ĐẠI HỌC TÔN ĐỨC THẮNG TON DUC THANG UNIVERSITY	FACULTY OF INFORMATION TECHNOLOGY FINAL EXAMINATION SEMESTER 1 ACADEMIC YEAR: 2022 - 2023

SUBJECT: DATA STRUCTURES AND ALGORITHMS

SUBJECT CODE: 504008

REVISION FOR THE FINAL EXAMINATION

I. SORTING

Given an array of integers arr = [89, 40, 46, 55, 54, 5, 50, 73, 23, 47]

1) Present steps to sort the array in descending order using Bubble Sort.
2) Present steps to sort the array in descending order using Selection Sort.
3) Present steps to sort the array in descending order using Insertion Sort.
4) Present steps to sort the array in descending order using Merge Sort.
5) Implement, in Java, the method below to sort an array ascendingly using Bubble Sort
```java
public void BubbleSort(int[] arr) {}
```
6) Implement, in Java, the method below to sort an array ascendingly using Selection Sort
```java
public void SelectionSort(int[] arr) {}
```
7) Implement, in Java, the method below to sort an array ascendingly using Insertion Sort
```java
public void InsertionSort(int[] arr) {}
```
8) (*optional*) Create a class MyComparator that helps to sort an array of integers so that even numbers are all before odd numbers, even numbers are sorted ascendingly, and odd numbers are sorted descendingly.
```java
class MyComparator implements Comparator<Integer> {}
```

II. BINARY SEARCH TREE & AVL TREE

a) Binary Search Tree

Given a list of keys [89, 40, 46, 55, 54, 5, 50, 73, 23, 47]

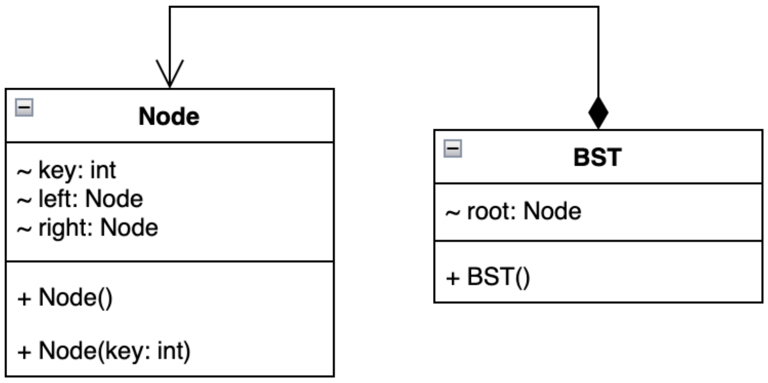
1) Present steps to build up a Binary Search Tree.
2) Delete node (47)
3) Delete node (46)
4) Delete node (89)
5) Delete node (40) using successors

b) AVL Tree

Given a list of keys [89, 40, 46, 55, 54, 5, 50, 73, 23, 47]

1) Present steps to build up a AVL Tree.
2) Delete node (47)
3) Delete node (89)
4) Delete node (46) using predecessors

c) Implementation



Given the class diagram above. Students implement, in Java, recursive functions below to perform the designated tasks.

1) Count the number of leaves
2) Compute the size of a subtree
3) Count the number of primes
4) Count the number of nodes that have one child only.

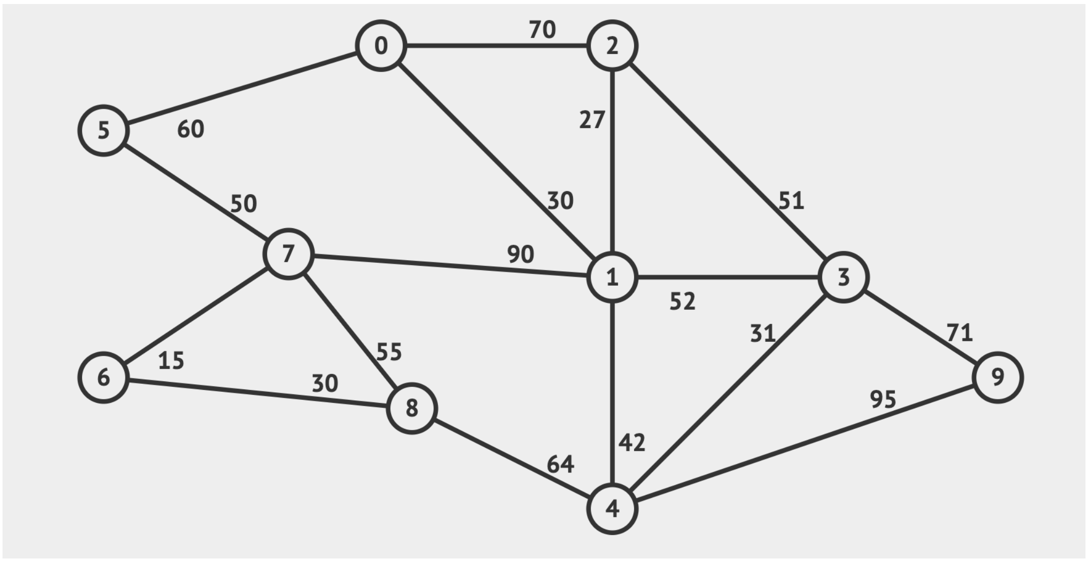
III. HEAP

Given a list of keys [89, 40, 46, 55, 54, 5, 50, 73, 23, 47]

1) Present steps to build up a Binary Min Heap
2) Present steps to build up a Binary Max Heap
3) (*optional*) Using `java.util.PriorityQueue<>` class to build up a heap of integers in which
    a. Even numbers have higher priority than odd ones
    b. Among even numbers, larger integers have higher priority
    c. Among odd numbers, smaller integers have higher priority

IV. GRAPH TRAVERSAL

Given the graph below

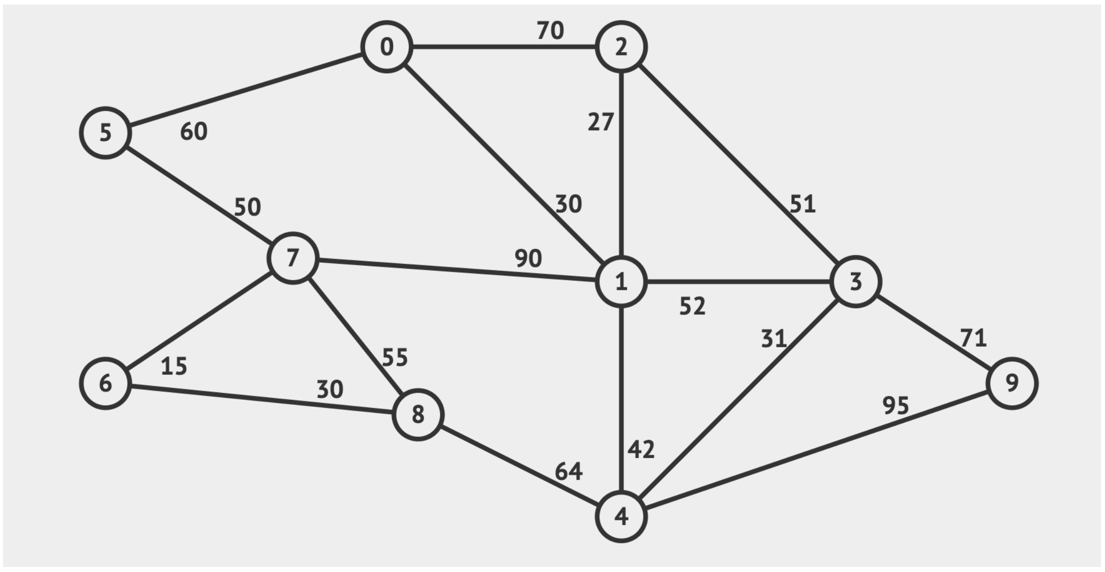


For each algorithm, including BFS and DFS,

- Perform the algorithm, starting from (0)
- Write down the list of keys in traversal order

If a vertex has several neighbors, then select the neighbor with the lower key to handle first.

V. MINIMUM SPANNING TREE



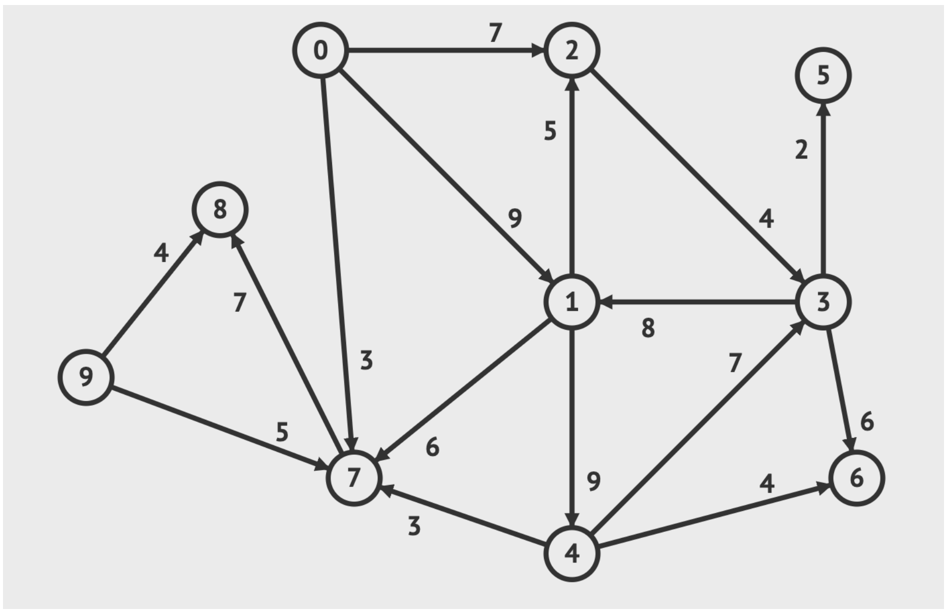
For each algorithm, including Prim's and Kruskal's

- Perform the algorithm
- Draw the final minimum spanning tree

- Write down the total cost of the minimum spanning tree

VI. SINGLE-SOURCE SHORTEST PATHS

Given the directed graph below



For each algorithm, including Bellman Ford's and Dijkstra,

- Perform the algorithm to find the shortest path from vertex 0 to the others
- Write down the path results and the corresponding cost.