Kunci Jawaban Tugas Pendahuluan - Modul 3: Sorting and Searching

Skema Penilaian (Total 100 Poin)

- Soal 1: 10 Poin
- Soal 2: 15 Poin
- Soal 3: 10 Poin
- Soal 4: 15 Poin
- Soal 5: 10 Poin
- Soal 6: 15 Poin
- Soal 7: 10 Poin
- Soal 8: 10 Poin
- Soal 9: 5 Poin

1. Jelaskan pengertian dari sorting dan searching! (10 Poin)

Jawaban (ID):

- **Sorting** adalah proses mengatur sekumpulan data ke dalam urutan tertentu (menaik atau menurun).
- **Searching** adalah proses menemukan posisi suatu data spesifik di dalam sebuah kumpulan data.

Answer (EN):

- **Sorting** is the process of arranging a collection of data into a specific order (ascending or descending).
- **Searching** is the process of finding the position of a specific piece of data within a data collection.

2. Sebutkan dan jelaskan jenis-jenis sorting! (minimal 4) (15 Poin)

Jawaban (ID):

1. **Bubble Sort:** Algoritma ini bekerja dengan berulang kali "menggeser" elemen terbesar ke posisi paling akhir. Caranya adalah dengan membandingkan setiap pasangan elemen yang bersebelahan dan menukarnya jika urutannya salah. Proses ini diibaratkan seperti gelembung yang naik ke permukaan.
2. **Selection Sort:** Algoritma ini membagi data menjadi dua bagian: terurut dan tidak terurut. Secara berulang, ia akan mencari elemen dengan nilai terkecil dari bagian yang tidak terurut dan menukarnya dengan elemen pertama dari bagian tidak terurut tersebut, sehingga batas bagian terurut bertambah satu.
3. **Insertion Sort:** Algoritma ini membangun larik terurut satu per satu. Ia mengambil satu elemen dari data yang tidak terurut, lalu mencari posisi yang tepat dan menyisipkannya ke dalam bagian data yang sudah terurut. Proses ini mirip seperti mengurutkan kartu di tangan.
4. **Merge Sort:** Menggunakan pendekatan "Bagi dan Taklukkan" (*Divide and Conquer*). Algoritma ini membagi data menjadi dua bagian yang sama besar secara rekursif hingga setiap bagian hanya berisi

satu elemen (yang dianggap sudah terurut). Kemudian, setiap bagian tersebut digabungkan (*merge*) kembali secara terurut.

5. **Quicksort:** Juga menggunakan pendekatan "Bagi dan Taklukkan". Algoritma ini memilih satu elemen sebagai "pivot", lalu mempartisi elemen lain ke dalam dua sub-larik berdasarkan apakah mereka lebih kecil atau lebih besar dari pivot. Sub-larik tersebut kemudian diurutkan secara rekursif.

Answer (EN):

1. **Bubble Sort:** This algorithm works by repeatedly "shifting" the largest element to its final position at the end. It does this by comparing each adjacent pair of elements and swapping them if they are in the wrong order. This process is likened to a bubble rising to the surface.
2. **Selection Sort:** This algorithm divides the data into two parts: sorted and unsorted. Repeatedly, it finds the element with the smallest value from the unsorted part and swaps it with the first element of that unsorted part, thereby expanding the boundary of the sorted part.
3. **Insertion Sort:** This algorithm builds the final sorted array one by one. It takes an element from the unsorted data, finds its correct position within the sorted part, and inserts it there. The process is similar to sorting playing cards in your hand.
4. **Merge Sort:** Uses a "Divide and Conquer" approach. This algorithm recursively divides the data into two equal halves until each sub-array contains only one element (which is considered sorted). Then, it merges these sub-arrays back together in a sorted manner.
5. **Quicksort:** Also uses a "Divide and Conquer" approach. This algorithm picks an element as a "pivot" and partitions the other elements into two sub-arrays according to whether they are less than or greater than the pivot. The sub-arrays are then sorted recursively.

3. Jelaskan jenis-jenis searching berikut: Sequential Search, Binary Search (10 Poin)

Jawaban (ID):

- **Sequential Search:** Mencari data dengan memeriksa setiap elemen satu per satu dari awal hingga akhir. Cocok untuk data yang tidak terurut.
- **Binary Search:** Hanya untuk data terurut. Mencari data dengan berulang kali membagi interval pencarian menjadi dua. Jika nilai elemen tengah kurang dari target, interval pencarian dipindah ke setengah kanan, dan sebaliknya.

Answer (EN):

- **Sequential Search:** Searches for data by checking every element one by one from beginning to end. Suitable for unsorted data.
- **Binary Search:** For sorted data only. It searches for data by repeatedly dividing the search interval in half. If the value of the middle element is less than the target, the interval is moved to the right half; otherwise, it is moved to the left half.

4. Jelaskan algoritma dari setiap jenis sorting dari soal no 2! (15 Poin)

Jawaban (ID):

1. **Bubble Sort:** Ulangi dari elemen pertama hingga akhir, bandingkan elemen i dengan i+1. Jika

data[i] > data[i+1], tukar. Terus ulangi proses ini hingga tidak ada pertukaran dalam satu iterasi penuh.
2. **Selection Sort:** Untuk setiap posisi i, cari indeks min dari elemen terkecil di sisa larik (i hingga n-1). Tukar elemen di posisi i dengan elemen di posisi min.
3. **Insertion Sort:** Mulai dari elemen kedua (i=1). Simpan nilai data[i] sebagai key. Bandingkan key dengan elemen-elemen sebelumnya (j=i-1). Geser elemen yang lebih besar dari key ke kanan, lalu sisipkan key di posisi yang kosong.
4. **Merge Sort:** Jika larik hanya punya satu elemen, kembalikan. Jika tidak, bagi larik menjadi dua. Panggil Merge Sort untuk kedua bagian. Gabungkan (merge) kedua bagian yang sudah terurut menjadi satu larik terurut.
5. **Quicksort:** Pilih elemen pivot (misalnya, elemen terakhir). Atur ulang larik sehingga elemen yang lebih kecil dari pivot berada di sebelah kiri, dan yang lebih besar di sebelah kanan. Panggil Quicksort secara rekursif untuk sub-larik kiri dan sub-larik kanan.

Answer (EN):

1. **Bubble Sort:** Iterate from the first to the last element, comparing element i with i+1. If data[i] > data[i+1], swap them. Keep repeating this process until no swaps occur in a full pass.
2. **Selection Sort:** For each position i, find the index min of the smallest element in the rest of the array (i to n-1). Swap the element at position i with the element at position min.
3. **Insertion Sort:** Start from the second element (i=1). Store data[i] as a key. Compare the key with the preceding elements (j=i-1). Shift elements greater than the key to the right, then insert the key into the empty spot.
4. **Merge Sort:** If the array has only one element, return it. Otherwise, divide the array in half. Call Merge Sort for both halves. Merge the two sorted halves into a single sorted array.
5. **Quicksort:** Choose a pivot element (e.g., the last element). Rearrange the array so that elements smaller than the pivot are on its left, and larger elements are on its right. Recursively call Quicksort for the left sub-array and the right sub-array.

5. Jelaskan algoritma dari setiap jenis searching dari soal no 3! (10 Poin)

Jawaban (ID):

- **Sequential Search:** Mulai dari indeks 0. Bandingkan elemen di indeks saat ini dengan data yang dicari. Jika sama, kembalikan indeks. Jika tidak, lanjut ke indeks berikutnya. Jika sampai akhir dan tidak ditemukan, kembalikan -1.
- **Binary Search:** Tentukan low=0 dan high=n-1. Selama low <= high, hitung mid = (low+high)/2. Jika data[mid] sama dengan target, kembalikan mid. Jika data[mid] < target, set low = mid+1. Jika data[mid] > target, set high = mid-1. Jika loop selesai, data tidak ditemukan.

Answer (EN):

- **Sequential Search:** Start at index 0. Compare the element at the current index with the target data. If they are equal, return the index. If not, move to the next index. If the end is reached and the data is not found, return -1.
- **Binary Search:** Set low=0 and high=n-1. While low <= high, calculate mid = (low+high)/2. If data[mid] equals the target, return mid. If data[mid] < target, set low = mid+1. If data[mid] > target,

set high = mid-1. If the loop finishes, the data was not found.

6. Pilih 2 jenis sorting lalu bandingkan keduanya! (15 Poin)

Jawaban (ID):
Perbandingan: Bubble Sort vs. Merge Sort

- **Bubble Sort:**
  - **Kelebihan:** Sangat sederhana untuk dipahami dan diimplementasikan.
  - **Kekurangan:** Sangat tidak efisien untuk data dalam jumlah besar, dengan kompleksitas waktu rata-rata dan terburuk $O(n^2)$.
- **Merge Sort:**
  - **Kelebihan:** Jauh lebih efisien dengan kompleksitas waktu O(n log n) di semua kasus (terbaik, rata-rata, terburuk). Stabil dan dapat diandalkan untuk data besar.
  - **Kekurangan:** Lebih kompleks untuk diimplementasikan dan membutuhkan memori tambahan (O(n)) untuk proses penggabungan.

Answer (EN):
Comparison: Bubble Sort vs. Merge Sort

- **Bubble Sort:**
  - **Advantage:** Very simple to understand and implement.
  - **Disadvantage:** Highly inefficient for large datasets, with an average and worst-case time complexity of $O(n^2)$.
- **Merge Sort:**
  - **Advantage:** Much more efficient with a time complexity of O(n log n) in all cases (best, average, worst). It is stable and reliable for large datasets.
  - **Disadvantage:** More complex to implement and requires extra memory space (O(n)) for the merging process.

7. Jelaskan perbedaan kedua jenis searching pada soal no 3! (10 Poin)

Jawaban (ID):

- **Perbedaan Utama:** Syarat data. **Sequential Search** bisa digunakan pada data terurut maupun tidak terurut. **Binary Search** *wajib* digunakan pada data yang sudah terurut.
- **Kondisi Lebih Efisien:**
  - **Sequential Search** lebih efisien untuk **data kecil** atau jika data yang dicari kemungkinan besar ada di **awal** kumpulan data.
  - **Binary Search** jauh lebih efisien untuk **data besar**, karena secara drastis mengurangi jumlah perbandingan yang perlu dilakukan (kompleksitas O(log n) vs O(n)).

Answer (EN):

- **Main Difference:** Data requirement. **Sequential Search** can be used on both sorted and unsorted data. **Binary Search** *must* be used on data that is already sorted.
- **More Efficient Condition:**
  - **Sequential Search** is more efficient for **small datasets** or if the target data is likely to be at the **beginning** of the collection.

- **Binary Search** is far more efficient for **large datasets**, as it drastically reduces the number of comparisons needed (O(log n) complexity vs. O(n)).

8. Gambarkan visualisasi dari salah satu jenis sorting dalam LinkedList! (10 Poin)

Jawaban (ID/EN):
Visualisasi Bubble Sort pada Linked List: [7] -> [4] -> [9] -> [2] -> NULL
Pass 1:

1. Bandingkan 7 dan 4. Tukar. List: [4] -> [7] -> [9] -> [2] -> NULL
2. Bandingkan 7 dan 9. Tidak ada tukar. List: [4] -> [7] -> [9] -> [2] -> NULL
3. Bandingkan 9 dan 2. Tukar. List: [4] -> [7] -> [2] -> [9] -> NULL (9 di posisi akhir)

Pass 2:

1. Bandingkan 4 dan 7. Tidak ada tukar. List: [4] -> [7] -> [2] -> [9] -> NULL
2. Bandingkan 7 dan 2. Tukar. List: [4] -> [2] -> [7] -> [9] -> NULL (7 di posisi benar)

Pass 3:

1. Bandingkan 4 dan 2. Tukar. List: [2] -> [4] -> [7] -> [9] -> NULL (4 di posisi benar)

Hasil Akhir: [2] -> [4] -> [7] -> [9] -> NULL

9. Gambarkan visualisasi dari salah satu jenis searching dalam LinkedList! (5 Poin)

Jawaban (ID/EN):
Visualisasi Sequential Search pada Linked List: Mencari nilai 9 dalam [4] -> [7] -> [2] -> [9] -> NULL

1. Mulai dari head. Apakah 4 == 9? Tidak. Pindah ke next.
   [4]* -> [7] -> [2] -> [9] -> NULL
2. Pointer sekarang di 7. Apakah 7 == 9? Tidak. Pindah ke next.
   [4] -> [7]* -> [2] -> [9] -> NULL
3. Pointer sekarang di 2. Apakah 2 == 9? Tidak. Pindah ke next.
   [4] -> [7] -> [2]* -> [9] -> NULL
4. Pointer sekarang di 9. Apakah 9 == 9? Ya. Data ditemukan.
   [4] -> [7] -> [2] -> [9]* -> NULL