Question 1

PART (i)  (2 points)
Given the array \( A[1, \ldots , 8] = \langle 6, 10, 13, 5, 8, 3, 2, 11 \rangle \), the pivot element \( A[1] = 6 \), and the Partition pseudo-code

\[
\text{PARTITION}(A, p, q):
\begin{align*}
1. & \quad \text{let } x \leftarrow A[p] \quad \text{(pivot)} \\
2. & \quad \text{let } i \leftarrow p \\
3. & \quad \text{for } j \leftarrow p + 1 \text{ to } q \\
4. & \quad \text{do if } A[j] \leq x, \text{ then } i \leftarrow i + 1 \\
5. & \quad \text{exchange } A[i] \leftrightarrow A[j] \\
6. & \quad \text{print } A \\
7. & \quad \text{exchange } A[i] \leftrightarrow A[j] \\
8. & \quad \text{print } A \\
9. & \quad \text{return } i
\end{align*}
\]

Show the arrays that are printed each time lines 7 and 9 are executed (for a total of 4 arrays).

PART (ii)  (3 points)
Assume all array elements are distinct. Given the Pseudo-Code of the quicksort program, provide the worst-case running time, \( T(n) \). First, set up the recurrence equation (2 points) and then solve it (1 point)

Hint: The worst-case happens when the array input is provided sorted or reverse sorted and the Partition program splits the array with one side having zero elements.

QuickSort pseudo-code

\[
\text{QUICKSORT}(A, p, r):
\begin{align*}
1. & \quad \text{if } p < r \\
2. & \quad \quad \text{then } q \leftarrow \text{PARTITION}(A, p, r) \\
3. & \quad \quad \text{QUICKSORT}(A, p, q - 1) \\
4. & \quad \quad \text{QUICKSORT}(A, q + 1, r)
\end{align*}
\]

Initial call: \( \text{QUICKSORT}(A, 1, n) \)

PART (iii)  (3 points) What is the best-case running time, \( T(n) \)? First, set up the recurrence equation (2 points) and then solve it (1 point)

Hint: The best-case happens when the Partition program splits the array evenly at every step.

PART (iv)  (2 points)
Suppose the input array is such that the Partition program alternates, at one step splitting the array evenly (lucky) and at the next step splitting the array with one side having zero elements (unlucky). What will be the running time of quicksort for such arrays of size \( n \)?
Solution to Question 1

PART (i)
1. \( A = \langle 6, 5, 10, 13, 8, 3, 2, 11 \rangle \)
2. \( A = \langle 6, 5, 3, 10, 13, 8, 2, 11 \rangle \)
3. \( A = \langle 6, 5, 3, 2, 10, 13, 8, 11 \rangle \)
4. \( A = \langle 5, 3, 2, 6, 10, 13, 8, 11 \rangle \)

PART (ii)

\[
T(n) = T(0) + T(n - 1) + \Theta(n) \\
= \Theta(1) + T(n - 1) + \Theta(n) \\
= T(n - 1) + \Theta(n) \\
= \Theta(n^2)
\]

since

\[
\sum_{k=1}^{n} \Theta(k) = \Theta(n^2).
\]

PART (iii)

\[
T(n) = 2 T\left(\frac{n}{2}\right) + \Theta(n) \\
= \Theta(n \log n)
\]

since it gives a recursion Tree with height \( \log n \) and the sum of the costs of the nodes at each depth is \( \Theta(n) \).

PART (iv)
We alternate lucky, unlucky, lucky, unlucky, lucky, ...

\[
L(n) = 2U\left(\frac{n}{2}\right) + \Theta(n) \quad \text{lucky}
\]
\[
U(n) = L(n - 1) + \Theta(n) \quad \text{unlucky}
\]

Solving it

\[
L(n) = 2L\left(\frac{n}{2}\right) + \Theta\left(\frac{n}{2}\right) + \Theta(n) \\
= 2L\left(\frac{n}{2}\right) - 1 + \Theta(n) \\
= \Theta(n \log n) \quad \text{(1)}
\]
Questions 2

Let $G$ be a directed graph with edges that have lengths. Present, in high-level pseudo-code, an efficient algorithm that computes, for all pairs of vertices $i, j$, the length of the shortest path from $i$ to $j$, and the number of paths from $i$ to $j$ that have this shortest length. For this problem, the length of a path is the sum of its edge lengths. Assume that all cycles have positive total length. The edge lengths are stored in the array $E[1..n, 1..n]$, so that if $(i, j)$ is an edge in $G$, then $E[i, j]$ is the length of the edge. You can also assume that $E(i, j)$ is infinite $(i, j)$ is not in $G$.

Two paths are different if the sequence of edges that define the path are different. For example, in the figure below, there are seven different shortest paths from $i$ to $j$.

![Diagram of a graph with labeled edges and vertices](image)

Solution to Question 2

```pseudo
def Count(C[1..n, 1..n], Pcount[1..n, 1..n]):
    for all pairs $i, j$ do
        if $C[i, j] \neq \infty$ then
            $Pcount[i, j] \leftarrow 1$
        else
            $Pcount[i, j] \leftarrow 0$
        endif
    endfor
    for $k \leftarrow 1$ to $n$ do
        for $i \leftarrow 1$ to $n$ do
            for $j \leftarrow 1$ to $n$ do
                if $C[i, j] > C[i, k] + C[k, j]$ then
                    $C[i, j] \leftarrow C[i, k] + C[k, j]$;
                    $Pcount[i, j] \leftarrow Pcount[i, k] \times Pcount[k, j]$
                elseif $C[i, j] = C[i, k] + C[k, j]$ then
                    $Pcount[i, j] \leftarrow Pcount[i, j] + Pcount[i, k] \times Pcount[k, j]$
                endif
            endfor
        endfor
    endfor
end Count.
```

3
Question 3

1) The Towers of Hanoi problem is the following.

You are given three posts, $A$, $B$, and $C$. There are $n$ rings $r_n$, $r_{n-1}$, ..., $r_2$, $r_1$ sitting on post $A$, with $r_i$ directly on top of $r_{i+1}$, for $1 \leq i < n$, so that ring $r_1$ is the top ring. The other posts are empty.

The problem is to move the rings so that they all wind up on post $B$ subject to the following rules:

A ring $r_i$ cannot be placed on a pole that holds a (smaller indexed) ring $r_h$ where $h < i$.

Only the top ring on a post can be removed at any step.

Only one ring can be moved at each step, and it must be removed from one post and placed on another.

Recursion gives an easy solution to the problem.

procedure $TH(n,A,B,C)$:
  if $n = 1$ then move top ring on $A$ to $B$
  else
    $TH(n-1,A,C,B)$: \{ Move the top $N - 1$ rings from $A$ to $C$ \}
    move the top ring on $A$ to $B$;
    $TH(n-1,C,B,A)$: \{ Move the top $N - 1$ rings from $C$ to $B$ \}
  endif
end-$TH$.

a) Present the recurrence equation for the exact number of ring moves for $TH(n,A,B,C)$. Be sure to include the initial condition as well as the recurrence equation.

b) Now suppose that in addition to posts $A$, $B$ and $C$, there is a short post $D$, which can hold a single ring of any type. All other aspects of the problem are unchanged. Present, in high level code, as efficient a solution to this new problem as you can.

c) Present the recurrence equation for the exact number of ring moves for the solution given in part b. Be sure to include the initial condition as well as the recurrence equation.

Solution to Question 3

a) $T(1) = 1$;

$T(n) = 2T(n - 1) + 1$.

b) procedure $THD(n,A,B,C)$:
  if $n = 1$ then move top ring on $A$ to $B$
  else $n = 2$ then
    move top ring on $A$ to $D$;
    move top ring on $A$ to $B$;
    move top ring on $D$ to $B$
  else
    $TH(n-2,A,C,B)$;
    $THD(2,A,B,C,D)$;
    $TH(n-2,C,B,A)$
  endif
end-$THD$.

c) $T(1) = 1$;

$T(2) = 3$;

$T(n) = 2T(n - 2) + 3$, $n > 2$. 