Problem 1 (8 points). Prove the following statements:

(a) Show that \( \lfloor \sqrt{n} \rfloor = \Theta(\sqrt{n}) \).

(b) Let \( f : \mathbb{N} \to \mathbb{R} \) be an asymptotically positive function. Prove that \( o(f(n)) \subseteq O(f(n)) \).

(c) Prove that \( \log(n!) = \Theta(n \log n) \).

Problem 2 (10 points). Given a sorted array \( A \) of size \( n \), design an algorithm to check if there are two numbers in the array whose sum is 0. That is, decide whether there are two indices \( i, j \in \{1, 2, 3, \ldots, n\} \) such that \( A[i] + A[j] = 0 \). (The two indices can be the same; thus if the array contains the number 0, we should output “yes”.)

Example: if the input is \((-8, -5, -2, 1, 4, 6, 8, 9)\), then the output is “yes” since \((-8) + 8 = 0\). If the input is \((-8, -5, -2, 1, 4, 6, 7, 9)\), then the output is “no”.

(a) (2 points) How can we use the binary search algorithm as a black-box to design an \( O(n \log n) \)-time algorithm?

(b) (8 points) Design an \( O(n) \)-time algorithm for the problem.

Problem 3 (12 points).

Figure 1: Cycles in undirected and directed graphs. \((1, 2, 5, 3)\) is a cycle in the undirected graph. \((1, 2, 5, 6, 7, 3)\) is a cycle in the directed graph. However, \((1, 2, 5, 8, 3)\) is not a cycle in the directed graph.

(a) (6 points) A cycle in an undirected graph \( G = (V, E) \) is a sequence of \( t \geq 3 \) different vertices \( v_1, v_2, \ldots, v_t \) such that \((v_i, v_{i+1}) \in E\) for every \( i = 1, 2, \ldots, t - 1 \) and \((v_t, v_1) \in E\). Given the linked-list representation of an undirected graph \( G = (V, E) \), design an \( O(n + m) \)-time algorithm to decide if \( G \) contains a cycle or not; if it contains a cycle, output one (you only need to output one cycle).
(b) (6 points) A cycle in a directed graph $G = (V, E)$ is a sequence of $t \geq 2$ different vertices $v_1, v_2, \cdots, v_t$ such that $(v_i, v_{i+1}) \in E$ for every $i = 1, 2, \cdots, t - 1$ and $(v_t, v_1) \in E$. Given the linked-list representation of a directed graph $G = (V, E)$, design an $O(n + m)$-time algorithm to decide if $G$ contains a cycle or not; if it contains a cycle, output one (you only need to output one cycle).

**Remark** In a cycle of a directed graph, the directions of the edges have to be consistent. So, converting a directed graph to an undirected graph and then using algorithm for (a) does not give you a correct algorithm for (b).

**Problem 4 (10 points).** Give an $O(n + m)$-time algorithm to check if a given graph is bipartite or not, using depth-first-search as the graph traversal algorithm. You can use either the implementation of DFS using stack, or the implementation using recursion.