Struktur Data : Sorting

Struktur Data - Sorting

- Sorting adalah proses menyusun data (array atau list) dalam urutan tertentu, biasanya menaik (ascending) atau menurun (descending)
- Manfaat Sorting
  - Mempermudah pencarian (seperti binary search)
  - Memudahkan pengolahan data
  - Mengoptimalkan algoritma lain

Struktur Data - Algoritma Sorting

- **Bubble Sort**
  - **Cara kerja:** Bandingkan elemen berurutan, tukar jika salah urutan, ulangi hingga tidak ada perubahan
  - Contoh: "*Membandingkan tinggi buku*", menyusun **buku pelajaran berdasarkan tinggi buku** di rak

Awal	:	(5, 3, 8, 4, 2)	
pass 1	:	(3, 5, 4, 2, 8)	<- 5 > 3 tukar, 5 < 8 tetap, 8 > 4 tukar, 8 > 2 tukar
pass 2	:	(3, 4, 2, 5, 8)	<- 3 < 5 tetap, 5 > 4 tukar, 5 > 2 tukar
pass 3	:	(3, 2, 4, 5, 8)	<- 3 > 2 tukar, 3 < 4 tetap
pass 4	:	(2, 3, 4, 5, 8)	<- 2 < 3 tetap
Hasil	:	(2, 3, 4, 5, 8)	

Struktur Data - Algoritma Sorting

- **Selection Sort**
  - **Cara kerja:** Cari elemen terkecil dan tempatkan di depan, lanjutkan dengan elemen berikutnya
  - Contoh: *"Pemilihan minimum lalu penempatan"*, Menyusun **kaos dari ukuran terkecil ke terbesar (S, M, L, XL), menyortir barang jualan di toko secara manual.**

Awal	:	(5, 3, 8, 4, 2)	
Step 1	:	(2, 3, 8, 4, 5)	<- minimum di index 4 tukar dengan index 0
Step 2	:	(2, 3, 8, 4, 5)	<- minimum di index 1 tetap
Step 3	:	(2, 3, 4, 8, 5)	<- minimum di index 3 tukar dengan index 2
Step 4	:	(2, 3, 4, 5, 8)	<- minimum di index 4 tukar dengan index 3
Step 5	:	(2, 3, 4, 5, 8)	<- terakhir
Hasil	:	(2, 3, 4, 5, 8)	

Struktur Data - Algoritma Sorting

- **Insertion Sort**
  - **Cara kerja:** Sisipkan elemen ke posisi yang sesuai dalam array terurut
  - Contoh: *"Menyisipkan di Tempat yang Pas"*, Menyusun dokumen atau file fisik.

Awal	:	(5, 3, 8, 4, 2)	
Step 1	:	(3, 5, 8, 4, 2)	<- 3 disisipkan sebelum 5
Step 2	:	(3, 5, 8, 4, 2)	<- 8 sudah benar tempatnya
Step 3	:	(3, 4, 5, 8, 2)	<- 4 disisipkan sebelum 5
Step 4	:	(2, 3, 4, 5, 8)	<- 2 disisipkan paling awal
Hasil	:	(2, 3, 4, 5, 8)	

Struktur Data - Algoritma Sorting

- **Quicksort Sort**
  - **Cara kerja:** Pilih pivot, bagi data jadi dua bagian, rekursif ke masing-masing bagian
  - Contoh: ***"Rekursif, pembagian dua sisi berdasarkan pivot"*** , Menyortir data banyak seperti daftar siswa di spreadsheet

Awal : [5, 3, 8, 4, 2]
Step 1 : Pilih pivot 5 (Elemen < 5) , bagi: [3, 4, 2][5][8]
Step 2 : Subarray kiri -> pivot 3: [2] [3] [4]
Step 3 : Gabung kiri: [2, 3, 4]
Step 4 : Gabung akhir: [2, 3, 4, 5, 8]
Hasil : [2, 3, 4, 5, 8]

Struktur Data - Algoritma Sorting

- **Counting Sort**
  - **Cara kerja:** Hitung jumlah elemen tiap nilai, susun berdasarkan jumlah
  - Contoh: *"Menghitung Lalu Menyusun"*, Membuat rekap absensi atau statistik

Awal : (5, 3, 8, 4, 2) <- Nilai maksimum = 8

Buat Array Frekuensi :

Index : 0 1 2 3 4 5 6 7 8

Frekuensi : [0, 0, 0, 0, 0, 0, 0, 0, 0]

Hitung Frekuensi Tiap Angka :

Index : 0 1 2 3 4 5 6 7 8

Frekuensi : [0, 0, 1, 1, 1, 1, 0, 0, 1]

Cetak ulang berdasarkan frekuensi : (2, 3, 4, 5, 8)

Hasil : (2, 3, 4, 5, 8)

Struktur Data - Algoritma Sorting

- **Radix Sort**
  - **Cara kerja:** Urut berdasarkan digit satu per satu (satuan, puluhan, dst)
  - Contoh: *"menyusun berdasarkan digit, dari kecil ke besar"*, menyortir data numerik seperti NIK, nomor invoice

Awal : (5, 3, 8, 4, 2)
Step 1 : Urut berdasarkan digit satuan: (2, 3, 4, 5, 8)
Step 2 : Tidak ada digit puluhan, selesai
Hasil : (2, 3, 4, 5, 8)

Struktur Data - Algoritma Sorting

- **Merge Sort**
  - **Cara kerja:** Divide and conquer: bagi dua array, urutkan masing-masing, lalu gabungkan
  - Contoh: ***"penggabungan dua daftar urut secara efisien"***, dua daftar belanja dari dua toko dan ingin **menggabungkan dan mengurutkan semuanya jadi satu**

Awal : (5, 3, 8, 4, 2)
Step 1 : Bagi dua: (5,3) (8, 4, 2)
Step 2 : Bagi lagi : (5) (3) (8) (4) (2)
Step 3 : Gabung : (3, 5,) (4, 8,) (2)
Step 4 : Gabung : (3, 5,) (2, 4, 8)
Step 5 : Gabung Akhir : (2, 3, 4, 5, 8)
Hasil : (2, 4, 8)

Struktur Data - Algoritma Sorting

❖ Bubble Sort

```python
def bubble_sort(arr):
    n = len(arr)
    for i in range(n):
        print(f"Pass {i + 1}: {arr}")
        for j in range(0, n - i - 1):
            if arr[j] > arr[j + 1]:
                arr[j], arr[j + 1] = arr[j + 1], arr[j]

arr = [64, 34, 25, 12, 22, 11, 90]
bubble_sort(arr)
print("Sorted array:", arr)
```

Struktur Data - Algoritma Sorting

❖ Selection Sort

```python
def selection_sort(arr):
    n = len(arr)
    for i in range(n):
        min_idx = i
        for j in range(i+1, n):
            if arr[min_idx] > arr[j]:
                min_idx = j
        arr[i], arr[min_idx] = arr[min_idx], arr[i]
        print(f"Step {i + 1}: {arr}")

arr = [64, 25, 12, 22, 11]
selection_sort(arr)
print("Sorted array:", arr)
```

Struktur Data - Algoritma Sorting

❖ Insertion Sort

```python
def insertion_sort(arr):
    for i in range(1, len(arr)):
        key = arr[i]
        j = i - 1
        while j >= 0 and key < arr[j]:
            arr[j + 1] = arr[j]
            j -= 1
        arr[j + 1] = key
        print(f"Step {i}: {arr}")

arr = [12, 11, 13, 5, 6]
insertion_sort(arr)
print("Sorted array:", arr)
```

Struktur Data - Algoritma Sorting

❖ Quicksort Sort

```python
def quicksort(arr):
    if len(arr) <= 1:
        return arr
    pivot = arr[0]
    less = [x for x in arr[1:] if x < pivot]
    greater = [x for x in arr[1:] if x >= pivot]
    print(f"Pivot: {pivot}, Less: {less}, Greater: {greater}")
    return quicksort(less) + [pivot] + quicksort(greater)

arr = [10, 7, 8, 9, 1, 5]
sorted_arr = quicksort(arr)
print("Sorted array:", sorted_arr)
```

Struktur Data - Algoritma Sorting

❖ Counting Sort

```python
def counting_sort(arr):
    print("Input:", arr)
    max_val = max(arr)
    count = [0] * (max_val + 1)

    for num in arr:
        count[num] += 1

    print("Frekuensi (count array):")
    for i, c in enumerate(count):
        if c > 0:
            print(f"Angka {i}: {c} kali")

    sorted_arr = []
    for i in range(len(count)):
        sorted_arr.extend([i] * count[i])

    print("Output (terurut):", sorted_arr)
    return sorted_arr

data = [5, 3, 8, 4, 2]
counting_sort(data)
```

Struktur Data - Algoritma Sorting

❖ Radix Sort

```
def counting_sort_digit(arr, exp):
    n = len(arr)
    output = [0] * n
    count = [0] * 10

    for num in arr:
        index = (num // exp) % 10
        count[index] += 1

    for i in range(1, 10):
        count[i] += count[i - 1]

    for i in range(n - 1, -1, -1):
        index = (arr[i] // exp) % 10
        output[count[index] - 1] = arr[i]
        count[index] -= 1

    for i in range(n):
        arr[i] = output[i]

    print(f"Hasil sort digit (exp={exp}):", arr)

def radix_sort(arr):
    print("Input:", arr)
    max_val = max(arr)
    exp = 1
    while max_val // exp > 0:
        counting_sort_digit(arr, exp)
        exp *= 10
    print("Output (terurut):", arr)
    return arr

data = [5, 3, 8, 4, 2]
radix_sort(data)
```

Struktur Data - Algoritma Sorting

❖ Merge Sort

```python
def merge_sort(arr):
    if len(arr) > 1:
        mid = len(arr) // 2
        left = arr[:mid]
        right = arr[mid:]

        print(f"Splitting: {arr}")
        merge_sort(left)
        merge_sort(right)

        i = j = k = 0

        while i < len(left) and j < len(right):
            if left[i] < right[j]:
                arr[k] = left[i]
                i += 1
            else:
                arr[k] = right[j]
                j += 1
            k += 1

        while i < len(left):
            arr[k] = left[i]
            i += 1
            k += 1
        while j < len(right):
            arr[k] = right[j]
            j += 1
            k += 1
        print(f"Merged: {arr}")

arr = [38, 27, 43, 3, 9, 82, 10]
merge_sort(arr)
print("Sorted array:", arr)
```

Struktur Data - Algoritma Sorting

- Contoh implementasi
  - **penjualan kaos (t-shirt)** menggunakan:
    - **Warna** : Merah, Putih, Hitam
    - **Ukuran** : S, M, L, XL
    - **Harga berdasarkan ukuran** :
      - S = 20.000
      - M = 25.000
      - L = 30.000
      - XL = 35.000
  - Analisis Statistik & Probabilitas dari data penjualan kaos

    Menampilkan:
    - **Probabilitas warna paling sering dibeli**
    - **Total penjualan selama 6 hari pertama**
    - **Visualisasi**

Tanggal	Warna	Ukuran	Jumlah	Harga Satuan	Total Harga
01-07-2025	Merah	M	2	25.000	50.000
01-07-2025	Putih	L	1	30.000	30.000
02-07-2025	Hitam	XL	3	35.000	105.000
03-07-2025	Merah	S	4	20.000	80.000
04-07-2025	Putih	M	2	25.000	50.000
04-07-2025	Hitam	L	1	30.000	30.000
05-07-2025	Merah	XL	1	35.000	35.000
05-07-2025	Putih	S	3	20.000	60.000
06-07-2025	Hitam	S	2	20.000	40.000
06-07-2025	Merah	L	3	30.000	90.000

Nama Variabel Statistik	Jenis Variabel Statistik	Contoh Nilai
Tanggal	Kualitatif – Data Waktu	01-07-2025
Warna	Kualitatif – Nominal	Merah, Putih, Hitam
Ukuran	Kualitatif – Ordinal	S, M, L, XL
Jumlah	Kuantitatif – Diskret	1, 2, 3, 4
Harga Satuan	Kuantitatif – Kontinu	20.000, 25.000
Total Harga	Kuantitatif – Kontinu	50.000, 105.000

Part - 01

```python
import pandas as pd
import matplotlib.pyplot as plt

# Data penjualan selama 6 hari pertama (10 baris)
data = [
    {"Tanggal": "2025-07-01", "Warna": "Merah", "Ukuran": "M", "Jumlah": 2, "Harga": 25000},
    {"Tanggal": "2025-07-01", "Warna": "Putih", "Ukuran": "L", "Jumlah": 1, "Harga": 30000},
    {"Tanggal": "2025-07-02", "Warna": "Hitam", "Ukuran": "XL", "Jumlah": 3, "Harga": 35000},
    {"Tanggal": "2025-07-03", "Warna": "Merah", "Ukuran": "S", "Jumlah": 4, "Harga": 20000},
    {"Tanggal": "2025-07-04", "Warna": "Putih", "Ukuran": "M", "Jumlah": 2, "Harga": 25000},
    {"Tanggal": "2025-07-04", "Warna": "Hitam", "Ukuran": "L", "Jumlah": 1, "Harga": 30000},
    {"Tanggal": "2025-07-05", "Warna": "Merah", "Ukuran": "XL", "Jumlah": 1, "Harga": 35000},
    {"Tanggal": "2025-07-05", "Warna": "Putih", "Ukuran": "S", "Jumlah": 3, "Harga": 20000},
    {"Tanggal": "2025-07-06", "Warna": "Hitam", "Ukuran": "S", "Jumlah": 2, "Harga": 20000},
    {"Tanggal": "2025-07-06", "Warna": "Merah", "Ukuran": "L", "Jumlah": 3, "Harga": 30000},
]

df = pd.DataFrame(data)
df["Total"] = df["Jumlah"] * df["Harga"]


total_penjualan = df["Total"].sum()


warna_order = ["Merah", "Putih", "Hitam"]
warna_terjual = df.groupby("Warna")["Jumlah"].sum().reindex(warna_order)
total_kaos = warna_terjual.sum()


probabilitas = (warna_terjual / total_kaos) * 100


print("Total Penjualan Selama 6 Hari: Rp {:,.0f}".format(total_penjualan))
print("\nProbabilitas Warna Paling Sering Dibeli:")
for warna, prob in probabilitas.items():
    print(f"{warna}: {prob:.2f}%")
```

Part - 02

```
warna_grafik = ["red", "white", "black"]
plt.figure(figsize=(8, 5))
plt.bar(probabilitas.index, probabilitas.values, color=warna_grafik, edgecolor='gray')
plt.title("Probabilitas Pembelian Kaos per Warna (6 Hari)")
plt.ylabel("Persentase (%)")
plt.xlabel("Warna")
plt.ylim(0, 50)
plt.xticks(rotation=0)
plt.grid(axis="y", linestyle="--", alpha=0.5)
plt.tight_layout()
plt.show()


warna_data = list(zip(warna_terjual.index, warna_terjual.values))

def bubble_sort(data):
    n = len(data)
    for i in range(n):
        for j in range(0, n - i - 1):
            if data[j][1] < data[j + 1][1]:
                data[j], data[j + 1] = data[j + 1], data[j]
    return data

sorted_data = bubble_sort(warna_data)

print("\nHasil Pengurutan (Bubble Sort) Berdasarkan Jumlah Terbanyak:")
for warna, jumlah in sorted_data:
    print(f"{warna}: {jumlah} kaos")


plt.figure(figsize=(8, 5))
sorted_warna, sorted_jumlah = zip(*sorted_data)
warna_grafik_sorted = ["red" if w == "Merah" else "white" if w == "Putih" else "black" for w in sorted_warna]

plt.bar(sorted_warna, sorted_jumlah, color=warna_grafik_sorted, edgecolor="gray")
plt.title("Hasil Bubble Sort: Warna Kaos Paling Banyak Terjual")
plt.ylabel("Jumlah Kaos")
plt.xlabel("Warna")
plt.grid(axis="y", linestyle="--", alpha=0.5)
plt.tight_layout()
plt.show()
```