You have a singly-linked list and want to check if it contains a cycle.

A singly-linked list is built with nodes, where each `node` has:

- `node.next`—the next node in the list.
- `node.value`—the data held in the node. For example, if our linked list stores people in line at the movies, `node.value` might be the person's name.

For example:

```
class LinkedListNode(object):


    def __init__(self, value):
        self.value = value
        self.next  = None
```

Python 3.6

A **cycle** occurs when a node's `next` points *back to a previous node in the list*. The linked list is no longer linear with a beginning and end—instead, it cycles through a loop of nodes.

Write a function `contains_cycle()` that takes the first node in a singly-linked list and returns a boolean indicating whether the list contains a cycle.