Description The next structure you will investigate is the Hash table and its methods implemented in pseudo code when linked lists are used to handle collisions. The task is to complete the pseudo code for the following hash table operations: Insert Remove Assume Hash table is a simple array of size 8, with indices 0..7. Numeric keys are mapped by a Hash function that gives the mod(8,n) value for any key “n” yielding the Hashtable index for that key (0..7). A Hashtable entry is null unless a key exists with that index as its hashed index; if so, the Hashtable entry points to the first node of a linked list of keys with that hash index. The last node on this linked list has a null reference for the next referenced node. Assume the occurrence of a linked list node is represented by the object “Node” and its “Data” and “NextRef” attributes. Deliverables: 1 pseudo code implementation of each Hash table operation: Insert and Remove Fully documented pseudo code. Add the completed pseudo code and the output to the Key Assignment template Section 2: Hashing, Heaps and Trees. Sort algorithms have many trade-offs, in-fact even the sorted output sequences might differ. Part 1: Your tasks for are the following: Identify at least five (5) algorithm differences that might be considered when choosing a sort algorithm. Offer examples of related sorts with the discussion of each difference considered. Part 2: Rationalize: You have formed a hypothesis that Big O analysis is not appropriate for comparing different sort algorithms, but rather for comparing the performance of a given sort algorithm as the number of sort keys change. (Hint: consider locality differences among sorts). Deliverables: 5 fully documented differences among sorting algorithms. Support the differences with Sort algorithms that exemplify the related characteristics. Generate a summary of Sort differences