# CSC 225 Assignment 4

Dryden Bryson

March 14, 2025

# Question 1:

## a) Selection Sort

| Steps | Result of steps | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| (0) Initial Array | 1 | 6 | 2 | 4 | 3 | 0 | 7 | 5 |
| (1) Swap: $0 \leftrightarrow 1$ | 0 | 6 | 2 | 4 | 3 | 1 | 7 | 5 |
| (2) Swap: $6 \leftrightarrow 1$ | 0 | 1 | 2 | 4 | 3 | 6 | 7 | 5 |
| (3) No change since 2 is already in order | 0 | 1 | 2 | 4 | 3 | 6 | 7 | 5 |
| (4) Swap: $4 \leftrightarrow 3$ | 0 | 1 | 2 | 3 | 4 | 6 | 7 | 5 |
| (5) No change since 4 is already in order | 0 | 1 | 2 | 3 | 4 | 6 | 7 | 5 |
| (6) Swap: $5 \leftrightarrow 6$ | 0 | 1 | 2 | 3 | 4 | 5 | 7 | 6 |
| (7) Swap: $7 \leftrightarrow 6$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Array is Sorted

(1)

## (b) Bubble Sort

| Steps | Result of steps | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| (0) Initial Array | 1 | 6 | 2 | 4 | 3 | 0 | 7 | 5 |
| (1) Swap $6 \leftrightarrow 2$, $6 \leftrightarrow 4$, $6 \leftrightarrow 3$, $6 \leftrightarrow 0$, $7 \leftrightarrow 5$ | 1 | 2 | 4 | 3 | 0 | 6 | 5 | 7 |
| (2) Swap $4 \leftrightarrow 3$, $4 \leftrightarrow 0$, $6 \leftrightarrow 5$ | 1 | 2 | 3 | 0 | 4 | 5 | 6 | 7 |
| (3) Swap $3 \leftrightarrow 0$ | 1 | 2 | 0 | 3 | 4 | 5 | 6 | 7 |
| (4) Swap $2 \leftrightarrow 0$ | 1 | 0 | 2 | 3 | 4 | 5 | 6 | 7 |
| (5) Swap $1 \leftrightarrow 0$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| (6) Check Array is Sorted | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Array is Sorted

(2)

## (c) Insertion Sort

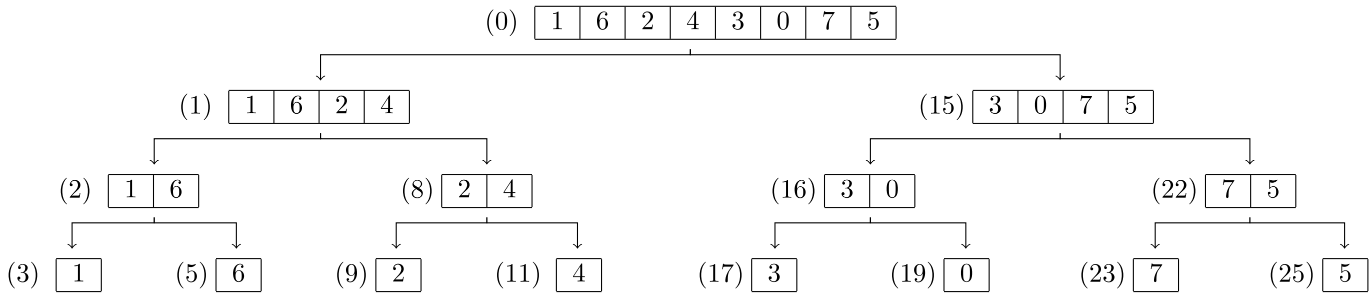
| Steps | Result of steps | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| (0) Initial Array | 1 | 6 | 2 | 4 | 3 | 0 | 7 | 5 |
| (1) No change since 6 is larger than one | 1 | 6 | 2 | 4 | 3 | 0 | 7 | 5 |
| (2) Insert 2 at index 1 | 1 | 2 | 6 | 4 | 3 | 0 | 7 | 5 |
| (3) Insert 4 at index 2 | 1 | 2 | 4 | 6 | 3 | 0 | 7 | 5 |
| (4) Insert 3 at index 2 | 1 | 2 | 3 | 4 | 6 | 0 | 7 | 5 |
| (5) Insert 0 at index 0 | 0 | 1 | 2 | 3 | 4 | 6 | 7 | 5 |
| (6) No change since 7 is larger than 6 | 0 | 1 | 2 | 3 | 4 | 6 | 7 | 5 |
| (6) Insert 5 at index 5 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Array is Sorted

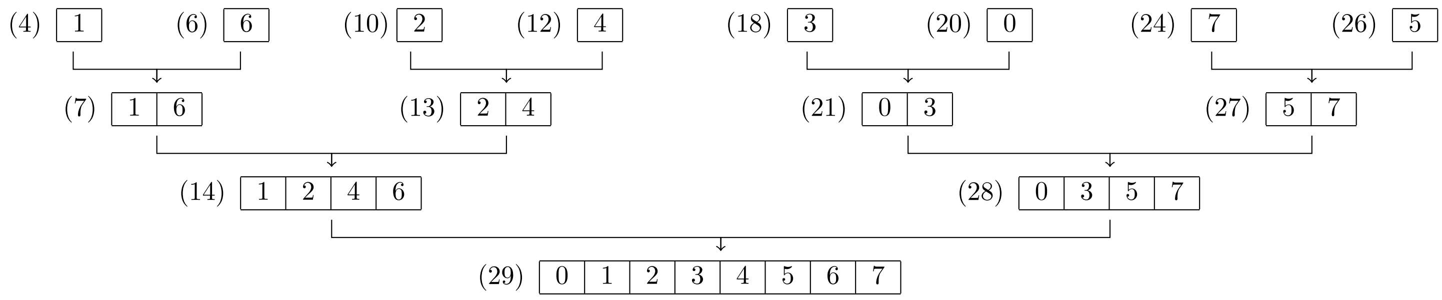
(3)

## d) Merge Sort

**Divide:**




**Merge:**



## Question 2:

The invariant property we aim to show is true throughout the execution of the loop is as follows: Let $k$ be the loop iteration counter starting at 1, then, the subarray of the array $A$ from indexes 0 to $k-1$ is sorted.

1. **Initialization**

   During initialization $k = 1$, and thus the given sorted subarray is the subarray from 0 to $1 - 1 = 0$, which is a single element. An array containing a single element is always sorted; thus, the invariant holds throughout initialization.

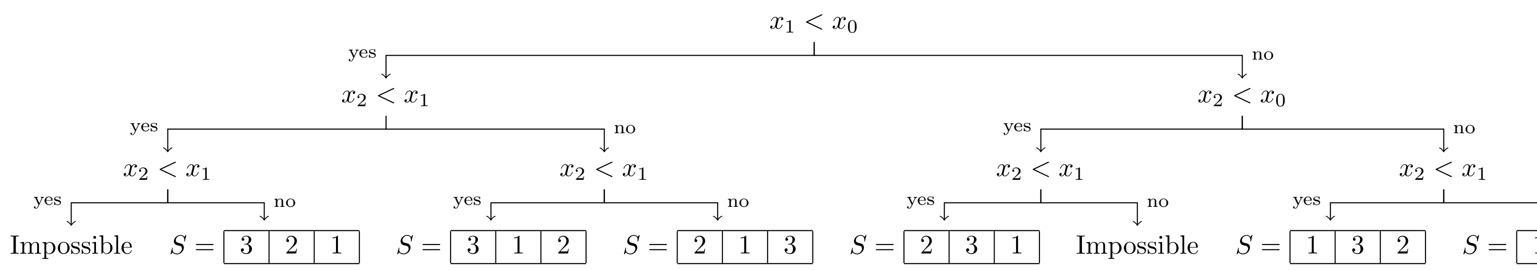
2. **Maintenance**

   We assume that the subarray $A[0..k-1]$ is sorted, and we assume that the inner loop is correct. We begin by storing the value at $A[k]$. Then, the inner loop shifts all elements of the subarray $A[0..k-1]$ that are greater than the stored value of $A[k]$ to the right once, with $A[k]$ getting overwritten on the first shift. This leaves a duplicate pair of values after the element that was found to be less than or equal to the stored value from $A[k]$. Then, $A[k]$ is inserted in the correct spot of the array after the element that was found to be less than or equal to $A[k]$. Thus, the subarray $A[0..k]$ is sorted, and the loop invariant holds during maintenance.

3. **Termination**

   Upon termination of the loop, $k = n-1$, where $n$ is the length of the array $A$. Thus, the subarray $A[0..n-1]$ is sorted, and the loop invariant holds.

## Question 3:



$x_1 < x_0$
yes
no
$x_2 < x_1$
$x_2 < x_0$
yes
no
yes
no
$x_2 < x_1$
$x_2 < x_1$
$x_2 < x_1$
$x_2 < x_1$
yes
no
yes
no
yes
no
yes
Impossible
$S =$ 3 2 1
$S =$ 3 1 2
$S =$ 2 1 3
$S =$ 2 3 1
Impossible
$S =$ 1 3 2
$S =$

## Question 4:

We count the following primitive operations:

- 1 primary operation calling $\boxed{T.isInternal(v)}$
- c primary operations calling $\boxed{processNode(v)}$
- T(n-1) primary operations calling $\boxed{preorder(T, T.leftChild(v))}$
- 1 primary operation checking if $\boxed{T.isInternal(v)}$ when calling $\boxed{preorder(T, T.rightChild(v))}$

Here I have not counted method calls as operations as the question seems to specifiy only counting the operations of T.isInternal(v) and proccessNode(v), thus the recurrence equation is:

$$T(n) = T(n-1) + c + 2, \quad T(0) = 1$$

To find the closed form of this equation we will first compute:

- $T(n-1) = T(n-2) + c + 2$
- $T(n-2) = T(n-3) + c + 2$
- $T(n-3) = T(n-4) + c + 2$

Then perform the following substitutions:

$$\begin{aligned} T(n) &= T(n-1) + (c+2) \\ &= T(n-2) + (c+2) + (c+2) = T(n-2) + 2(c+2) \\ &= T(n-3) + (c+2) + (c+2) + (c+2) = T(n-3) + 3(c+2) \end{aligned}$$

We then have the following pattern:

$$T(n) = T(n-i) + i(c+2)$$

Solving for $i$ we find that $i = n$ and thus our reccurence equation is as follows:

$$T(n) = 1 + n(c+2)$$

This worst case time complexity falls under the category of $O(n)$

## Question 5:

We will proceed with induction:

**Base Case:**
We want to show that $E(T) = I(T) + 2n$ when $n = 1$ first of all we have that for a tree with $n = 1$ meaning one internal node that,

- $E(t) = 2$
- $I(t) = 0$

Thus we have that $I(T) + 2n = (0) + 2(1) = 2$ which is equal to $E(t)$ and henceforth the base case is true for $n = 1$

**Inductive Hypothesis:**