Jurnal Riset Multidisiplin Edukasi
https://journal.hasbaedukasi.co.id/index.php/jurmie
Halaman : 864-874

PENGGABUNGAN SIFAT BINARY MIN HEAP DAN MAX HEAP PADA STRUKTUR DATA MIN-MAX HEAP DAN MAX-MIN HEAP: KONSEP DAN VISUALISASI

Edison Pardengganan Siahaan
Teknik Informatika – Fakultas Teknik Universitas Mpu Tantular
Email: edisonsiahaan@mputantular.ac.id[1], edison1814@gmail.com[2]

Informasi

Volume : 3
Nomor : 1
Bulan : Januari
Tahun : 2026
E-ISSN : 3062-9624

Abstract

This paper outlines the organizational mechanisms of two variants of the Binary Heap data structure, namely the Binary Min-Max Heap and the Binary Max-Min Heap. These variants are derived from a combination of characteristics found in the Binary Min Heap and Binary Max Heap, allowing users to efficiently retrieve both minimum and maximum values from a collection of stored and organized data. In addition to discussing the conceptual and operational aspects of these structures, the paper also presents the design of a software tool developed to visualize the organization process, particularly the procedures for inserting new nodes and deleting existing nodes from the data structures. The visualization is represented through a class diagram, which illustrates the internal structure of the program, including the attributes and methods involved in its implementation. This design aims to assist users in gaining a more intuitive understanding of the internal workings of both Binary Heap variants.

Keyword: *Binary Heap, Complete Binary Tree, Min-Max Heap, Max-Min Heap.*

Abstrak

Makalah ini menguraikan mekanisme pengorganisasian dua varian struktur data Binary Heap, yakni Binary Min-Max Heap dan Binary Max-Min Heap. Kedua varian tersebut merupakan hasil kombinasi karakteristik dari struktur data Binary Min Heap dan Binary Max Heap, sehingga pengguna setiap saat dapat memperoleh nilai minimum dan maksimum dari sejumlah data yang disimpan dan diorganisasikan dengan dua varian struktur data ini. Selain membahas aspek konseptual dan operasional dari kedua struktur data tersebut, makalah ini juga menyajikan perancangan perangkat lunak yang digunakan untuk memvisualisasikan proses pengorganisasian struktur data tersebut khususnya langkah-langkah dalam menyisipkan simpul baru dan langkah-langkah dalam menghapus simpul dari struktur data tersebut. Visualisasi ini diwujudkan melalui sebuah class diagram yang menggambarkan struktur internal dari program, termasuk atribut dan metode yang digunakan dalam implementasinya. Perancangan ini bertujuan untuk membantu pengguna memahami secara lebih intuitif cara kerja kedua varian struktur data Binary Heap.

Kata Kunci: *Binary Heap, Complete Binary Tree, Min-Max Heap, Max-Min Heap.*

A. PENDAHULUAN

Struktur data secara umum merupakan metode untuk menyimpan, menyusun, dan mengelola data dalam media penyimpanan komputer agar dapat diakses dan dimanfaatkan

secara efisien. Secara garis besar, struktur data terbagi menjadi dua kategori utama, yaitu Struktur Data Primitif dan Struktur Data Non-Primitif. Contoh dari struktur data primitif mencakup tipe data seperti boolean, real, integer, dan character. Sementara itu, struktur data non-primitif terbagi lagi menjadi dua bentuk, yaitu Struktur Data Linear dan Struktur Data Non-Linear. Yang termasuk dalam struktur data linear antara lain array, matriks, linked list, stack, dan queue. Adapun struktur data non-linear mencakup Tree dan Graph.

Pohon biner (binary tree) merupakan jenis struktur data Tree yang terdiri dari himpunan terbatas simpul (node). Di dalam himpunan ini terdapat satu simpul utama yang disebut **akar (root node)**, serta dua **sub pohon biner** yang masing-masing dikenal sebagai **left subtree (subpohon kiri)** dan **right subtree (subpohon kanan)**. Kedua subpohon tersebut bersifat **saling terpisah (disjoint)** dan masing-masing juga merupakan pohon biner.

Complete Binary Tree (Pohon Biner Lengkap) adalah jenis pohon biner di mana **setiap level terisi penuh**, kecuali **level terdalam** yang boleh tidak sepenuhnya terisi. Namun, pada level terdalam ini, **semua simpul daun harus berada di sisi kiri**, sehingga tidak ada kekosongan di bagian kiri pohon.

Binary Heap adalah struktur data berbasis **Complete Binary Tree**, di mana setiap level dari pohon harus terisi secara penuh, kecuali **level terdalam** yang diperbolehkan tidak lengkap, dengan syarat **penyisipan simpul dilakukan secara berurutan dari kiri ke kanan**. Secara matematis, **tinggi (height)** dari Binary Heap yang terdiri atas **n simpul** adalah $\mathbf{\lfloor \log_2 n \rfloor}$, yang menunjukkan jumlah level dari akar hingga simpul terdalam.

Binary Heap umumnya diklasifikasikan menjadi dua jenis utama, yaitu Binary Min Heap dan Binary Max Heap. Pada Binary Min Heap, karakteristik yang harus dipenuhi adalah bahwa setiap simpul induk (parent node) harus memiliki nilai yang lebih kecil atau sama dengan nilai kedua simpul anaknya (child nodes). Sebaliknya, pada Binary Max Heap, setiap simpul induk harus memiliki nilai yang lebih besar atau sama dengan nilai dari kedua simpul anaknya. Ketentuan ini memastikan bahwa struktur heap mempertahankan sifat min-heap atau max-heap sepanjang waktu.

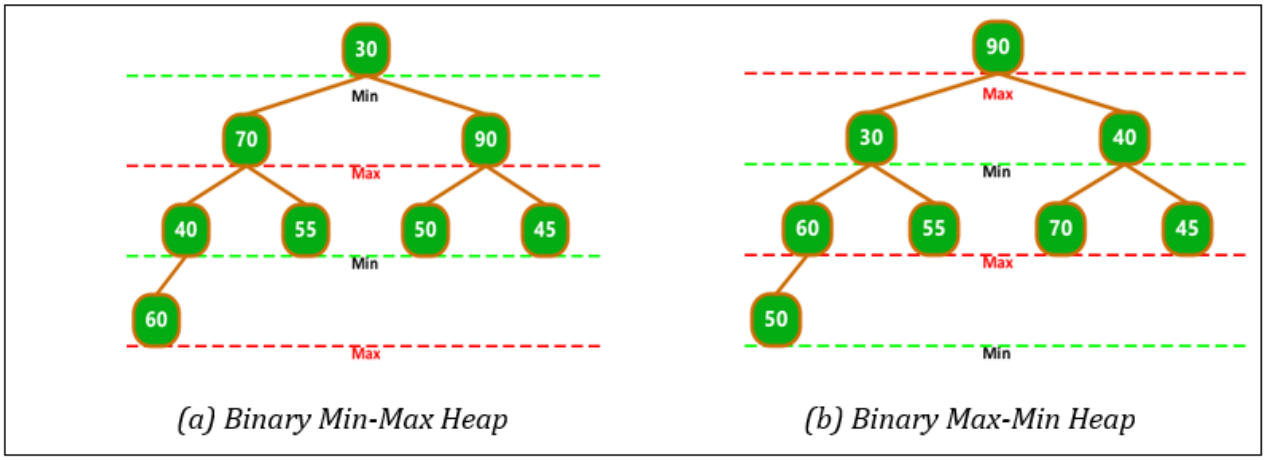
Makalah ini membahas dua varian dari struktur **Binary Heap** yang mengintegrasikan karakteristik **Min Heap** dan **Max Heap**, yaitu **Binary Min-Max Heap** dan **Binary Max-Min Heap**. Selain menguraikan prinsip dasar dan mekanisme kerja dari kedua struktur tersebut, makalah ini juga menyajikan **perancangan program visualisasi** yang bertujuan untuk memperlihatkan bagaimana struktur data **Min-Max Heap** dan **Max-Min Heap**

diorganisasikan. Program ini dirancang untuk mendukung pemahaman pengguna terhadap proses penyusunan dan sifat hierarkis yang berlaku pada kedua jenis heap tersebut.

B. METODE PENELITIAN

Struktur data **Binary Min-Max Heap** dan **Binary Max-Min Heap** merupakan varian dari struktur **Binary Heap** yang mengintegrasikan karakteristik dari kedua jenis heap, yaitu **Min Heap** dan **Max Heap**. Pada kedua struktur ini, sejumlah prinsip dasar diterapkan, antara lain: pertama, Binary Heap merupakan pohon biner lengkap (**Complete Binary Tree**), di mana setiap level terisi penuh kecuali level terakhir, yang diizinkan untuk tidak penuh selama pengisian dilakukan dari kiri ke kanan. Kedua, tinggi pohon (**height**) dengan jumlah simpul nnn dinyatakan sebagai $\lfloor \log_2 n \rfloor$. Selanjutnya, pada struktur **Binary Min-Max Heap** dan **Binary Max-Min Heap**, setiap simpul yang berada pada **level Min** harus memiliki nilai lebih kecil dibandingkan semua simpul pada level di bawahnya, sedangkan setiap simpul pada **level Max** harus memiliki nilai lebih besar daripada seluruh simpul di bawahnya. Secara khusus, pada **Binary Min-Max Heap**, simpul akar (root) terletak pada level Min, sedangkan pada **Binary Max-Min Heap**, simpul akar berada pada level Max.

Gambar 1. dibawah ini menunjukkan contoh dari *Binary Min-Max Heap dan Binary Max-Min Heap*, yaitu pada gambar 1.(a) adalah contoh *Binary Min-Max Heap* sedangkan pada gambar 1.(b) adalah contoh *Binary Max-Min Heap*.



Gambar 1. *Binary Min-Max Heap* dan *Binary Max-Min Heap*

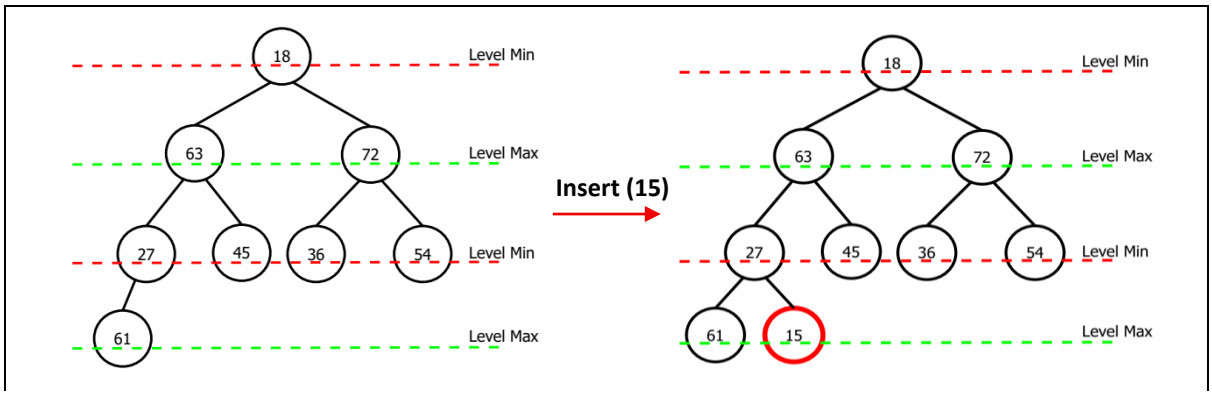
Berdasarkan ilustrasi Binary Heap pada Gambar 1, dapat diamati bahwa karakteristik **Min Heap** dan **Max Heap** dalam struktur data **Binary Heap** tersebut diterapkan secara bergantian pada setiap level pohon. Pada struktur **Binary Min-Max Heap**, level 0 atau level simpul akar memiliki karakteristik **Min Heap**, sehingga level akar di Min-Max adalah **level bertipe Min**. Sebaliknya, pada **Binary Max-Min Heap**, level 0 menerapkan karakteristik **Max**

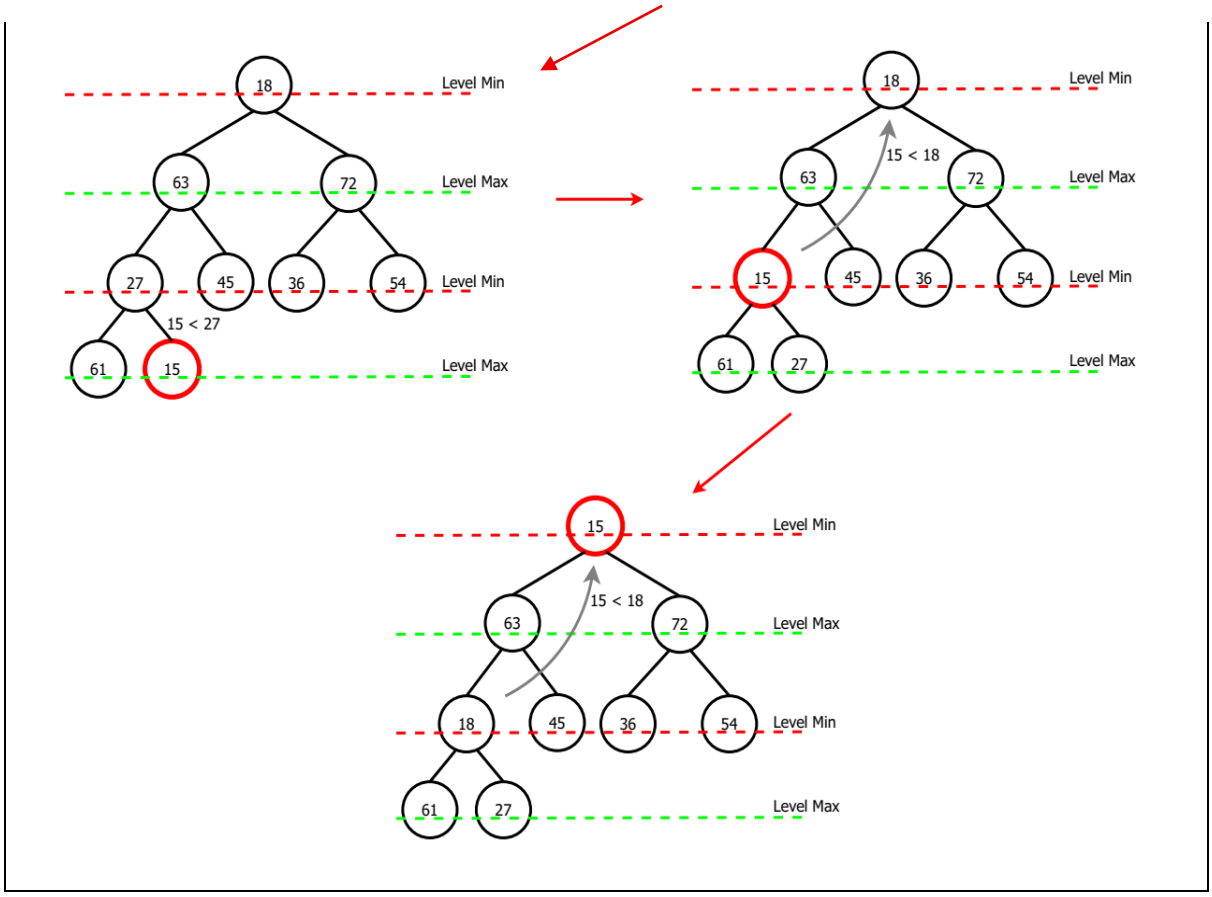
Heap, sehingga level akar-nya adalah **level bertipe Max**.

C. HASIL DAN PEMBAHASAN

Penyisipan Simpul Baru Pada Struktur Data Binary Min-Max Heap dan Binary Max-Min Heap

Prosedur penyisipan simpul baru pada struktur Binary Min-Max Heap dan Binary Max-Min Heap dilakukan melalui beberapa tahap. Pertama, dilakukan penelusuran secara Breadth-First Search (Level-Order Traversal) untuk menemukan posisi kosong pertama yang tersedia, kemudian simpul baru ditempatkan pada posisi tersebut guna menjaga sifat Complete Binary Tree. Selanjutnya, nilai kunci dari simpul yang disisipkan dibandingkan dengan nilai simpul induk dan moyang-nya. Apabila ditemukan pelanggaran terhadap aturan hierarki nilai yang berlaku pada Min-Max Heap atau Max-Min Heap, maka dilakukan penyesuaian melalui pertukaran posisi antara simpul baru dengan induk atau keturunan yang relevan, sesuai dengan tipe level (Min atau Max) tempat simpul tersebut berada. Proses ini diulangi hingga struktur heap memenuhi seluruh kriteria karakteristik yang ditetapkan untuk Binary Min-Max Heap dan Binary Max-Min Heap. Gambar 2 berikut merupakan contoh urut-urutan langkah penyisipan suatu simpul baru pada struktur data Binary Min-Max Heap.





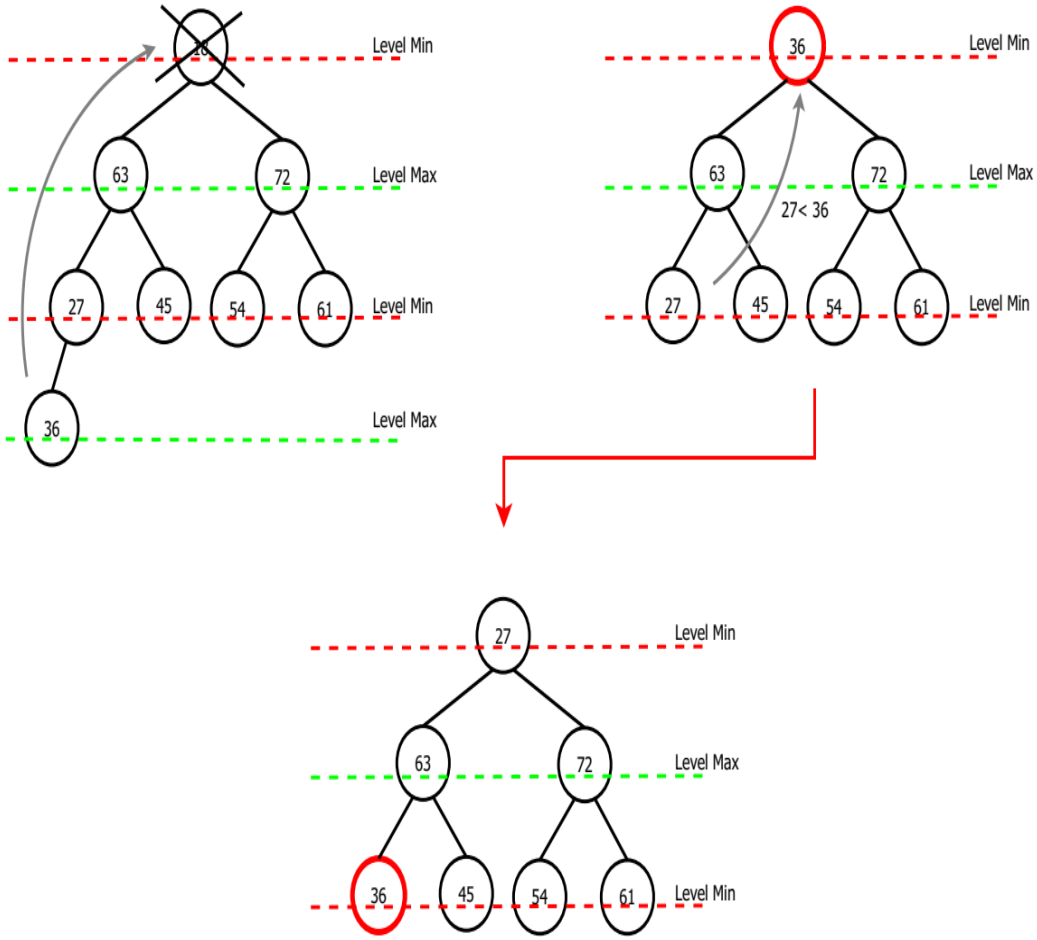
Gambar 2. Contoh Urut-urutan Langkah Penyisipan Simpul Baru pada Binary Min-Max

Penghapusan/Pengeluaran Simpul Akar (Root) Pada Binary *Min-Max Heap* (Mendapatkan Nilai Minimum) dan Binary Max-Min Heap (Mendapatkan Nilai Maksimum)

Proses penghapusan simpul akar pada struktur Binary Min-Max Heap dan Binary Max-Min Heap dapat dijabarkan dalam beberapa langkah berikut. Pertama, simpul akar dihapus dari pohon heap. Selanjutnya, simpul terakhir yang ditemukan berdasarkan urutan penelusuran level order dipindahkan ke posisi akar untuk mempertahankan sifat pohon biner lengkap. Kemudian, nilai atau kunci simpul yang baru dipindahkan dibandingkan dengan nilai anak-anak dan keturunannya. Jika terjadi pelanggaran terhadap karakteristik struktural Binary Min-Max Heap atau Binary Max-Min Heap, maka dilakukan penyesuaian melalui pertukaran posisi simpul secara berulang hingga kondisi heap memenuhi kriteria yang ditetapkan.

Pada Binary Min-Max Heap, jika simpul akar dikeluarkan maka simpul akar tersebut akan berisi nilai terkecil dari seluruh nilai yang ada pada simpul-simpul di Binary Min-Max Heap, sebaliknya pada Binary Max-Min Heap maka simpul akar yang dikeluarkan tersebut akan berisi nilai terbesar dari seluruh nilai yang ada pada simpul-simpul Binary Max-Min

Heap. Gambar 3 dibawah ini menunjukkan contoh langkah-langkah penghapusan/pengeluaran simpul akar dari suatu struktur data Binary Min-Max Heap.



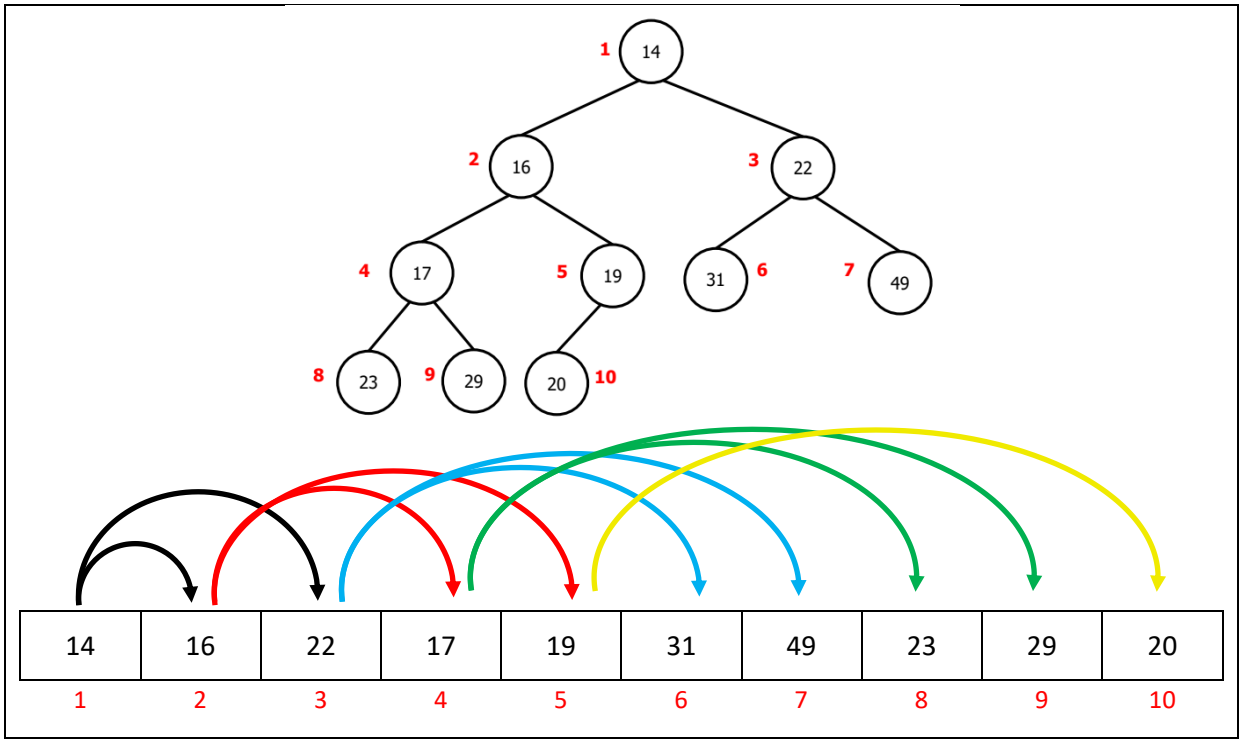
Nilai Simpul Akar yang dikeluarkan adalah : **18** (Nilai Terkecil)

Gambar 3. Contoh Langkah-Langkah Penghapusan Simpul Akar (Ekstraksi Nilai Terkecil)

Representasi Struktur Data Binary *Min-Max Heap* dan Max-Min Heap ke Dalam Struktur Data Array

Binary Min-Max Heap dan Max-Min Heap, sebagaimana jenis struktur data pohon biner heap lainnya, dapat direpresentasikan secara efisien menggunakan struktur data array. Representasi array yang digunakan untuk menyimpan simpul-simpul dari Binary Min-Max Heap dan Max-Min Heap dengan indeks awal 1 memenuhi aturan-aturan sebagai berikut: *(1)* simpul akar (root node) disimpan pada indeks pertama array, yaitu indeks 1; *(2)* jika suatu simpul disimpan pada indeks ke-i, maka anak kiri (left child) dari simpul tersebut berada pada indeks $2i$, sedangkan anak kanan (right child) berada pada indeks $2i + 1$; dan *(3)* untuk simpul selain akar, jika simpul tersebut berada pada indeks ke-i, maka simpul induknya (parent node) berada pada indeks $\lfloor i/2 \rfloor$. Representasi ini memungkinkan pengaksesan dan

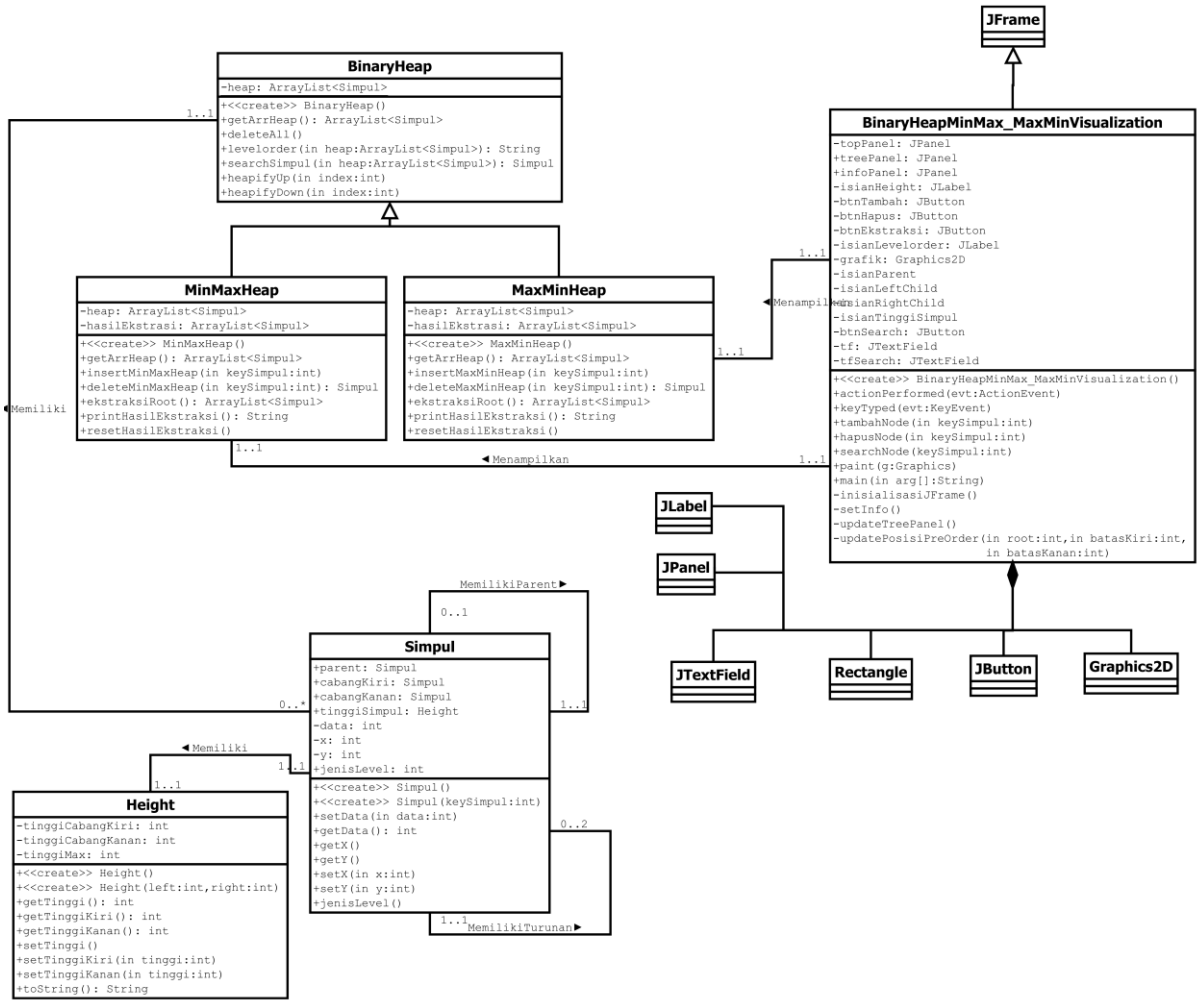
manipulasi simpul secara sistematis dan efisien dalam konteks implementasi struktur data heap. Contoh representasi dan pemetaan struktur data Binary Heap ke Struktur Data Array dengan indeks awal 1 dapat dilihat pada gambar 4. dibawah ini.



Gambar 4. Representasi Struktur Data Binary Min-Max Heap dan Max-Min Heap Jika disimpan Pada Struktur Data Array

Rancangan Program Binary Min-Max Heap dan Max-Min Heap Tree Visualization

Untuk memfasilitasi pemahaman pengguna terhadap cara kerja dan pengorganisasian struktur data Binary Min-Max Heap dan Max-Min Heap, telah dikembangkan sebuah program visualisasi yang merepresentasikan kedua struktur tersebut secara grafis. Visualisasi ini dirancang untuk menampilkan bagaimana elemen-elemen dalam heap disusun dan dikelola. Sebagai bagian dari dokumentasi dan perancangan perangkat lunak, class diagram digunakan untuk menggambarkan struktur kelas dan relasi antar komponen dalam program, sehingga memperjelas arsitektur sistem yang diimplementasikan. Gambar 5 dibawa ini adalah class diagram dari program tersebut.



Gambar 5. Class Diagram Program Visualisasi Binary Min-Max Heap dan Max-Min Heap

Tampilan Program Visualisasi Struktur Data Binary Min-Max Heap dan Max-Min Heap

Perangkat lunak berupa program Visualisasi Binary Min-Max Heap dan Max-Min Heap berbasis java telah berhasil dibuat dengan menggunakan perangkat lunak Apache NetBeans IDE Versi 20 dan Java JDK 21. Tampilan dari program yang dijalankan dan berhubungan dengan Binary Min-Max Heap dapat dilihat pada gambar 6 sebagai berikut.

Tampilan Min-Max Heap Sebelum Tampilan Min-Max Heap Sesudah

penambahan Simpul

penambahan Simpul dengan key 15

(a) Tampilan Penambahan Simpul Pada Binary Min-Max Heap

Tampilan Min-Max Heap Sebelum Ekstraksi Root

Tampilan Min-Max Heap Setelah ekstraksi Root dengan key 15

(b) Tampilan Hasil Ekstraksi Elemen Root dari Min-Max Heap

Gambar 6. Tampilan Program Visualisasi Binary Min-Max Heap

Tampilan dari program yang dijalankan dan berhubungan dengan Binary Max-Min Heap dapat dilihat pada gambar 7 sebagai berikut.

Tampilan Max-Min Heap Sebelum Penambahan Simpul

Tampilan Max-Min Heap Sesudah penambahan Simpul dengan key 15

(a) Tampilan Penambahan Simpul pada Binary Max-Min Heap

Tampilan MaxMin-Heap Sebelum Ekstraksi Root

Tampilan MaxMin-Heap Setelah Ekstraksi Root dengan Key 70

(b) Tampilan Hasil Ekstraksi Elemen Root dari Max-Min Heap

Gambar 7. Tampilan Program Visualisasi Binary Max-Min Heap

D. KESIMPULAN

Makalah ini membahas prinsip dasar serta mekanisme pengorganisasian dua varian dari struktur data Binary Heap, yaitu Binary Min-Max Heap dan Max-Min Heap. Pembahasan mencakup prosedur pembentukan struktur, operasi penyisipan (insertion), dan penghapusan (deletion) simpul pada kedua varian tersebut. Selain penjabaran teoritis, makalah ini juga menyajikan rancangan perangkat lunak yang dirancang untuk memvisualisasikan proses pengorganisasian masing-masing struktur data. Visualisasi ditampilkan dalam bentuk diagram pohon, sehingga memungkinkan pengguna untuk memahami struktur dan dinamika perubahan yang terjadi pada Binary Min-Max Heap dan Max-Min Heap. Perangkat lunak ini diharapkan dapat menjadi alat bantu dalam pembelajaran dan analisis kedua varian struktur data tersebut. Seluruh implementasi program, termasuk kode sumber, tersedia dan dapat diakses melalui cloud storage yang disediakan oleh penulis.

E. DAFTAR PUSTAKA

Apache Software Foundation, “Apache Netbean - Development Environment, Tooling Platform and Application Framework”, https://netbeans.apache.org/, Diakses : 17 April 2023

Farrell, J., “JavaTM Programming 8th Edition”, Cengage Learning, 2015

Fowler, M., “UML Distilled Edisi 3”, Andi Yogyakarta, 2004

Goodrich, M.T., Tamassia,R., Goldwasser, M.H. “Data Structures and Algorithms in Java”, Wiley & Sons, Inc., 2014

Munir, R., Lidya.L, “Algoritma dan Pemrograman”, Andi Yogyakarta, 2016.

Nugroho, A., “Rekayasa Perangkat Lunak Berorientasi Objek”, Andi Yogyakarta, 2010

Siahaan, E. "Pengembangan Perangkat Lunakan AVL Tree Visualization Berbasis Java", Visikom Universitas Mpu Tantular, 2023

Thareja, R., "Data Structures Using C 2nd Edition", 2014

Wikipedia, "AVL Tree", https://en.wikipedia.org/wiki/AVL_tree, Diakses tgl : 22 April 2024

Zakaria, T.M., Prijono, A., "Konsep dan Implementasi Struktur Data", 2005