FINAL PROJECT DOCUMENTATION

GROUP 1

Dibuat Oleh:

Aditya Sutanto - 422024013

Alvin Bungur - 422024017

Jonathan Christiandinata - 422024018

Fakultas Teknik dan Ilmu Komputer

Sistem Informasi Angkatan 2024

Universitas Kristen Krida Wacana

2024

Description

Arrays Algorithm:

1. Kadane's Algorithm: Algoritma ini berfungsi untuk mencari jumlah maksimum dari sebuah sub array.

2. Floyd's Algorithm (detection Algorithm): Algoritma ini berfungsi untuk mendeteksi apakah ada sebuah siklus yang terjadi di dalam suatu linked list.

3. Knuth-Morris-Pratt (KMP) Algorithm: Algoritma ini befungsi sebagai algoritma pencocokan string yang dirancang untuk mencari sebuah pola (substring) dalam sebuah teks (string).

4. Quick Select Algorithm: Algoritma ini berfungsi untuk menemukan elemen ke-k terkecil (misal elemen ke-1 terkecil atau elemen ke-2 terkecil) dalam sebuah array yang tidak tersusun (unsorted array).

5. Boye-Moore Algorithm: salah satu algoritma yang berfungsi untuk pencarian pola (substring) secara efisien dalam sebuah string.

Basic Algorithm:

1. Euclidean Algorithm: Algoritma yang berfungsi untuk menghitung Greatest common Divisor (GCD) dari dua bilangan.

2. Huffman Algorithm: Algoritma yang berfungsi untuk mengompresi data berdasarkan dari frekuensi munculnya karakter dalam teks input dengan menggunakan Huffman tree.

3. Union-Find Algorithm: Algoritma yang digunakan untuk melacak dan mengelola partisi dari himpunan yang saling terpisah (disjoint sets).

Sorting Algorithms:

1. Insertion Sort: Algoritma ini bekerja dengan membangun array yang sudah terurut secara bertahap. Pada setiap iterasi, elemen dari array yang belum terurut dipilih dan dimasukkan ke dalam posisi yang benar di array yang sudah terurut.

2. Selection Sort: Algoritma ini bekerja dengan membagi array menjadi dua bagian: terurut dan belum terurut. Pada setiap langkah, elemen terkecil dari bagian yang belum terurut dipilih dan dipindahkan ke bagian terurut.

3. Heap Sort: Algoritma ini menggunakan struktur data heap untuk membuat array terurut. Heap adalah struktur data berbentuk pohon biner yang memenuhi sifat heap.

4. Merge Sort: Algoritma ini adalah algoritma rekursif yang membagi array menjadi dua bagian, mengurutkan masing-masing bagian, dan kemudian menggabungkannya kembali.

5. Quick Sort: Algoritma ini adalah algoritma divide and conquer yang memilih elemen sebagai pivot dan mempartisi array ke dalam dua sub-array: elemen yang lebih kecil dari pivot dan elemen yang lebih besar.

6. Counting Sort: Algoritma ini bekerja dengan menghitung jumlah elemen dengan nilai tertentu, kemudian menggunakan informasi tersebut untuk menentukan posisi elemen dalam array terurut.

Searching Algorithms:

1. Binary Search: Algoritma ini digunakan untuk mencari elemen di array yang sudah terurut dengan cara membagi array menjadi dua bagian pada setiap iterasi.

2. Linear Search: Algoritma ini mencari elemen dengan memeriksa setiap elemen satu per satu dari awal hingga akhir.

3. Depth First Search (DFS): Algoritma ini digunakan untuk menjelajahi graf atau pohon dengan menjelajahi sebanyak mungkin cabang sebelum kembali.

4. Breadth First Search (BFS): Algoritma ini digunakan untuk menjelajahi graf atau pohon dengan menjelajahi semua node pada tingkat yang sama sebelum turun ke tingkat berikutnya.

Graphs Algorithms:

1. Kruskal's Algorithm: Algoritma ini digunakan untuk menemukan Minimum Spanning Tree (MST) dengan memilih edge terkecil yang tidak membentuk siklus.

2. Dijkstra's Algorithm: Algoritma ini digunakan untuk mencari jarak terpendek dari satu sumber ke semua node dalam graf dengan bobot non-negatif.

3. Bellman-Ford Algorithm: Algoritma ini digunakan untuk mencari jarak terpendek dalam graf, termasuk graf dengan bobot negatif.

4. Floyd-Warshall Algorithm: Algoritma ini digunakan untuk menemukan jarak terpendek antara semua pasangan node dalam graf.

5. Topological Sort: Algoritma ini digunakan untuk mengurutkan node dalam graf berarah asiklik (DAG).

6. Flood Fill Algorithm: Algoritma ini digunakan untuk mengganti warna pada area terhubung dalam grid.

7. Lee Algorithm: Algoritma ini digunakan untuk menemukan jarak terpendek dalam grid menggunakan pendekatan BFS.

Repository

Link repository GitHub: https://github.com/zen552/Group_1_2024_UAS_SIWP1001.git

Time Complexity and Space Complexity (Arrays Algorithm)

Algorithms	Time Complexity			Space Complexity
	Best	Average	Worst	
Kadane's Algorithm	O(n)	O(n)	O(n)	O(1)
Floyd's Algorithm (Detection Algorithm)	O(n)	O(n)	O(n)	O(1)
Knuth-Morris-Pratt (KMP) Algorithm	O(n + m)	O(n + m)	O(n + m)	O(m)
Quick Select Algorithm	O(n)	O(n)	$O(n^2)$	O(logn)
Boye-Moore Algorithm	O(n/m)	O(n)	O(mn)	O(1)

Time Complexity and Space Complexity (Basic Algorithm)

Algorithms	Time Complexity			Space Complexity
	Best	Average	Worst	
Euclidean Algorithm	O(log min(a, b))	O(log min(a, b))	O(log min(a ,b))	O(1)
Huffman Algorithm	O(n log n)	O(n log n)	O(n log n)	O(n)
Union-Find Algorithm	O(a(n))	O(a(n))	O(a(n))	O(n)

Time Complexity and Space Complexity (Sorting Algorithm)

Algorithms	Time Complexity			Space Complexity
	Best	Average	Worst	
Insertion Sort	O(n)	$O(n^2)$	$O(n^2)$	O(1)
Selection Sort	$O(n^2)$	$O(n^2)$	$O(n^2)$	O(1)
Heap Sort	O(n log n)	O(n log n)	O(n log n)	O(1)
Merge Sort	O(n log n)	O(n log n)	O(n log n)	O(n)
Quick Sort	O(n log n)	O(n log n)	$O(n^2)$	O(log n)
Counting Sort	O(n + k)	O(n + k)	O(n + k)	O(n + k)

Time Complexity and Space Complexity (Searching Algorithm)

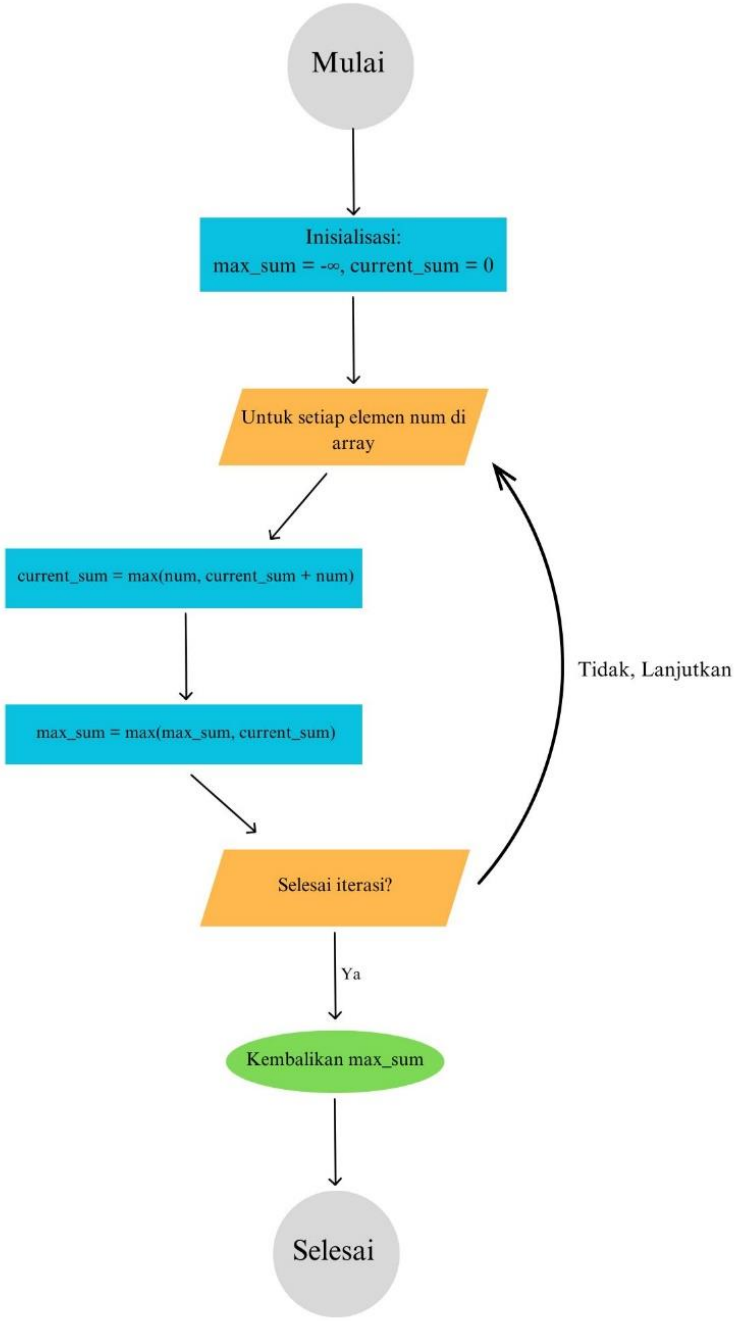
Algorithms	Time Complexity			Space Complexity
	Best	Average	Worst	
Binary Search	O(1)	O(log n)	O(log n)	O(1)
Linear Search	O(1)	O(n)	O(n)	O(1)
Depth First Search	O(V + E)	O(V + E)	O(V + E)	O(V) (stack)
Breadth First Search	O(V + E)	O(V + E)	O(V + E)	O(V) (queue)

Time Complexity and Space Complexity (Graphs Algorithm)

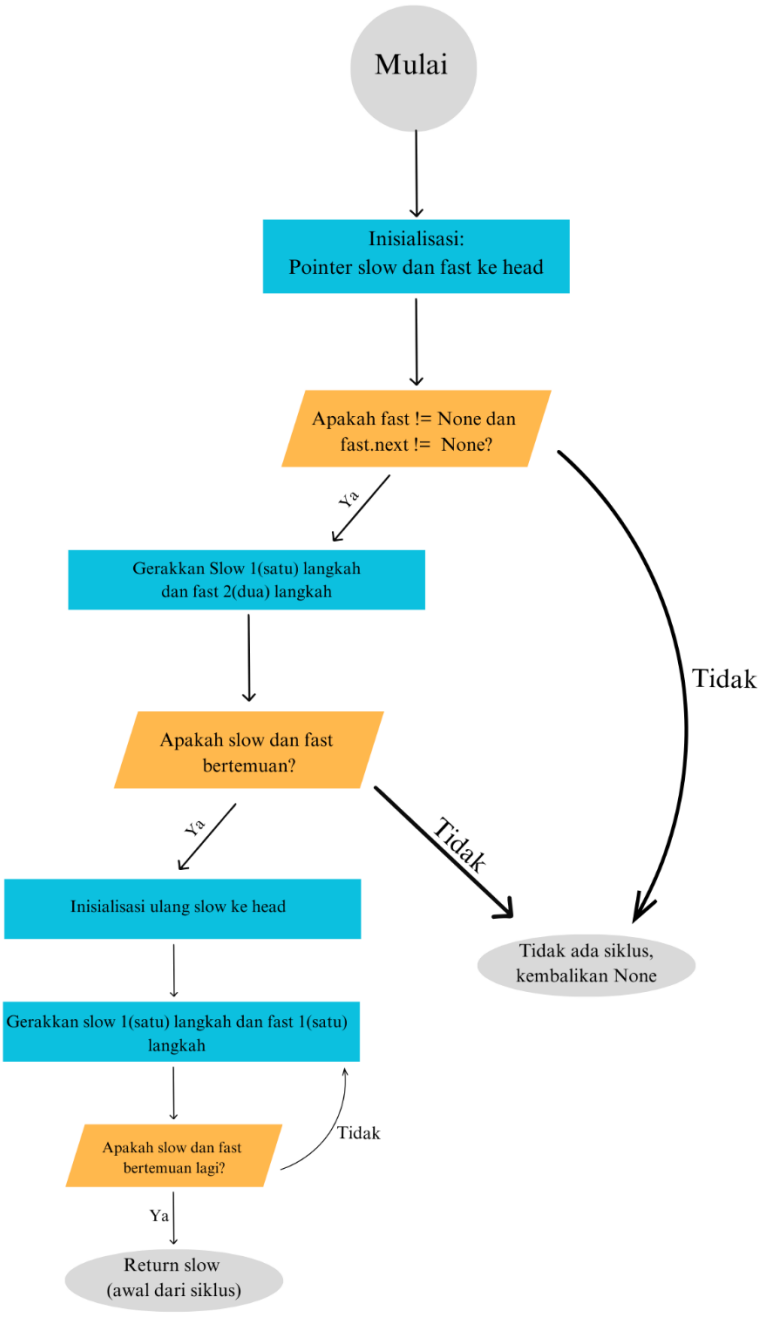
Algorithms	Time Complexity	Space Complexity
Kruskal's Algorithm	O(E log E)	O(V + E)
Dijkstra's Algorithm	O((V + E) log V)	O(V + E)
Bellman-Ford Algorithm	O(V * E)	O(V)
Floyd-Warshall Algorithm	$O(V^3)$	$O(V^2)$
Topological Sort	O(V + E)	O(V + E)
Flood Fill Algorithm	O(N * M)	O(N * M)
Lee Algorithm	O(N * M)	O(N * M)

Flowchart (arrays Algorithm)

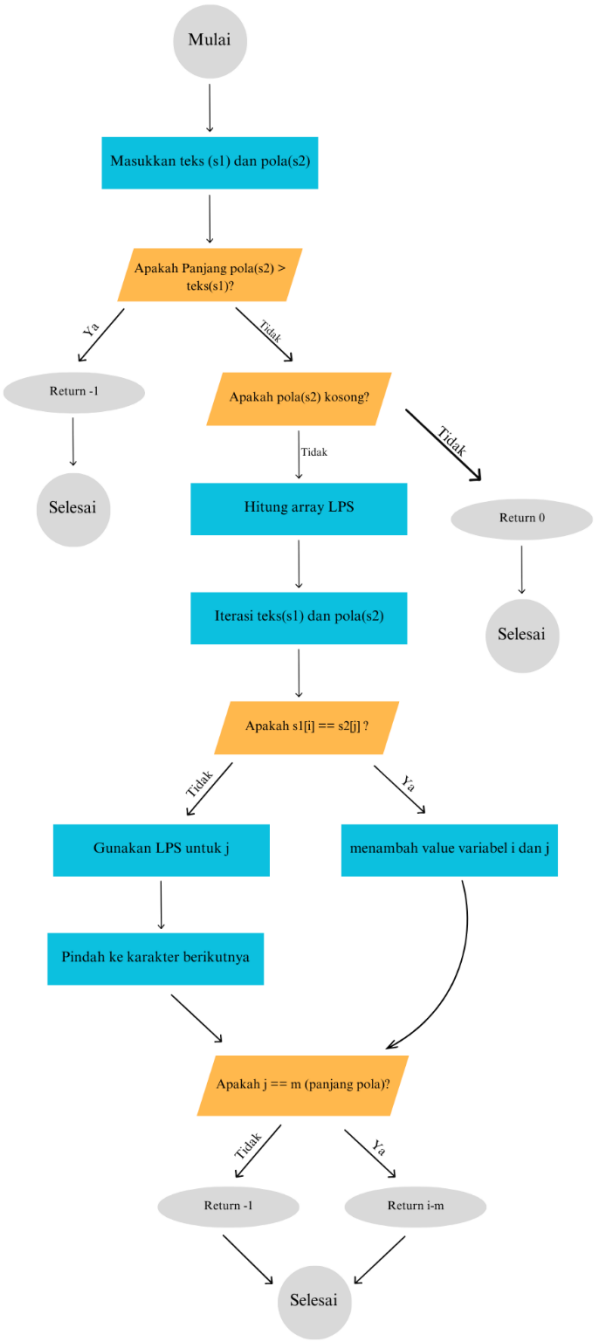
Kadane’s Algorithm:



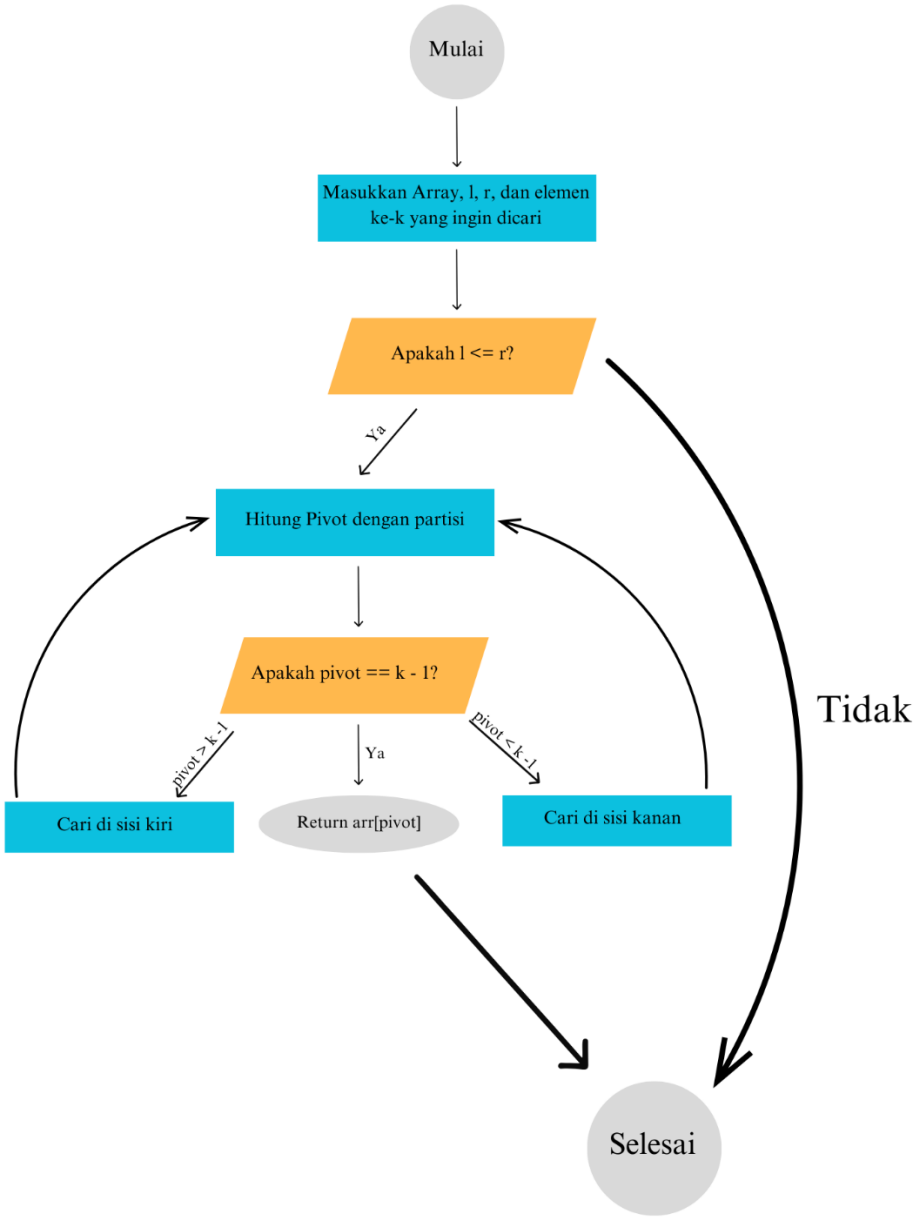
Floyd’s Algorithm (Detection Algorithm):



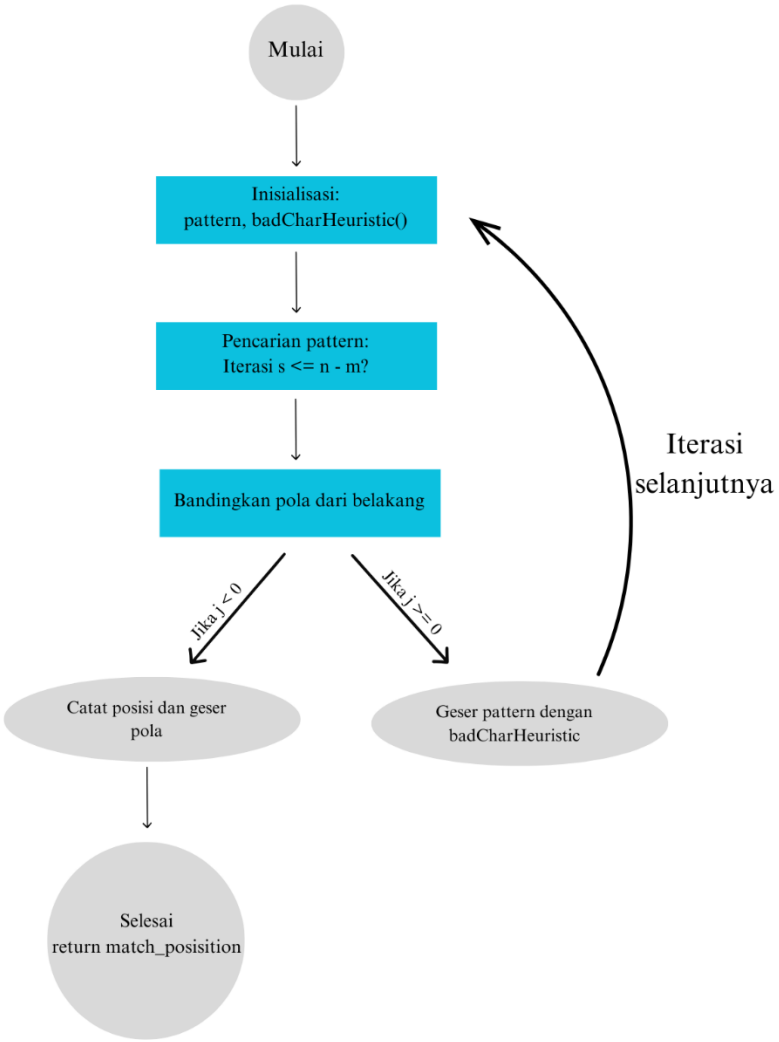
Knuth-Morris-Pratt (KMP) Algorithm:



Quick Select Algorithm:

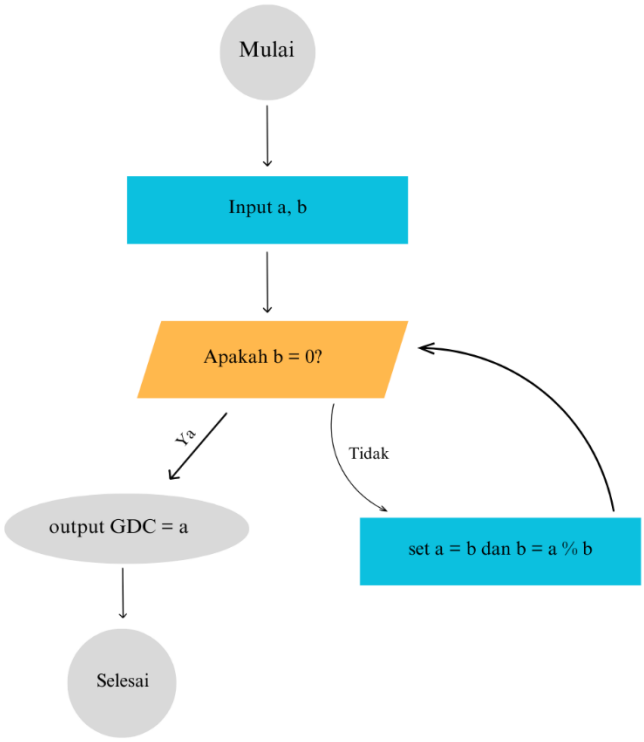


Boye-Moore Algorithm:

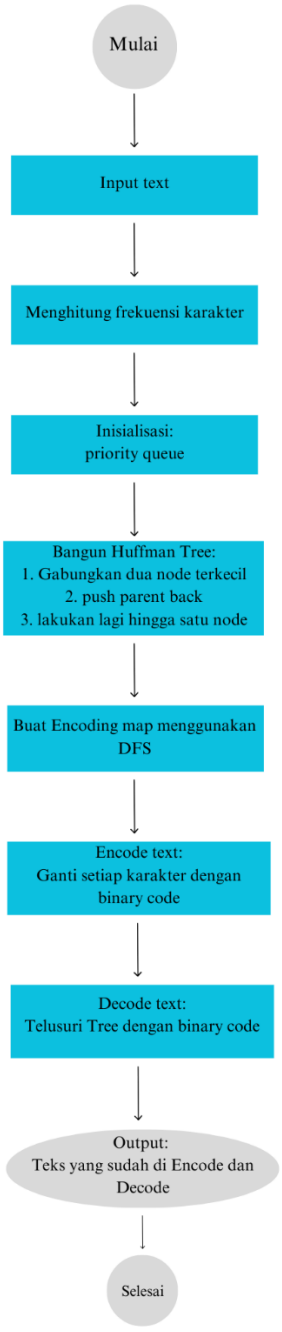


Flowchart (Basic Algorithm)

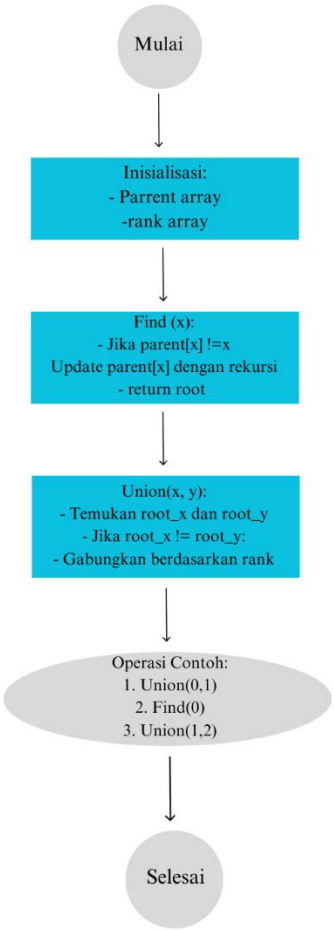
Euclidean Algorithm:



Huffman Algorithm:

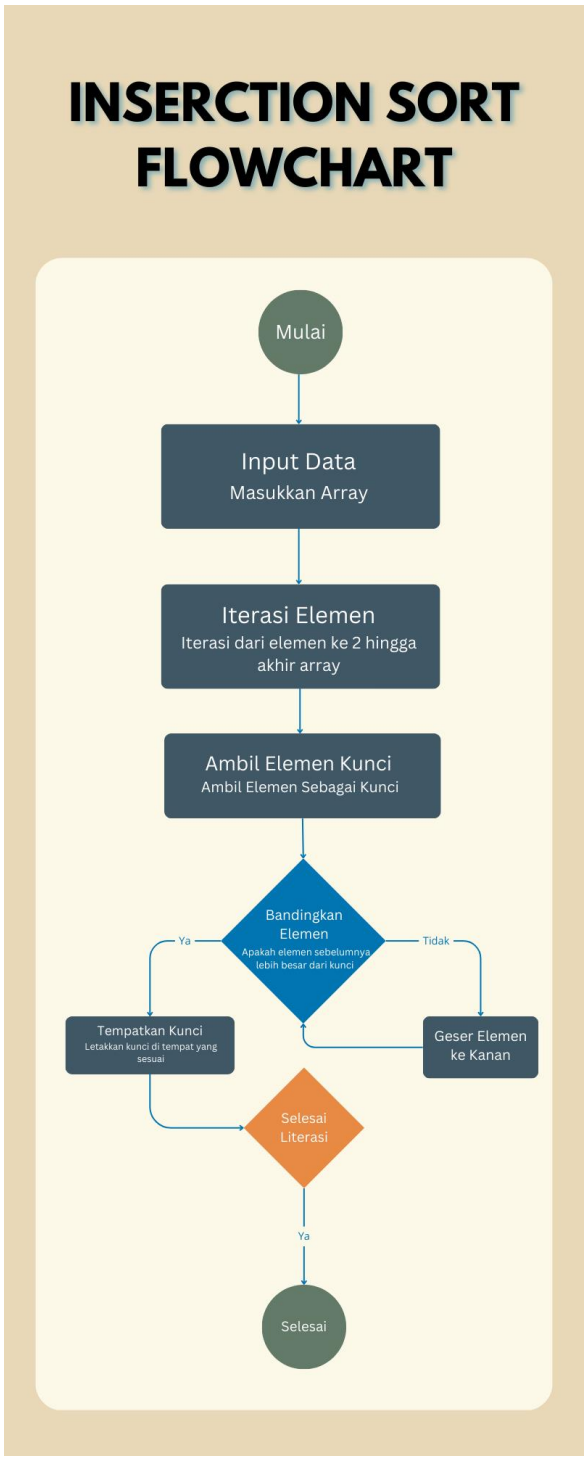


Union-Find Algorithm:

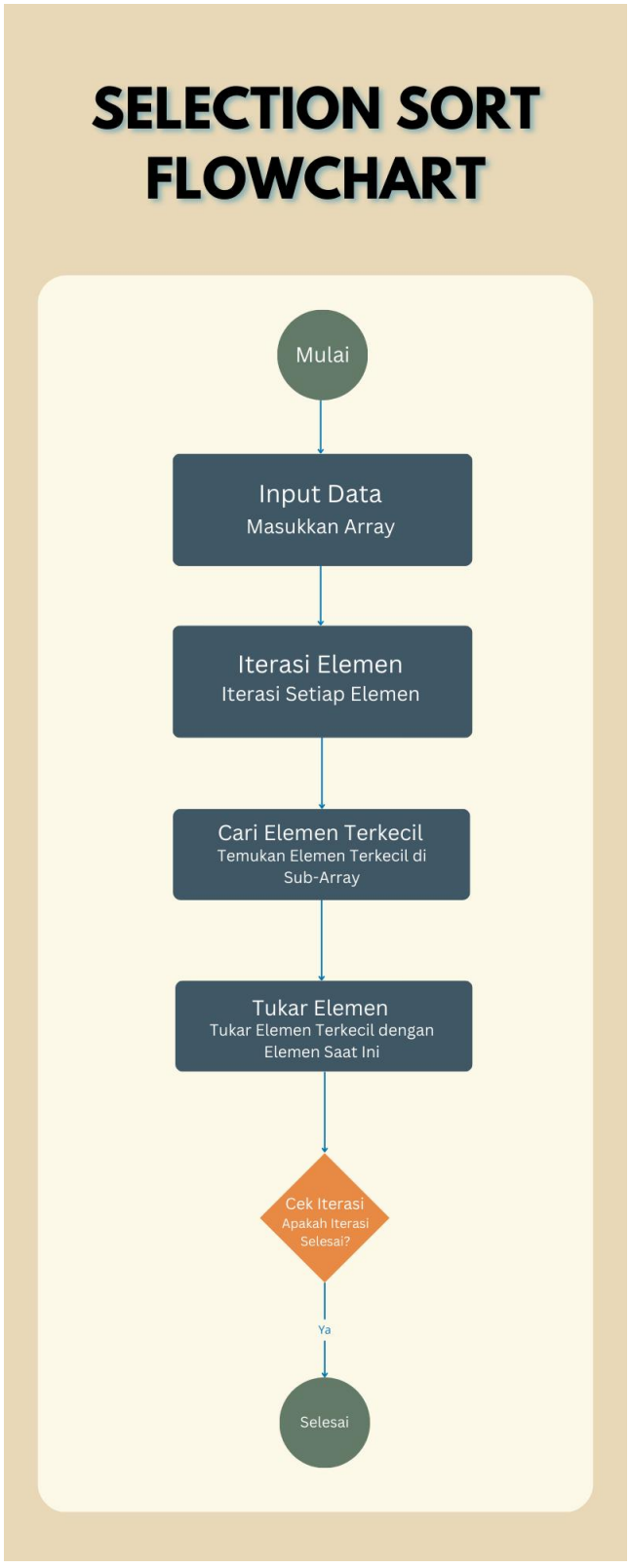


Flowchart (Sorting Algorithm)

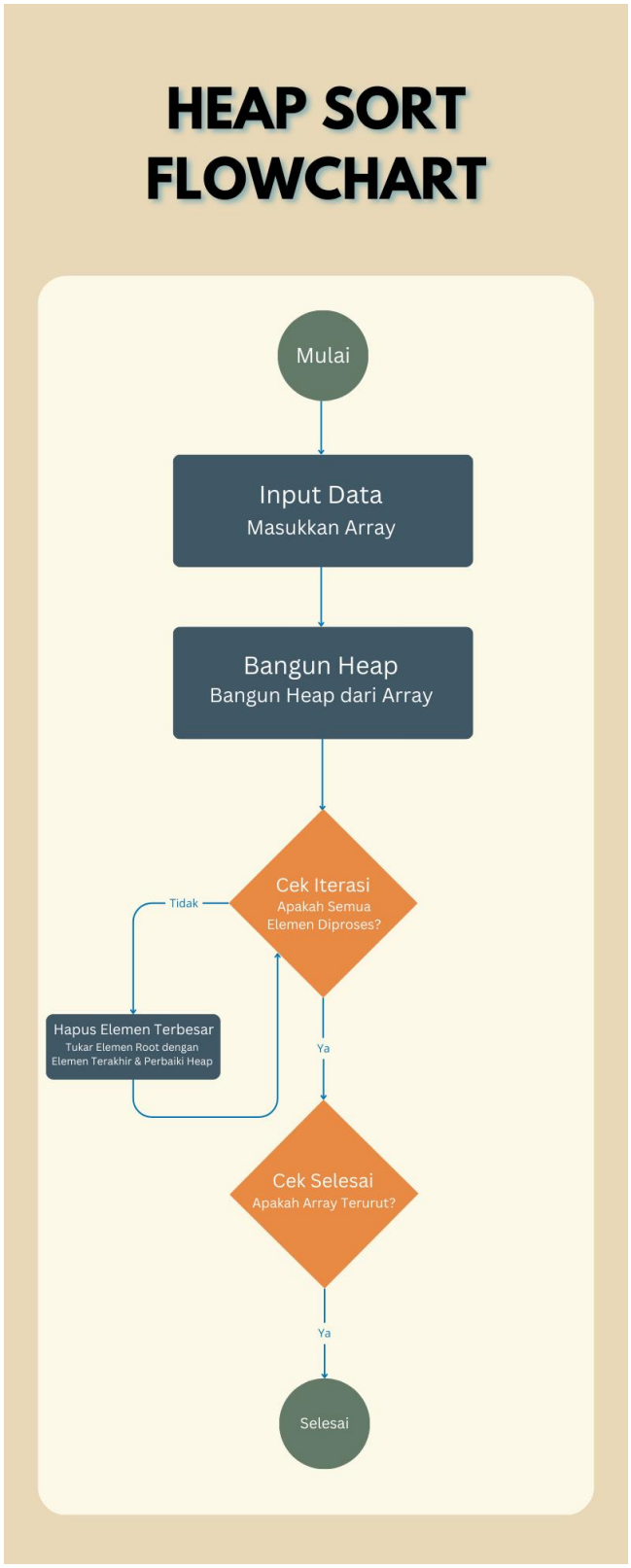
Insertion Sort:



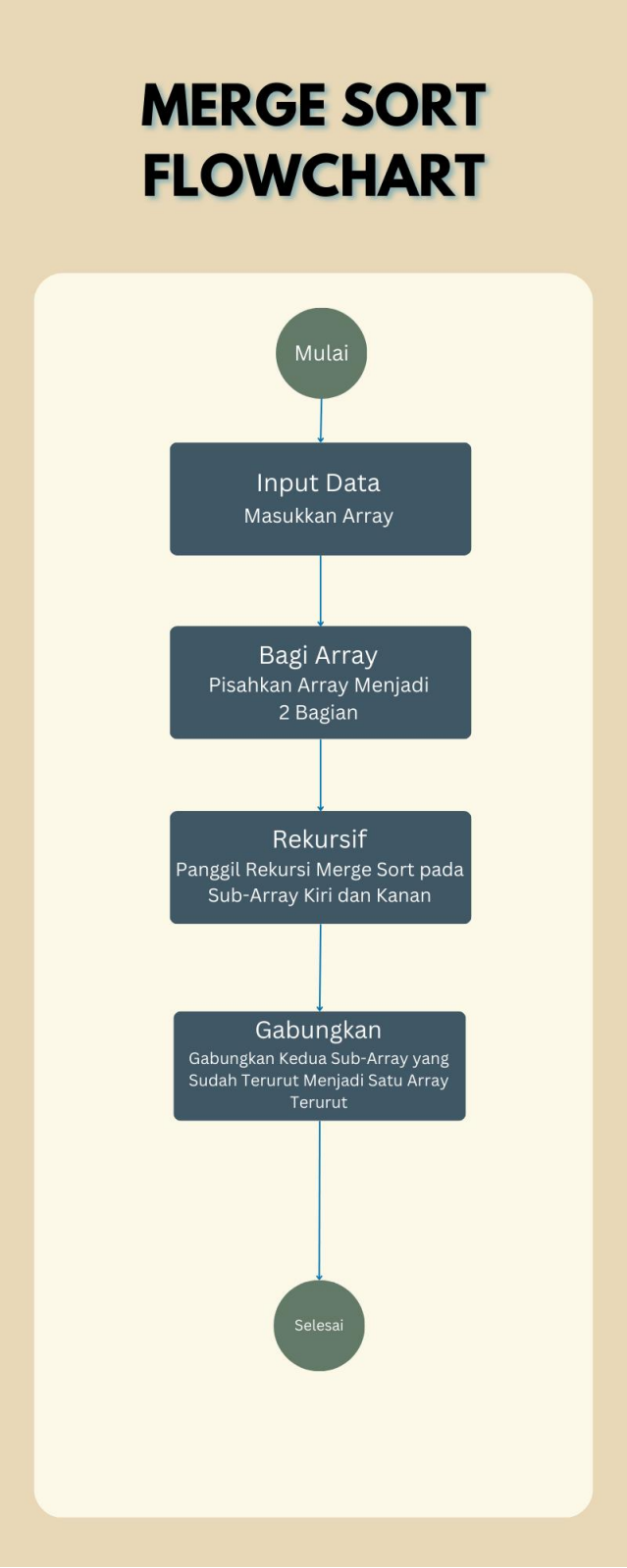
Selection Sort:



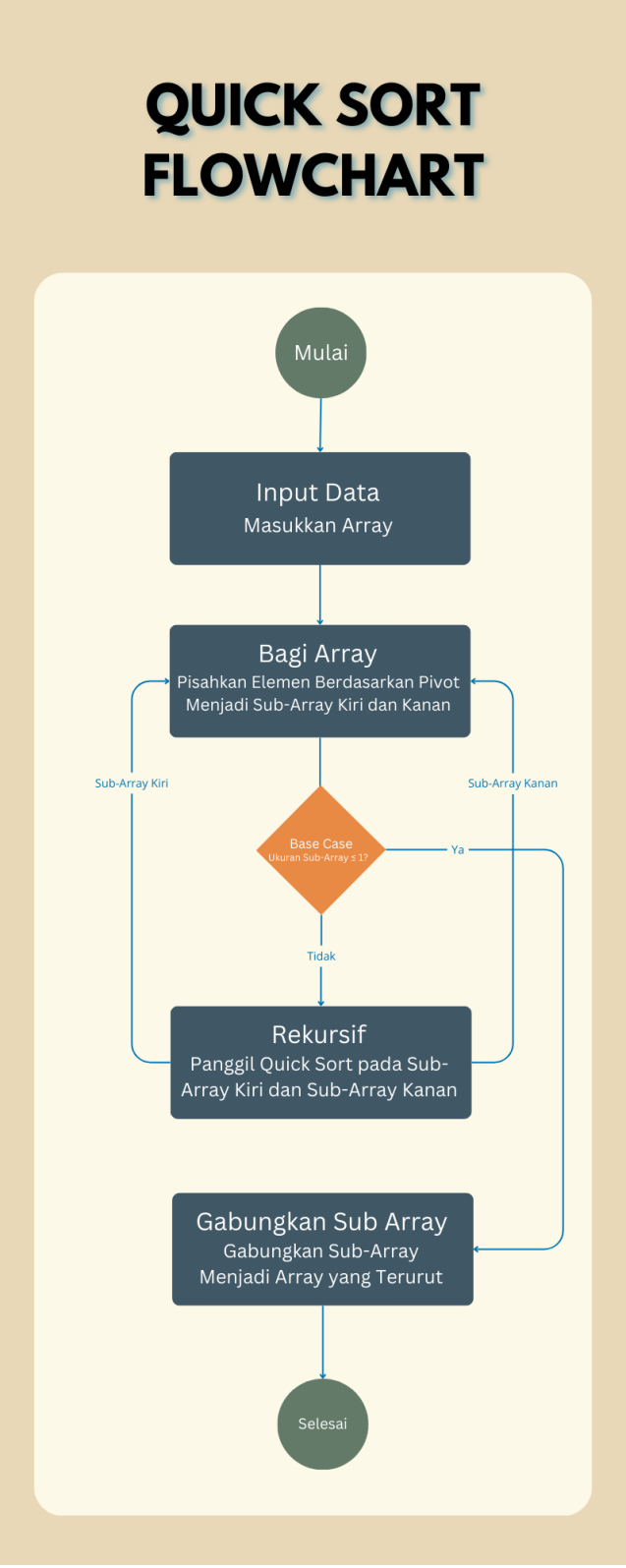
Heap Sort:


HEAP SORT
FLOWCHART
Mulai
Input Data
Masukkan Array
Bangun Heap
Bangun Heap dari Array
Cek Iterasi
Apakah Semua
Elemen Diproses?
Tidak
Hapus Elemen Terbesar
Tukar Elemen Root dengan
Elemen Terakhir & Perbaiki Heap
Ya
Cek Selesai
Apakah Array Terurut?
Ya
Selesai

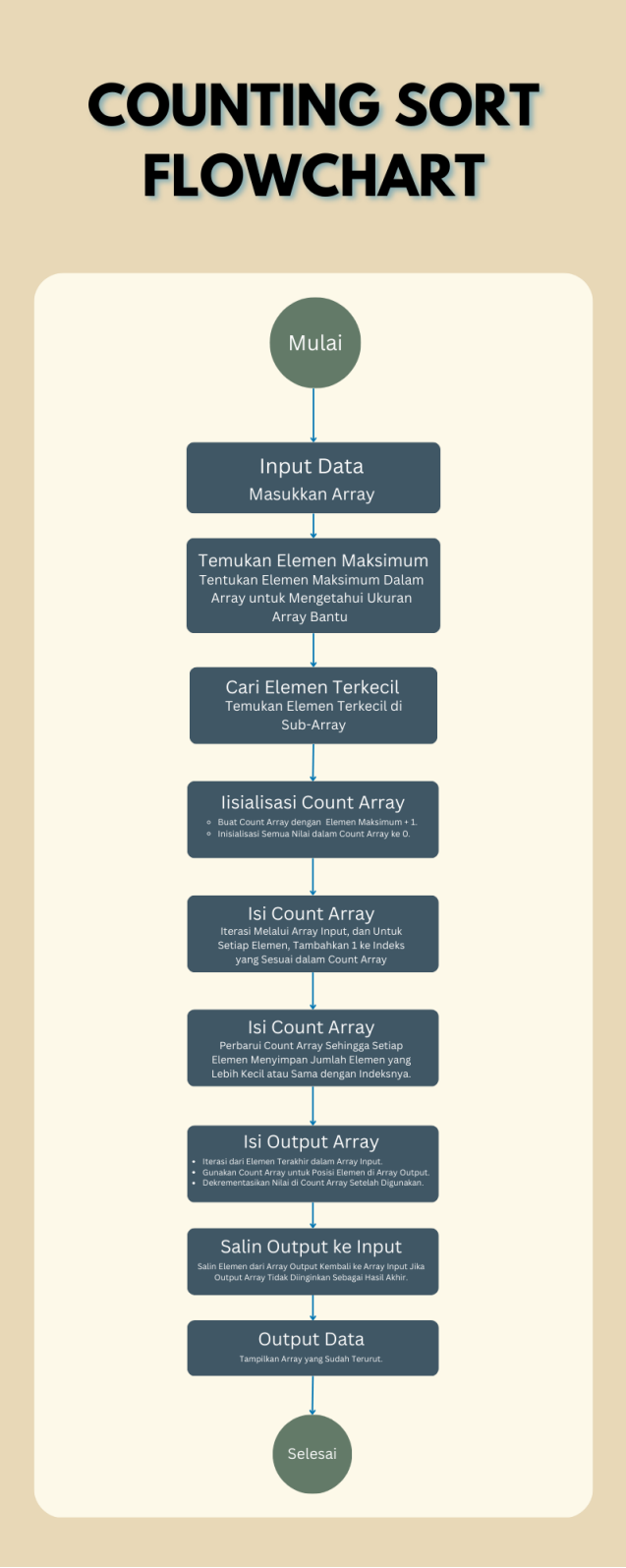
Merge Sort:


MERGE SORT
FLOWCHART
Mulai
Input Data
Masukkan Array
Bagi Array
Pisahkan Array Menjadi
2 Bagian
Rekursif
Panggil Rekursi Merge Sort pada
Sub-Array Kiri dan Kanan
Gabungkan
Gabungkan Kedua Sub-Array yang
Sudah Terurut Menjadi Satu Array
Terurut
Selesai

Quick Sort:

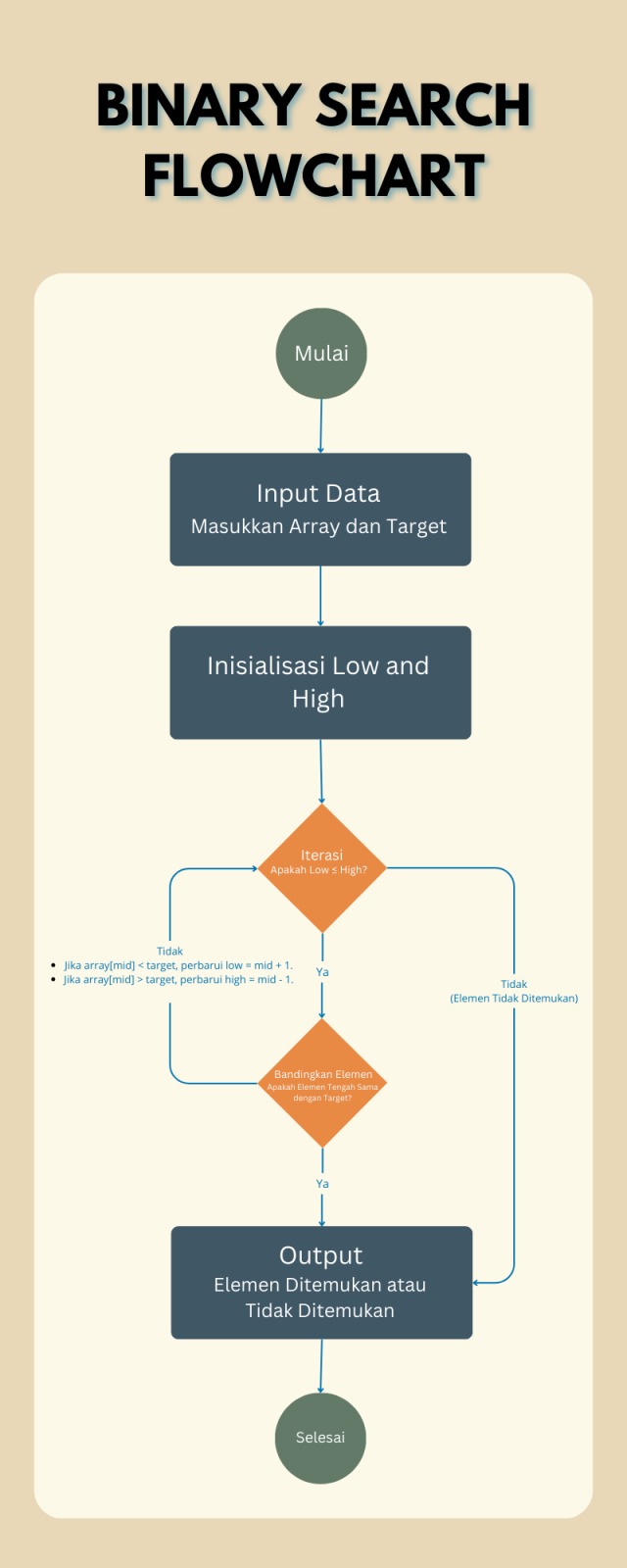

QUICK SORT
FLOWCHART
Mulai
Input Data
Masukkan Array
Bagi Array
Pisahkan Elemen Berdasarkan Pivot
Menjadi Sub-Array Kiri dan Kanan
Sub-Array Kiri
Sub-Array Kanan
Base Case
Ukuran Sub-Array ≤ 1?
Ya
Tidak
Rekursif
Panggil Quick Sort pada Sub-
Array Kiri dan Sub-Array Kanan
Gabungkan Sub Array
Gabungkan Sub-Array
Menjadi Array yang Terurut
Selesai

Counting Sort

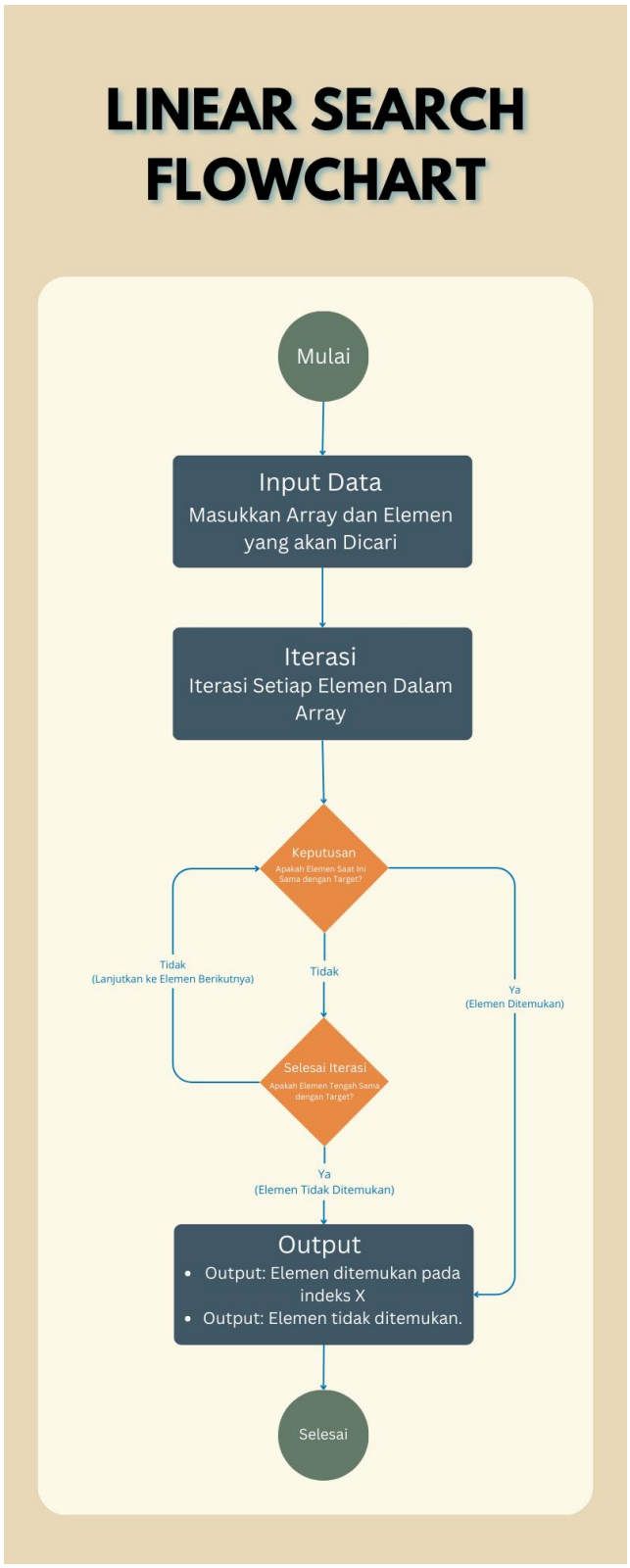


Flowchart (Searching Algorithm)

Binary Search:

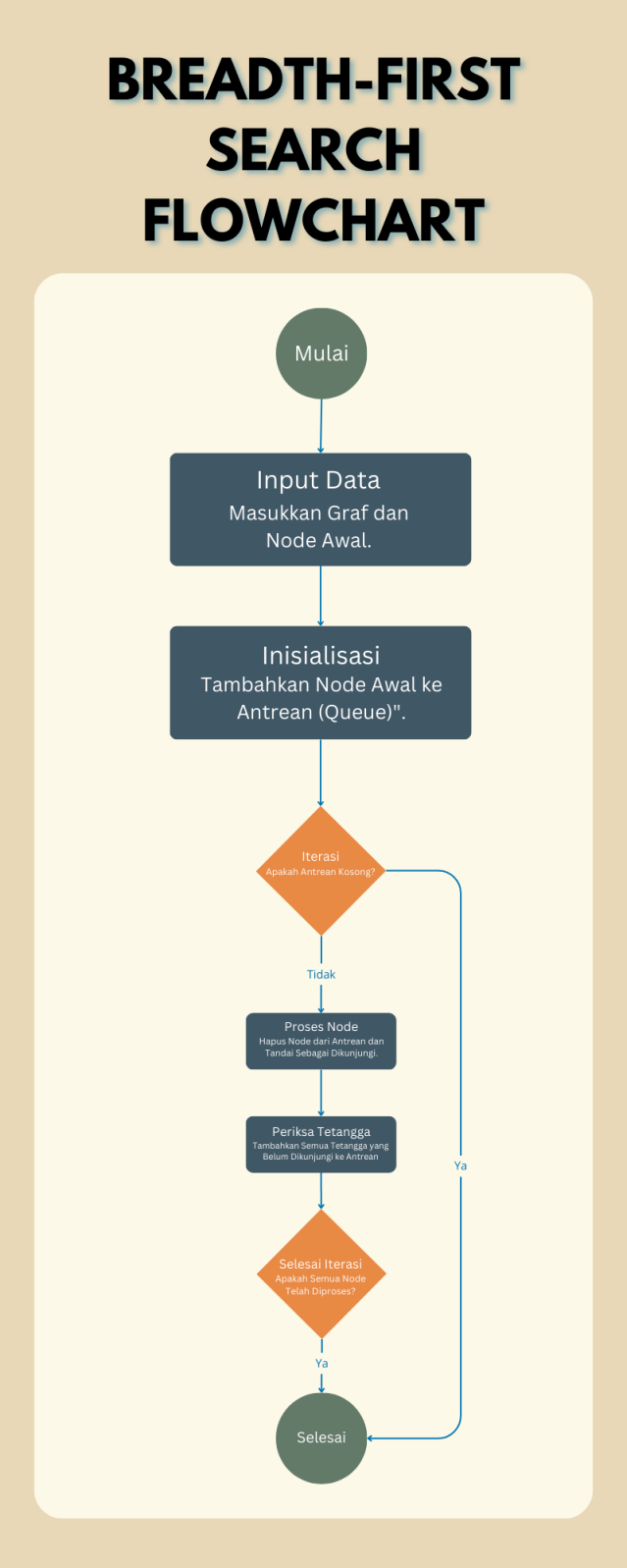


Linear Search:



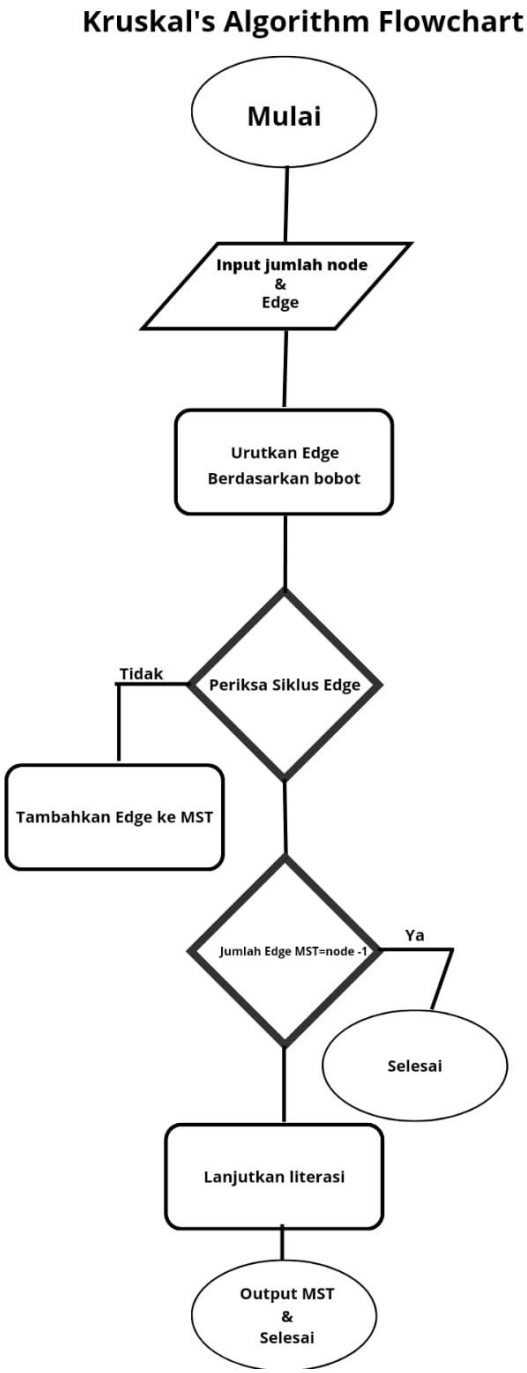
Depth First Search:

Breadth First Search:

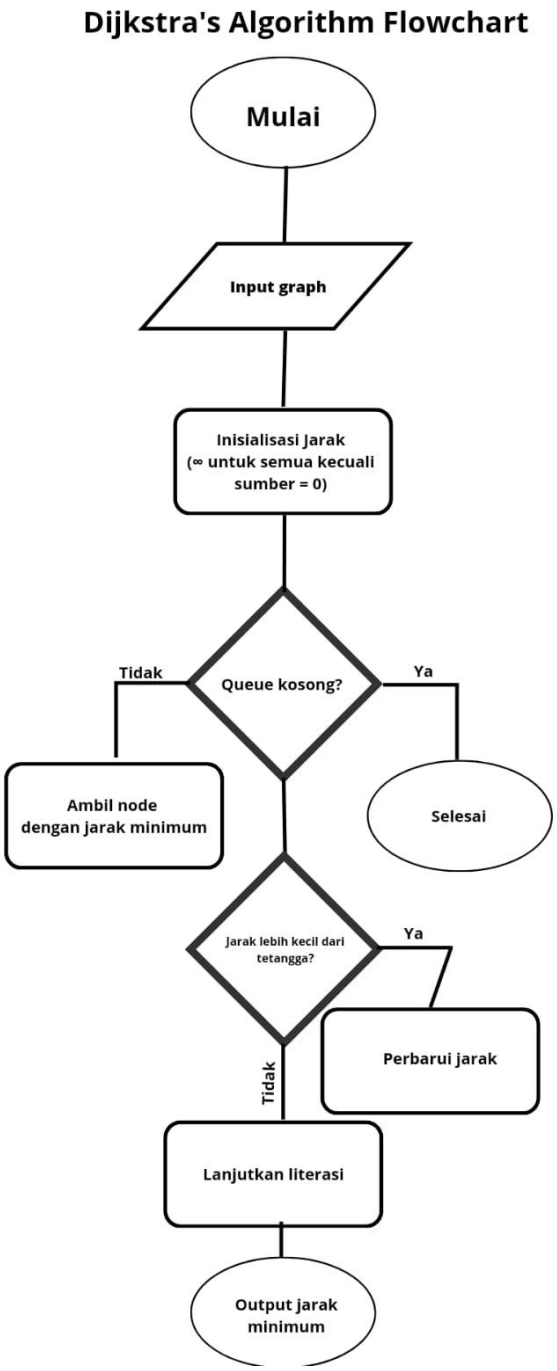


Flowchart (Graphs Algorithm)

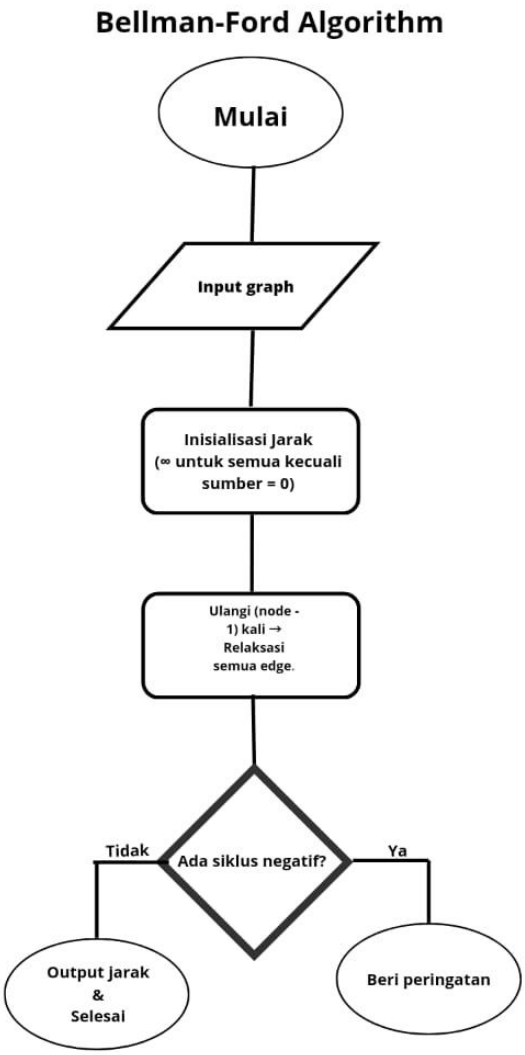
Kruskal’s Algorithm:



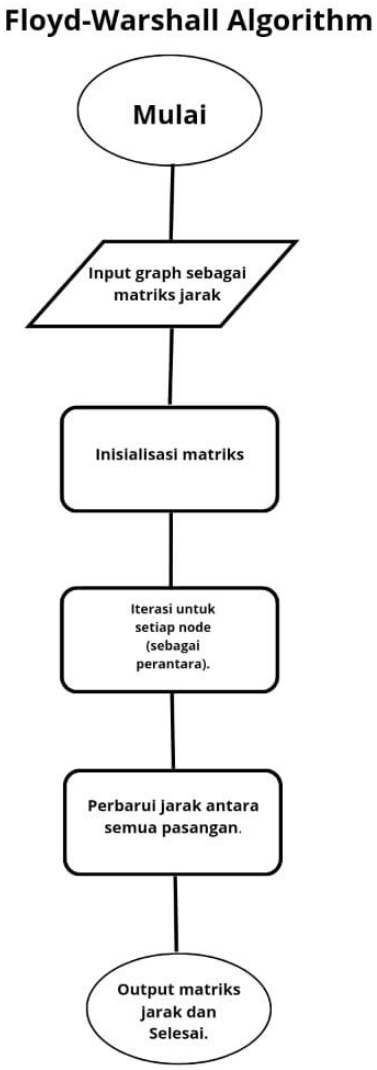
Dijkstra’s Algorithm:



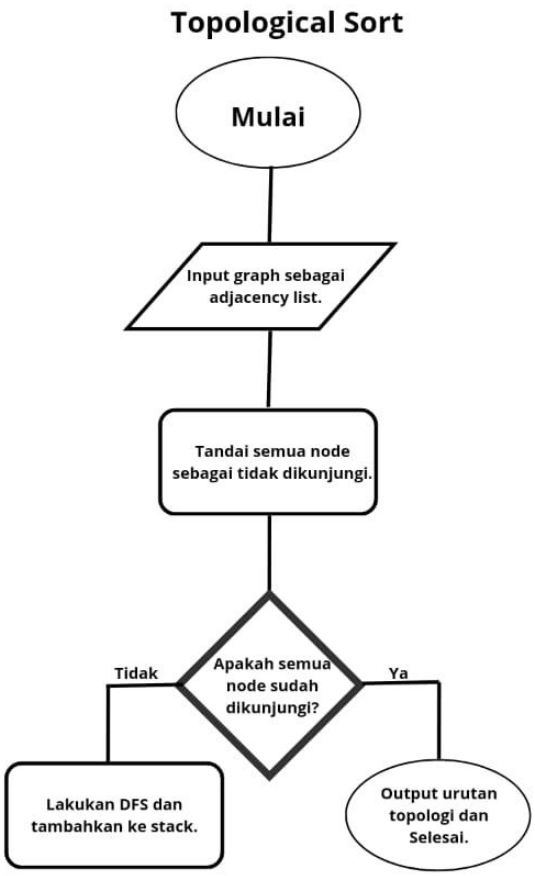
Bellman-Ford Algorithm:



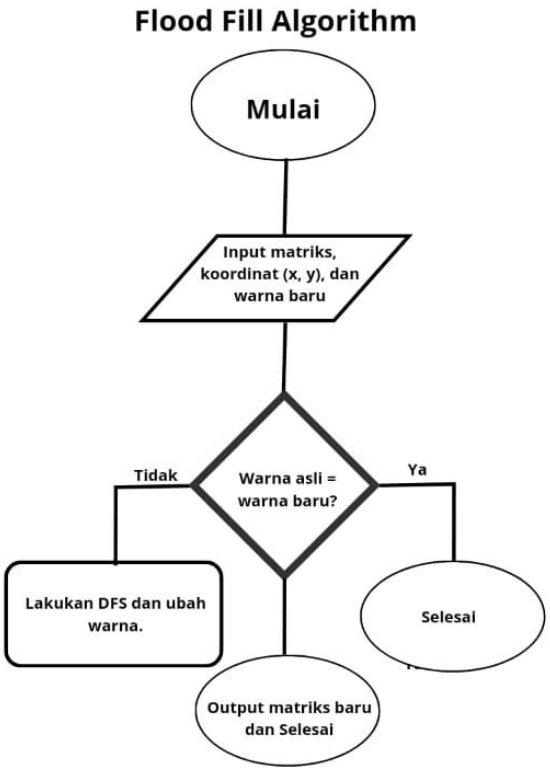
Floyd-Warshall Algorithm:



Topological Sort:



Flood Fill Algorithm:



Lee Algorithm: