**05. Operation Data Breach**

**Условие:**

Кибератака от неизвестна хакерска група е компрометирала база данни, съдържаща класифицирана информация. Всеки запис в базата представлява криптиран фрагмент от данни, съхраняван в йерархична структура, наречена **"Криптографско дърво"**.

Криптографското дърво е подобно на стандартно дърво, но със специфични правила:

1. Всяка възлова точка съдържа криптиран запис.
2. Всяко дете зависи от своя родител чрез ключ за декриптиране, който се извлича на базата на състоянието на родителя.
3. Коренът на дървото е единственият възел, който може да бъде декриптиран директно с "Главен ключ".

Хакерската група е успяла да добие "Главния ключ" и криптографското дърво, но за да извлекат цялата информация, се нуждаят от автоматизиран процес за декриптиране. Вашата задача е да помогнете на екипа по киберсигурност да възстанови компрометираната база данни, като декриптирате всички възли в дървото.

**Правила за декриптиране:**

1. Всеки възел има уникален идентификатор **ID** и елемент **E**, който съдържа криптирания запис.
2. Декриптирането на елемент **E** изисква следните стъпки:
   * Ако възелът е корен (няма родител), използвайте "Главния ключ".
   * Ако възелът има родител, генерирайте ключ от декриптирания елемент на родителя чрез хеширане със SHA256.
   * Декриптирайте елемента на възела, използвайки получения ключ.
3. Данните за дървото са представени в йерархичен формат, като всеки възел е описан с:
   * **ID** (уникален идентификатор).
   * **ParentID** (идентификатор на родителския възел; ако е -1, възелът е корен).
   * **E** (криптираният запис).
4. След декриптиране на възел, полученото съдържание трябва да се запише във възела за по-нататъшна употреба.

**Вход:**

1. На първия ред е даден **N** (1 ≤ **N** ≤ 10,000) – броят на възлите в дървото.
2. Следват **N** реда, всеки съдържащ:
   * **ID** (1 ≤ **ID** ≤ 10,000).
   * **ParentID** (или -1, ако е корен).
   * **E** (криптирания запис като низ).
3. Последният ред съдържа **MasterKey** – главният ключ за декриптиране на коренния възел.

**Изход:**

Изведете декриптираната информация за всички възли, подредени по техните идентификатори.

**Пример:**

|  |  |
| --- | --- |
| **Вход** | **Изход** |
| 5  1 -1 "f1e2d3c4"  2 1 "a1b2c3d4"  3 1 "e4d3c2b1"  4 2 "abc123ef"  5 3 "efc1b2a3"  "key\_master\_001" | 1: "data\_root"  2: "data\_child1"  3: "data\_child2"  4: "data\_leaf1"  5: "data\_leaf2" |

**Допълнителни условия:**

* Гарантирано е, че входът винаги ще представлява валидно дърво.
* Хеширането със SHA256 трябва да се извършва върху низовете, без допълнителни форматирания.
* Декриптирането на елементи може да бъде емулирано чрез функция decrypt(E, Key) (сигурна библиотечна функция).
* Вашето решение трябва да бъде ефективно, за да работи с големи входни данни.

**Указания за решаване:**

1. **Изграждане на дървото:**
   * Организирайте възлите в йерархична структура, използвайки ParentID.
2. **Декриптиране:**
   * Започнете от корена, декриптирайте елемента му с главния ключ.
   * Рекурсивно преминете към децата, генерирайки ключове чрез SHA256 от вече декриптираните елементи на родителите.
3. **Рекурсия или BFS:**
   * Използвайте рекурсия или обхождане в ширина, за да обработите всички възли.
4. **Ефективност:**
   * Организирайте данните във формат за бързо намиране на децата на всеки възел (напр. хеш таблици).