❖ Intersection of List:

- Explanation:
  The program identifies the first common element between two linked lists, which indicates their intersection point. It uses a HashSet to store values from the first list and checks for these values while traversing the second list. If a common value is found, it outputs that value; otherwise, it indicates that there is no intersection.
- Time Complexity and Space Complexity:
  - O(m+n) O(m)
- Flowchart:
  [Start]
  |
  [Input Linked List A]
  |
  [Input Linked List B]
  |
  [Create empty HashSet]
  |
  [Set Current A to Head of List A]
  |
  [Current A is not null?] -- No --> [Set Current B to Head of List B]
  | Yes
  |
  [Add Current A's value to HashSet]
  |
  [Move Current A to Current A's next]
  |

[Back to Current A is not null?]

[Set Current B to Head of List B]
    |
[Current B is not null?] -- No --> [Output: "No intersection found."]
    | Yes
    |
[Is HashSet contains Current B's value?] -- No --> [Move Current B to Current B's next]
    | Yes
    |
[Output: "The first intersection value is: Current B's value"]
    |
[End]