \frac{1}{m}$
- масштабируемость
- необратимость
- (*optional*) лавинный эффект (при маленьком изменении объекта хэш меняется сильно)

Важно, что мы хотим брать случайную функцию из семейства $\mathbb{H}$ на момент старта программы, потому что псевдослучайная функция на самом деле нам дает детерменированный алгоритм, который неверен.

**Идеальная хэш-функция** Так как объектов счетно, то отсортируем их, далее для каждого объекта запомним случайную величину от 1 до $m$ (или даже можем запоминать ее лениво!).

**Полиномиальный хэш** Сводим объекты к строкам и хэшируем строки:

$$s = c_0 c_2 c_3 \ldots c_{n-1},\ c_i > 0$$

$$h_b(s) = \sum_{i=0}^{n-1} c_i \cdot b^i\ (mod\ m)$$

$$p_b(h_b(s_1) = h_b(s_2)) \leq \frac{n}{m} \text{ для фиксированного m}$$

Доказательство: Рассмотрим функцию как многочлен, теперь для равных функций смотрим на то, равна ли разность многочленов нулю для какого-то $b$. У многочлена от $b$ степень равна $n$, а вероятность попасть в конкретный модуль $\frac{1}{m}$.

$$h(s_1 + s_2) = h(s_1) + h(s_2) \cdot b^{|s_1|}$$