**PENERAPAN BINARY SEARCH TREE PADA PROGRAM PERUMAHAN ‘AVA’**

LAPORAN PROYEK AKHIR

MATA KULIAH COMP6362 – DATA STRUCTURES

KELAS BF20



Oleh:

1. 2440060502 - Achmad Aqil Susanto
2. 2440065346 - Aldi Dwi Purnomo
3. 2440065283 - Viana Salsabila Tauda

Semester Genap 2021

MALANG

**LEMBAR PERSETUJUAN PROYEK AKHIR**

**PENERAPAN BINARY SEARCH TREE PADA PROGRAM PERUMAHAN ‘AVA’**

MATA KULIAH COMP6362 – DATA STRUCTURES

KELAS BF20

Semester Genap 2021

Laporan akhir proyekini adalah benar karya kami:

|  |  |  |
| --- | --- | --- |
| <ttd>  Achmad Aqil Susanto  2440060502 | <ttd>  Aldi Dwi Purnomo  2440065346 | <ttd>  Viana Salsabila Tauda  2440065283 |

**Malang, 20 Juni 2021**

**<ttd>**

**Elizabeth Paskahlia G**

**D5707**

**LATAR BELAKANG**

Struktur data merupakan mata kuliah yang sudah tidak asing lagi bagi para mahasiswa yang bergelut di dalam bidang studi Teknik Informatika. Hal ini dikarenakan struktur data menjadi salah satu dasar bagi seorang programmer untuk mengembangkan programnya. Struktur data sendiri bisa digunakan diberbagai bahasa seperti bahasa C, C++, Phyton, hingga Java Script. Namun dalam pengaplikasiannya kami menggunakan bahasa C dalam pembuatan program. Hal ini bukan tanpa alasan, karena Universitas Bina Nusantara melakukan pengajaran Struktur Data dengan mengguakan bahasa C.

Seperti yang telah kita pelajari di dalam mata kuliah Struktur Data ini terdapat berbagai macam topik diantaranya ada linked list, tree, binary tree, binary search tree, dan sebagainya. Sebagai perencanaan kami dalam menyelesaikan proyek akhir mata kuliah *Data Structures* topik yang akan kami angkat adalah mengenai *Binary Search Tree*.

Topik *Binary Search Tree* kami ambil dikarenakan pada topik ini kami masih bisa melakukan penganalogian terhadap coding dari BST ini. Selain itu penerapan BST dalam kehidupan sehari-hari pun cukup luas, sehingga mudah dalam pengimplementasian studi kasus. Topik ini juga telah dilakukan pengajaran oleh dosen pemangku sehingga kami lebih mudah dan lebih cepat dalam melakukan pengerjaan produk. Banyaknya sumber refrensi juga menjadi salah satu tujuan kami mengambil topik *Binary Seacrh Tree* ini untuk proyek akhir kami.

Program yang kami kembangkan ini adalah program pengimplementasian terhadap *Binary Search Tree*. Dan permasalahan yang kami angkat mengenai data perumahan. Selain itu melakukan penyimpanan data kami menggunakan salah satu topik pada mata kuliah *algorithm and programming* yaitu *file processing*. Dimana semua data akan disimpan di dalam data dan dibaca oleh program. Sehingga jika program ditutup data yang telah dimasukan tetap tersimpan.

**TINJAUAN PUSTAKA**

1. **Binary Seach Tree**

Seperti yang kita ketahui tree terbagai kedalam berbagai macam ada tries dan binary tree. Tries adalah tree yang tidak memiliki batasan anak yang harus dimiliki oleh sebuah node, misalkan suatu node memiliki anak sebanyak tiga ataupun lebih. Sedangkan binary tree adalah sebuah tree yang hanya dapat memuat anak paling banyak adalah dua buah anak.

Binary tree sendiri terbagi ke dalam berbagai jenis, salah satunya adalah Binary Search Tree. Binary Search Tree atau yang disebut sebagai tree terurut atau sorting tree merupakan tree yang hanya memiliki anak tidak lebih dari dua buah dan dalam penentuan peletakannya terdapat beberapa persyaratan-persyaratan. Persyaratan-persyaratan di dalam Binary Search Tree sendiri berpengaruh dalam pemberlakukan insert dan juga deleting di dalam Binary Search Tree ini sendiri. Untuk memperjelas mengenai persyaratan di dalam Binary Search Tree penulis akan menjelaskan melalui proses insert dan selet pada Binary Search Tree.

Sebelum kita membahas mengenai persyaratan dari Binary Search Tree kita harus mengetahui terlebih dahulu, apa itu root, parent, dan child. Hal ini akan menjadi faktor yang sangat penting di dalam sebuah Binary Search Tree.

1. Root.

Root atau yang di dalam Bahasa Indonesia disebut akar merupakan induk pertama dari sebuah tree. Jadi node ini akan menjadi pintu masuk dalam sebuah tree, atau node yang paling pertama. Root ini sangat berguna dalam Binary Search Tree, karena dengan root kita bisa mengetahui peletakan dari suatu node yang trelah diberikan inputan.

1. Parent.

Parent seperti namanya sendiri parent atau yang di dalam Bahasa disebut sebagai orangtua adalah sebuah node yang memiliki anak. Dimana parent ini dapat membantu melihat letak dari node-node yang berada di bawahnya.

1. Child.

Child atau anak merupakan sebuah anakan dari node-node parent. Child ini sendiri berfungsi sebagai penggantian parent pada saat parent dari child tersebut dihapus.

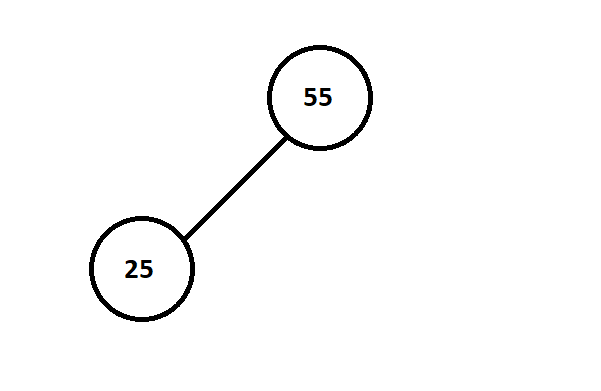
Setelah kita memahami konsep dari Root, Parent, dan Child kita memasuki kedalam persyaratan-persyaratan dari Binary Search Tree berdasarkan insertion dan deletion.

1. Insertion.

Pada bagian insertion yang mengimplementasikan Binary Search Tree akan ada persyarataan yaitu peletakan pada bagian kiri atau kanan dari parent. Peletakan ini berdasarkan lebih besar atau lebih kecilnya suatu nilai pada node yang di*insert* terhadap root ataupun parentnya.

Sebagai permisalan dalam kasus *insertion* adalah sebagai berikut. Misalkan program yang kita buat telah memiliki sebuah tree yang telah diketahui nilai rootnya adalah 55. Selanjutnya kita ingin memasukan nilai baru yaitu 25. Maka pada saat melakukan peletakan node yang berisi angka 25 adalah dengan cara seperti ini.

Kita melakukan perbandingan antara root dan node yang ingin dimasukan. Apakah 25 lebih besar atau lebih kecil dibandingkan 55. Tentu jawabannya adalah node 25 lebih kecil dari pada 55. Maka node yang berisi nilai 25 akan diletakan di bagian kiri dari node 55. Tampilan ilustrasi seperti gambar 1.1.



Gambar 1.1

1. Deletion.

Pengimplementasian deletion dalam *Binary Search Tree* ada terdapat dua cara. Jika node yang ingin dihapus terdapat anakan kiri dan anakan kanan. Sedangkan, jika node yang ingin dihapus hanya memiliki salah satu anak, maka dia akan menggunakan persyaratan berdasarkan anakan bagian kiri atau anakan bagian kanan. Tetapi, jika node yang ingin dihapus tidak memiliki anak sama sekali maka dia akan langsung dilakukan penghapusan.

Cara deletion yang tebagi menjadi dua diantaranya:

1. Mengambil anakan terbesar di bagian sebelah kiri (*max value*).

Pada cara pertama ini program akan mencari anakan yang paling besar disisi bagian kiri node yang ingin dihapus. Setelah menemukan node yang paling besar pada bagian kiri dari node yang ingin dihapus. Node yang paling besar tersebut akan menggantikan node yang ingin dihapus.

1. Mengambil anakan terkecil di bagian sebelah kanan (*min value*).

Pada cara kedua ini program akan mencari anakan yang paling kecil dari sisi bagian kanan node yang ingin dihapus. Setelah program menemukan node yang paling kecil pada bagian kiri dari node yang ingin dihapus. Setelah menemukan node yang paling kecil pada bagian kanan dari node yang ingin dihapus. Node yang paling kecil tersebut akan menggantikan node yang ingin dihapus.

1. **Linked List.**

Dalam pembuatan program yang menggunakan topik *Binary Search Tree* tentunya tidak luput dengan penggunaan linked list. Hal ini dapat dilihat dalam pengubung atau edge dari tree yang terbentuk dari program adalah pengimplementasian dari linked list. Jadi linked list menjadi salah satu topik pendukung bagi pembuatan projek yang mengambil topik *Binary Search Tree* ini.

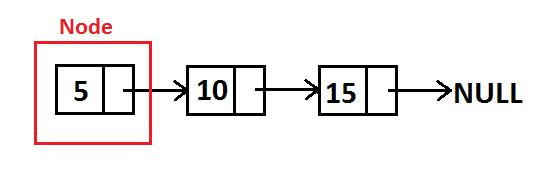
Linked list sendiri merupakan sautu struktur data linear yang bersifat dinamik. Berbeda dengan array yang sifatnya adalah statis (R, 2019). Hal ini dikarenakan dalam linked list kita tidak diminta untuk menentukan batas jumlah dari data yang bisa kita inputkan. Sedangkan bila kita menggunakan array kita diharuskan mendeklarasikan banyak data yang ingin kita simpan. Walaupun demikian terdapat beberapa kelebihan dan kekurangan dari linked list itu sendiri.

|  |  |
| --- | --- |
| Kelebihan Linked List | Kekurangan Linked List |
| 1. Pengalokasian memori yang dinamis membuat pengguna menjadi lebih bebas dalam memasukan data sebanyak-banyaknya. | 1. Pengimplementasian yang jauh lebih abstrak. |
| 1. Bersifat fleksibel, karena dapat memasukan dan menghapus data dibagian manapun. | 1. Waktu yang digunakan dalam mengakses data terbilang cukup memakan waktu. |
| 1. Jauh lebih hemat, karena kita tidak perlu menyiapkan memori yang berlebihan untuk penyedian tempat yang tidak diperlukan. |  |

Linked list sendiri terbagi kedalam berbagai macam jenis, diantaranya Single Linked List, Double Linked List, Circular Single Linked List dan Circular Double Linked List.

1. Single Linked List.

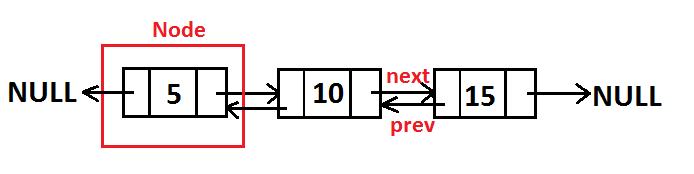
Single Linked List adalah suatu node yang memiliki beberapa jenis data yang setiap nodenya akan kita hubungkan dengan penghubung yang hanya menunjuk ke satu arah dan pada bagian akhir dari linked list akan diarahkan menuju NULL (Thareja, 2014). Pengimplementasian single linked list adalah seperti pada gambar 2.1.



Gambar 2.1 ( Sumber: <https://socs.binus.ac.id/files/2017/03/rini-1.jpg> )

1. Double Linked List.

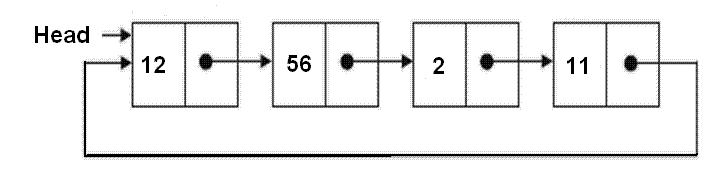
Double linked list adalah linked list yang setiap nodenya dihubungkan oleh dua penghubung, yang mana penghubung ini akan menunjuk sesudah dan sebelum node. Namun, node yang paling awal penghubung sebelumnya akan menunjuk ke NULL dan node yang paling terakhir penghubung setelahnya akan di arahkan menuju ke NULL juga. Contoh pengimplementasian dapat dilihat pada gambar 2.2.



Gambar 2.2 ( Sumber: <https://socs.binus.ac.id/files/2017/03/rini-8.jpg> )

1. Circular Single Linked List.

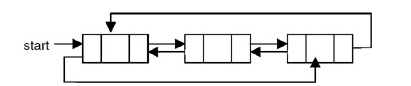
Circular Single Linked List adalah Single linked list yang pada bagian node paling terakhir akan dihubungkan menuju node awal, sehingga akan tercipta circle. Contoh pengimplementasian dari circular double linked list dapat dilihat pada gambar 2.3.



Gambar 2.2 ( Sumber: <https://www.geeksforgeeks.org/wp-content/uploads/cll_orig.gif> )

1. Circular Double Linked List.

Circular Double Linked List adalah Double Linked List yang pada bagian node paling awal penunjuk sebelumnya akan di arahkan menuju node paling terakhir dan pada bagian node paling terakhir penunjuk sesudahnya akan di arahkan menuju node paling awal. Contoh pengimplementasiannya dapat dilihat pada gambar 2.4.



Gambar 2.2 ( Sumber: <https://gmamaladze.files.wordpress.com/2011/08/doubly_linked_list2.png> )

1. **Fungsi dan Rekursi**
2. Fungsi

Fungsi dalam pemprograman dapat diartikan sebagai program yang peletakannya terpisah dari program utama namun masih berupa bagian dari program utama. Bukan hanya itu biasanya variabel yang digunakan dalam fungsi ini bersifat lokal (Salman et al., 2010). Fungsi sendiri dibuat agar mempermudah dalam proses pengcodingan program yang memiliki kepanjangan menengah hingga sangat panjang. Hal ini dikarenakan di dalam suatu fungsi kita hanya akan melakukan satu tugasan. Misalkan terdapat suatu program yang mampu menghitung luas permukaan dan volume suatu benda. Maka dapat dibuat sebuah fungsi yang bertujuan untuk menghitung luas permukaan dan satu lagi fungsi yang bertujuan untuk melakukan penghitungan volume.

Kelebihan fungsi dalam pengcodingan dapat kita lihat dari kinerjanya yang spesifik, sehingga apabila terjadi kesalahan atau terjadi error kita tidak perlu melakukan perbaikan codingan secara menyeluruh, namun kita dapat melakukan perbaikan pada fungsi yang bermasalah saja. Sehingga waktu pengerjaan dalam suatu program dapat terselesaikan dengan cepat. Selain itu juga dapat mencegah kerancuan pada bagian program utama pada menu lainnya, yang bisa saja membuat kita jauh mendapatkan masalah apabila terjadi error.

1. Rekursi

Rekursi sendiri merupakan sebuah fungsi. Namun rekursi merupakan seseuatu fungsi yang melakukan perulangan secara bersyarat, atau bisa dibilang fungsi yang dapat memanggil dirinya sendiri. Tentunya rekursi ini sangat berguna dalam penghitungan matematis seperti melakukan perhitungan pada bilangan factorial maupun fibonansi. Dimana di dalam perhitungannya kita perlu melakukan pemanggilan dari fungsi itu kembali untuk digunakan hingga syarat yang diberikan tidak terpenuhi (Deitel & Deitel, 2016).

1. **Operasi File**

Operasi file merupakan salah satu topik pembelajaran yang ada pada *algorithm and programming*. Operasi file sendiri merupakan sebuah cara dimana program dapat berinteraksi dengan sebuah file yang sudah kita buat atau yang ingin kita buat. Operasi file sendiri sangat berfungsi sebagai penyimpanan data. Hal ini dikarenakan dalam running program biasanya data yang dimasukan hanya bersifat sementara yaitu pada saat program dijalankan. Namun dengan melakukan penyimpanan data dengan cara operasi file data bisa tersimpan walaupun program yang kita jalankan telah kita tutup (Koding, 2016).

Ada banyak sekali mode-mode yang dapat digunakan dalam pengaplikasian operasi file ini. Tetapi terdapat tiga mode utama yang biasanya sering untuk digunakan.

|  |  |  |  |
| --- | --- | --- | --- |
| 1. | r (read) | : | berfungsi sebagai membaca file. Dengan catatan file harus sudah tersedia. |
| 2. | w (write) | : | berfungsi sebagai menulis file. File yang sudah ada akan terhapus dan tergantikan oleh file yang baru. |
| 3. | a (append) | : | Penambahan data pada file. Jika file belum tersedia maka akan dilakukan pembuatan file. |

Pembuatan file. Dalam operasi file pembuatan file juga menjadi kunci yang paling penting. Hal ini dikarenakan, tanpa adanya suatu file program tidak akan berjalan. Lalu untuk pembuatan filenya bisa dilakukan dengan melakukan pencodingan menuliskan fungsi **FILE** yang kemudian dilanjutkan oleh nama variabel file. Contoh:

**FILE \*fp;**

Dalam pembukaan file kita bisa melakukan dengan cara menyebutkan nama variabel lalu sama dengan **fopen** nama dari file yang ingin kita buka dan mode yang ingin digunakan. Seperti contoh di bawah ini.

**fp = fopen(“hadeh.txt”, “a”);**

Selain itu dalam melakukan pembacaan input dan pengeluaran output pada file juga ada beberapa perbedaan cara dengan pengeluaran output dan pembacaan input pada program biasa. Perbedaannya adalah dengan melakukan penambahan hurud “f” di depan scanf dan printf. Jadi ketika kita ingin melakukan pengeluaran output pada file maka kita akan menuliskan “fprintf” sedangkan untuk melakukan pembacan inputan dari file dapat dituliskan “fscanf”.

Setelah kita melakukan pembukaan terhadap suatu file dalam program jangan lupa untuk melakukan penutupan dari file tersebut. Hal ini harus dilakukan karena dalam pemprograman yang menggunakan file yang banyak ini mungkin saja bisa terjadi kebocoran atau bisa disebut error (Evan, 2009). Oleh Karena itu kita harus melakukan penutupan pada file yang kita buka. Cara melakukan penutupannya adalah dengan menyebutkan fungsi **fclose** lalu diikuti nama variabel dari file yang kita buka. Sebagai contoh seperti di bawah ini.

**fclose(fp);**

Studi kasus yang kami angkat sendiri sebagai bagian utamanya adalah pencarian rumah di dalam perumahan ‘AVA’ berdasarkan nomor telepon pemilik rumah. Berikut gambaran studi kasus yang lebih lengkapnya.

Developer perumahan ‘AVA’ merupakan seorang programmer handal beliau bernama Mr. Hadeh. Mr. Hadeh ingin membuat sebuah perumahan yang berbeda dari perumahan pada umumnya, sehingga dia ingin menerapkan perumahannya seperti *Binary Search Tree*. Dimana yang menjadi acuan terhadap persyaratan BST-nya adalah nomor rumah. Pada perumahan ‘AVA’ ini pemilik rumah bebas untuk menentukan nomor rumah yang mereka inginkan. Namun letak rumah yang mereka miliki harus sesuai dengan penerapan BST itu sendiri. Selain itu perumahan unik ini juga hanya menampung delapan buah rumah. Hal ini, untuk menciptakan kesan premium pada perumahan ini. Sering kali para pengantar barang maupun tamu dari pemilik rumah kebingungan letak rumah orang yang mereka tuju. Bukan hanya itu terkadang terdapat beberapa tamu yang tidak tahu alamat atau nomor rumah dari pemilik rumah yang ingin mereka tuju, mereka hanya memiliki nomor telepon pemilik rumah. Oleh karena itu untuk mempermudah pencariannya Mr. Hadeh ingin membuat sebuah program yang dapat membantu para tamu yang ingin berkunjung ke perumahan ‘AVA’ ini agar tidak sesat di jalan perumahan. Sebagai tim program development tim kami diminta oleh Mr. Hadeh untuk membuat suatu program yang dapat mempermudah para tamu dan pendataan kepemilikan rumah di perumahan ‘AVA’. Perogramnya sendiri terdapat beberapa menu, yaitu:

1. Pendaftaran Penghuni Baru di Peumahan ‘AVA’

Pada bagian Pendaftaran Penghuni Baru di Peumahan ‘AVA’ kita akan diminta memasukan input berupa Nomor rumah yang diinginkan, nama pemilik rumah, dan nomor telpon pemilik rumah yang terdiri dari 8 digit.

1. Perbaharuan Data Kepemilikan rumah di Perumahan ‘AVA’

Pada bagian Perbaharuan Data Kepemilikan rumah di Perumahan ‘AVA’ kita akan diminta untuk memasukan nomor rumah terlebih dahulu. Kemudian apabila nomor rumah terdapat pada tree maka akan diminta penggantian nama dan juga nomor telepon pemilik rumah.

1. Tampilan Data Kepemilikan Rumah di Perumahan ‘AVA’

Pada bagian menu Tampilan Data Kepemilikan Rumah di Perumahan ‘AVA’ pengguna program akan ditampilkan data dari semua pemilik rumah di perumahan ‘AVA’. Data yang ditampilkan terdiri dari Nomor Rumah, Nama Pemilik Rumah, dan Nomor Telepon Pemilik Rumah.

1. Pencarian Rumah Berdasarkan Nomor Telepon Pemilik Rumah

Pada bagian menu ke-4 ini pengguna program akan diminta menginputkan nomor telepon dari pemilik rumah. Apabila ditemukan maka akan tampil data dari pemilik rumah. Jika tidak maka aka nada pemberitahuan kesalahan.

1. Denah Perumahan ‘AVA’

Pada bagian Denah Perumah ‘AVA’ ini, kita diberikan output berupa gambaran dari perumahan ‘AVA’. Rumah digambarkan dalam bentukkan nomor rumah. Jadi nomor rumah menjadi wakil dari letak rumah tersebut.

1. Pencabutan Berkas Kepemilikan Rumah di Perumahan ‘AVA’

Pada bagian cabut berkas ini kita akan diminta untuk memasukan nomor rumah dari pemilik rumah. Kemudian apabila ditemukan datanya maka akan ditampilkan data dari kepemilikan rumah. Lalu berkas berhasil dicabut.

1. Menu Keluar

Pada bagian menu keluat ini program akan menutup program yang sedang berlangsung.

Pada program ini memuat:

1. Nomor Rumah (Ketentuannya)
2. Nama Pemilik Rumah
3. Nomor Telepon Pemilik rumah. (terdiri dari 8 digit angka)

Program tidak bisa langsung berhenti apabila pengguna program tidak memilih menu exit. Program juga akan meminta input ulang apabila inputan yang dilakukan oleh pengguna program tidak sesuai dengan apa yang diminta oleh program.

**DAFTAR PUSTAKA**

Deitel, P. J., & Deitel, H. M. (2016). *C How to Program with an introduction to C++ 8th Edition Global Edition*. http://www.pearsoned.co.uk/bookshop/detail.asp?item=100000000608187%0Ahttp://www.deitel.com/Books/C/CHowtoProgram8e/tabid/3664/Default.aspx%0Ahttps://bookshelf.vitalsource.com/#/books/9781292110981/

R, Z. R. (2019). *Linked List*.

Salman, W. P., Tisserand, O., & Toulout, B. (2010). How to Program. In *Forth*. https://doi.org/10.1007/978-1-349-07220-0\_3

Thareja, R. (2014). Data Structures Using C Second Edition. In *Journal of Chemical Information and Modeling* (Vol. 53, Issue 9).

Evan. (2009, December 12). *Tipe Data Lanjut, Pointer, & Operasi File : Penggunaan Fungsi fopen(), fclose(), fscan(), fprint(), fgets(), fputs(), fgetc(), fputc(), fread(), fwrite(), fseek(), & feof()*. Retrieved from Evan's Blog : Kuliah Informatika: https://kuliahinformatika.wordpress.com/2009/12/08/tipe-data-lanjut-pointer-operasi-file-penggunaan-fungsi-fopen-fclose-fscan-fprint-fgets-fputs-fgetc-fputc-fread-fwrite-fseek-feof/#:~:text=fclose()%20digunakan%20untuk%20menutup,file%20yang%20dapat%20beke

Koding, M. (2016, November 27). *Operasi File dalam Bahasa C*. Retrieved from mahirkoding: https://www.mahirkoding.com/operasi-file-dalam-bahasa-c/