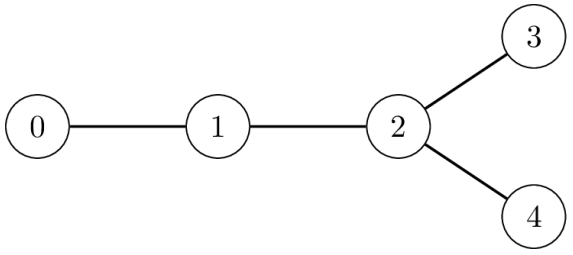
Migrations

The Natural History Museum is investigating the migration patterns of dinosaurs in Bolivia. Paleontologists have discovered dinosaur footprints at N different sites, numbered from 0 to $N-1$ in order of decreasing age: site 0 contains the oldest footprints, while site $N-1$ holds the youngest.

The dinosaurs migrated to each site (except site 0) from a specific, older site. For every site i such that $1 \leq i \leq N-1$, there exists exactly one older site $P[i]$ (with $P[i] < i$) such that some dinosaurs migrated directly from site $P[i]$ to site i. A single older site may have been the source of migrations to multiple younger sites.

The paleontologists model each migration as an **undirected connection** between sites i and $P[i]$. Note that for any two distinct sites x and y, it is possible to travel from x to y by traversing a sequence of connections. The **distance** between sites x and y is defined as the minimum number of connections required to travel from x to y.

For example, the following image displays a case with $N = 5$ sites, where $P[1] = 0$, $P[2] = 1$, $P[3] = 2$, and $P[4] = 2$. One can travel from site 3 to site 4 through 2 connections, so the distance between them is 2.



The Museum aims to identify a pair of sites with the maximum possible distance.

Note that the pair is not necessarily unique: for instance, in the example above, both pairs $(0, 3)$ and $(0, 4)$ have a distance of 3, which is the maximum. In such cases, any of the pairs attaining the maximum distance is considered valid.

Initially, the values of $P[i]$ are **not** known. The Museum sends a research team to visit the sites $1, 2, \ldots, N-1$, in order. Upon reaching each site i ($1 \leq i \leq N-1$), the team performs both of the following actions:

- Determines the value of $P[i]$, that is, the source of migration to site i.
- Decides whether to send **one** message to the Museum or not to send any message from this site, based on the previously gathered information.

Messages are transmitted via an expensive satellite communication system, so each message must be an integer between 1 and $20\,000$, inclusive. Additionally, the team is allowed to send **at most 50 messages** in total.

Your task is to implement a strategy, through which:

- The research team selects the sites from which messages are sent, along with the value of each message.
- The Museum can determine a pair of sites with the maximum distance, based solely on the messages received from each site, knowing from which sites these messages were sent.

Sending large numbers through the satellite is costly. Your score will depend on both the highest integer sent and the total number of messages transmitted.

Implementation Details

You need to implement two procedures; one for the research team, and another one for the Museum.

The procedure you should implement for the **research team** is:

```
int send_message(int N, int i, int Pi)
```

- N: the number of sites with footprints.
- i: the index of the site the team is currently visiting.
- Pi: the value of $P[i]$.
- This procedure is called $N-1$ times for each test case, in the order of $i = 1, 2, ..., N-1$.

This procedure should return $S[i]$ specifying the action taken by the research team at site i:

- $S[i] = 0$: the research team decides not to send a message from site i.
- $1 \leq S[i] \leq 20\,000$: the research team sends the integer $S[i]$ as the message from site i.

The procedure you should implement for the **Museum** is:

```
std::pair<int,int> longest_path(std::vector<int> S)
```

- S: an array of length N such that:
  - $S[0] = 0$.
  - For each $1 \leq i \leq N-1$, $S[i]$ is the integer returned by `send_message(N, i, Pi)`.
- This procedure is called exactly once for each test case.

This procedure should return a pair of sites (U, V) with the maximum distance.

In the actual grading, a program that calls the above procedures is run **exactly two times**.

- During the first run of the program:
  - `send_message` is called exactly $N - 1$ times.
  - **Your program can store and retain information across successive calls.**
  - The returned values (array S) are stored by the grading system.
  - In some cases the behavior of the grader is **adaptive**. This means that the value of $P[i]$ in a call to `send_message` may depend on the actions taken by the research team during the previous calls.
- During the second run of the program:
  - `longest_path` is called exactly once. The only information available to `longest_path` from the first run is the array S.

Constraints

- $N = 10\,000$
- $0 \leq P[i] < i$ for each i such that $1 \leq i \leq N - 1$.

Subtasks and Scoring

Subtask	Score	Additional Constraints
1	30	Site 0 and some other site have a maximum distance among all pairs of sites.
2	70	No additional constraints.

Let Z be the maximum integer appearing in the array S, and let M be the number of messages sent by the research team.

In any of the test cases, if at least one of the following conditions hold, then the score of your solution for that test case will be 0 (reported as `Output isn't correct` in CMS):

- At least one element in S is invalid.
- $Z > 20\,000$ or $M > 50$.
- The return value of the call to procedure `longest_path` is incorrect.

Otherwise, your score for subtask 1 is calculated as follows:

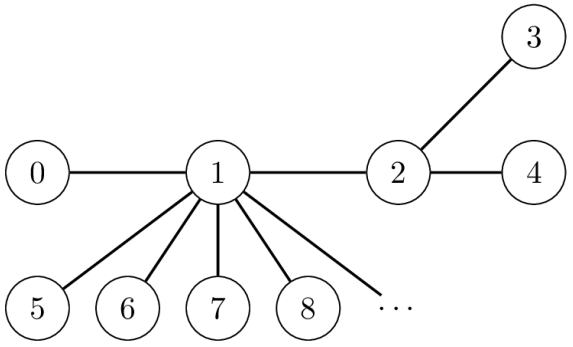
Condition	Score
$9\,998 \leq Z \leq 20\,000$	10
$102 \leq Z \leq 9997$	16
$5 \leq Z \leq 101$	23
$Z \leq 4$	30

Your score for subtask 2 is calculated as follows:

Condition	Score
$5 \leq Z \leq 20\,000$ and $M \leq 50$	$35 - 25 \log_{4000} \left(\frac{Z}{5}\right)$
$Z \leq 4$ and $32 \leq M \leq 50$	40
$Z \leq 4$ and $9 \leq M \leq 31$	$70 - 30 \log_4 \left(\frac{M}{8}\right)$
$Z \leq 4$ and $M \leq 8$	70

Example

Let $N = 10\,000$. Consider the situation in which $P[1] = 0$, $P[2] = 1$, $P[3] = 2$, $P[4] = 2$, and $P[i] = 1$ for $i > 4$.



Assume that the research team's strategy is to send the message $10 \cdot V + U$ whenever the pair of sites (U, V) with the maximum distance changes as a result of a `send_message` call.

Initially, the pair with the maximum distance is $(U, V) = (0, 0)$. Consider the following sequence of calls during the first run of the program:

Procedure call	(U, V)	Return value ($S[i]$)
`send_message(10000, 1, 0)`	$(0, 1)$	10
`send_message(10000, 2, 1)`	$(0, 2)$	20
`send_message(10000, 3, 2)`	$(0, 3)$	30
`send_message(10000, 4, 2)`	$(0, 3)$	0

Note that in all the remaining calls the value of $P[i]$ is 1. This means that the pair with the maximum distance does not change, and the team does not send any more messages.

Then, in the second run of the program, the following call is made:

```
longest_path([0, 10, 20, 30, 0, ...])
```

The Museum reads the last message sent by the research team, which is $S[3] = 30$, and deduces that $(0, 3)$ is the pair of sites with the maximum distance. Therefore, the call returns $(0, 3)$.

Note that this approach does not always make it possible for the Museum to determine the pair with the maximum distance correctly.

Sample Grader

The sample grader calls both `send_message` and `longest_path` in the same run, which is different from the actual grader.

Input format:

```
N
P[1] P[2] ... P[N-1]
```

Output format:

```
S[1] S[2] ... S[N-1]
U V
```

Note that you can use the sample grader with any value of N.