Digital Pakistan Speed Programing Competition Online Mock Contest

Instructions

- Do not open the booklet unless you are explicitly told to do so. You can only read these instructions below.
- If you have any question regarding the problems, seek a clarification from the judges using DOMJudge.
- Before submitting a run, make sure that it is executable via command line. For Java, it must be executable via "javac" and for GNU C++ via "g++". Java programmers need to remove any "package" statements and source code's file name must be the same as of main class. C++ programmers need to remove any getch() / system("pause") like statements.
- Do not attach input files while submitting a run, only submit/attach source code files, i.e., *.java or *.cpp or *.py.
- Language supported: C/C++, Java and Python3
- Source code file name should not contain white space or special characters.
- You must take input from Console i.e.: Standard Input Stream (stdin in C, cin in C++, System.in in Java, stdin in Python)
- You must print your output to Console i.e.: Standard Output Stream (stdout in C, cout in C++, System.out in Java)
- Please, don't create/open any file for input or output.
- Please strictly meet the output format requirements as described in problem statements, because your program will be auto judged by computer. Your output will be compared with judge's output byte-by-byte and not tolerate even a difference of single byte. So, be aware! **Pay special attention to spaces, commas, dots, newlines, decimal places, case sensitivity etc.**
- All your programs must meet the time constraint specified.
- The decision of judges will be absolutely final.

Problem 06: Ancestral Queries

Time limit: 1 second

The Khan family submitted their digital family tree to the Lahore Heritage Society. The judges were impressed by how efficiently the family had documented their ancestry using algorithmic thinking. They even asked Zara to present her solution at the Lahore Tech Expo, where it became a hit among historians and tech enthusiasts alike.

The Khan family tree looked like this:

- **Dada Abu**: The patriarch of the family.
  - His children: **Ali** and **Fatima**.
    - Ali's children: **Hassan** and **Ayesha**.
      - Hassan's children: **Bilal** and **Sana**.
        - Bilal's children: **Amna** and **Usman**.
    - Fatima's children: **Zainab** and **Omar**.
      - Zainab's children: **Leena** and **Yusuf**.
      - Omar's children: **Hania** and **Saad**.

With the digital family tree in place, the Khan family organized a **grand reunion** at their ancestral home in Lahore's **Old City**. Family members from all over Pakistan gathered to celebrate their shared history and learn more about their ancestors.

- **Amna** discovered that her 4th ancestor was **Dada Abu**, the patriarch of the family.
- **Leena** learned that her 2nd ancestor was **Fatima**, her great-grandmother.
- **Saad** was amazed to find out that his 3rd ancestor was **Ali**, a prominent businessman in Lahore's history.

The Khan family's story became a symbol of **unity, heritage, and innovation** in Lahore. Their digital family tree not only helped them connect with their past but also inspired other families in the city to document their own histories using technology. As Dada Abu often said, **"A family's roots are like the foundations of a building. The stronger they are, the taller we can rise."**

The story highlights the importance of family, heritage, and technology in preserving our connections to the past. Just like the binary lifting technique helps us efficiently find ancestors, understanding our roots helps us navigate the present and future with clarity and purpose.

Description:

Given a family tree of n members and q queries, for each query, find the ***k-th*** ancestor of a given family member. If the ***k-th*** ancestor does not exist, return -1.

Input:

- The first line contains the number of testcases (***t***).
- The second line contains n (number of family members), q (number of queries) and r (root of tree).
- The next $n-1$ lines describe the parent-child relationships in the family tree.
- The next q lines contain pairs (u, k), where u is the family member and k is the ancestor level.
- All numbers in a line are single space separated.

Output:

- For each query, print the ***k-th*** ancestor of u. If it doesn't exist, print **-1**.

Input	Output
1 5 2 5 5 3 1 3 4 3 1 2 4 2 1 3	5 -1