Piggy

Problem category: Logic
Expected difficulty: 800

Solution

First, let us consider the problem of finding Piggy. We can do it in at most $n-1$ queries:

1. Fix a position p and ask the oracle if Piggy is at that position $n-1$ times;
2. If Piggy was not found, then it will be at position p after all those queries.

However, in the worst case, this does not help us solve the original problem. Since there would be one query left, that query alone cannot be used to determine Piggy's direction.

Let us then save one query and see what we can do with it. Suppose that we make $n-2$ queries at position 1. There will be four cases in which Piggy is not found:

Direction	Starting position	Final position
forwards	2	n
forwards	3	1
backwards	n	2
backwards	$n-1$	1

We can check if Piggy is moving forwards by querying for position n. If this fails, and Piggy were moving forwards, then it was at position 1 and would now be at position 2. So we query for position 2 and, if this also fails, we can conclude that Piggy is moving backwards.

Note that if Piggy were moving backwards, none of the above queries would find it: in the second-to-last query (at n), it would move from 2 to 1 or from 1 to n, neither of which is the last query's position.

Code

```
bool query(int p) {
  cout << "? " << p << endl;
  bool ans; cin >> ans;
  return ans;
}

void solve() {
  int n; cin >> n;
  bool found = false;
  for (int rem = n; rem > 2 && !found; rem--) {
    found = query(1);
  }
  bool ans = !found && query(n) || query(2);
  cout << "! " << ans  << endl;
}
```