Jurnal Multimedia dan Teknologi Informasi
Volume 02 Number 02, 01 Oktober 2020, pp. 46-55
e-ISSN : 2721-1800 pISSN : 2722-0907
doi.org/10.54209/jatilima. v2i2.40
Homepage :https://journal.cattleyadf.org/index.php/jatilima/index

Aplikasi pembelajaran struktur data dengan indeks model pohon bermain menggunakan metode pohon pencarian biner

Article Info

Article history:

Received 12 Augustus 2020
Revised 20 September 2020
Accepted 01 October 2020

Keywords:

Learning Applications, Data Structures, Play Tree Models, Binary Search Tree

ABSTRACT

Index is a method used to test the process of taking data records in the process of searching for records that are commonly used in databases. Making index simulations using the play tree algorithm is done by entering a number that is used as an index, the numbers entered by the user will be processed by the system to determine the position index in the simulation. A play tree is a binary search tree conforming with additional properties that newly accessed elements are quick to access again. The application of tree theory is very useful in the study of data structures where tree theory will get an alternative data storage structure that is relatively better and more efficient. All normal operations in a binary search tree are combined with one basic operation, which is called playing. Playing a tree for a particular element composes the tree so that the element is placed at the root of the tree. One way to do this is to first perform a standard binary tree search for the element in question, and then use tree rotation specifically to bring the element up.

Corresponding Author:
Sony Bahagia Sinaga

Amik Stiekom North Sumatra, Information Management, Medan, Indonesia
Email Address: soy Bahagia@gmail.com

1. Pendahuluan

Pohon bermain adalah pohon pencarian biner yang disesuaikan dengan properti tambahan yang elemen baru diakses dengan cepat untuk akses ulang [1]. Splay melakukan operasi dasar seperti penyisipan, pencarian, dan penghapusan dalam angka waktu yang diamortisasi (logaritma n). Untuk banyak urutan operasi non-acak, pohon yang diperluas berkinerja lebih baik daripada pohon pencarian lainnya, bahkan ketika pola urutan tertentu tidak diketahui. Pohon yang berkembang ditemukan oleh Daniel Dominic Sleator dan Robert Endre Tarjan pada tahun 1985. Semua operasi normal dalam pohon pencarian biner digabungkan dengan satu operasi dasar, yang disebut splaying. Memainkan pohon untuk elemen tertentu menyusun pohon sehingga elemen tersebut ditempatkan di akar pohon. Salah satu cara untuk melakukannya adalah dengan terlebih dahulu melakukan pencarian pohon biner standar untuk elemen tunggal atau gabungan sesuai kebutuhan. Jika ketiganya digunakan bersama-sama, konsekuensinya adalah jaringan akan jauh lebih aman daripada

doi.org/10.54209/jatilima. v2i2.40

hanya menggunakan satu proses keamanan, dan kemudian menggunakan rotasi pohon tertentu untuk memunculkan elemen. Alternatifnya, algoritma top-down dapat menggabungkan pencarian dan reorganisasi pohon ke dalam satu fase. Metodologi yang digunakan dalam penelitian ini adalah Research & Development dengan struktur data tutorial pembelajaran [2], [3]. Struktur data non-linier Jika ketiganya digunakan bersama-sama, konsekuensinya adalah jaringan akan jauh lebih aman daripada hanya menggunakan satu proses keamanan, dan kemudian menggunakan rotasi pohon tertentu untuk memunculkan elemen. Alternatifnya, algoritma top-down dapat menggabungkan pencarian dan reorganisasi pohon ke dalam satu fase. Metodologi yang digunakan dalam penelitian ini adalah Research & Development dengan struktur data tutorial pembelajaran [4], [5]. Struktur data non-linier merupakan sistem yang tidak konsisten [6]. Pembelajaran inovatif adalah pembelajaran yang memberikan pengetahuan yang menghasilkan peluang. Algoritma berdasarkan optimasi play tree ini diusulkan untuk mengurangi pengelompokan kata yang terakumulasi dengan merekam data [7], [8]. Hasil eksperimen menunjukkan bahwa metode yang diusulkan efektif dalam mengidentifikasi berbagai kategori untuk pencarian . Prinsip desain struktur data yang sangat rinci untuk secara otomatis mensintesis struktur data baru[9], [10] . Ada banyak kesulitan dalam proses belajar mengajar Algoritma dan Struktur Data . Pohon pencarian biner dinamis adalah kelas dasar struktur data kamus yang memprediksi bahwa pohon tampilan adalah bentuk pohon pencarian biner yang efisien secara universal, untuk setiap urutan akses[11], [12] .

2. Metode

Metode penelitian adalah suatu proses atau metode yang dipilih secara khusus untuk memecahkan masalah yang diajukan dalam suatu penelitian. Tahapan yang akan digunakan peneliti dapat dilihat pada gambar dibawah ini.

Gambar 1. Alur Penelitian

3. Hasil dan Pembahasan

Tujuan dari analisis adalah untuk memperbaiki berbagai fungsi dalam sistem agar lebih efisien, serta mengubah tujuan sistem untuk mencapai tujuan yang maksimal. Perangkat lunak simulasi indeks menggunakan algoritma splay tree ditunjukkan pada analisis berikut. Misalnya, jika bentuk infiks diketahui: (a + b) * c, maka proses konversi ke bentuk 4-tuple adalah sebagai berikut:

doi.org/10.54209/jatilima. v2i2.40

a. Cari tanda kurung buka terdalam '('.
b. Cari tanda kurung penutup pertama ')' dimulai dari posisi tanda kurung buka '('. Sub-ekspresi antara dua tanda kurung diambil dan diubah menjadi bentuk 4-tupel. Proses ini mendapatkan 4-tupel pertama:
   +, a, b, E (1)
   Bentuk sisipan diubah menjadi: E (1) * c
c. Ulangi proses pengecekan di atas sampai semua sub-ekspresi telah dikonversi atau hanya tersisa E dalam bentuk infiks.

Proses ini akan menghasilkan bentuk 4-tupel kedua:
*, E(1), c, E(2)
Bentuk sisipan diubah menjadi: E (2)

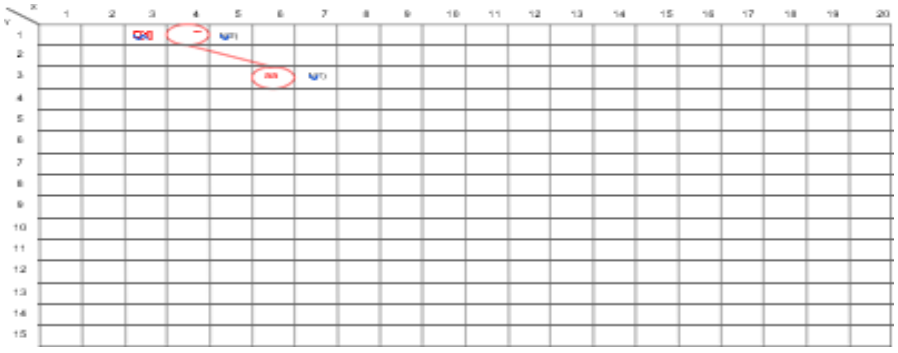
Jika ekspresi aritmatika diinput dalam bentuk postfix, maka proses konversi ke bentuk 4-tuple adalah sebagai berikut:

1. Proses pengecekan operand dan operator. Proses pengecekan ini sama dengan proses pada form prefix.
2. Proses pengecekan tanda kurung (delimiter). Proses pengecekan ini sama dengan proses pada form prefix.
3. Proses konversi ke bentuk 4-tupel. Untuk lebih jelasnya perhatikan contoh berikut ini: Misalkan kita mengetahui bentuk postfix: ab+c*, maka proses konversi ke bentuk 4-tuple adalah sebagai berikut:

a. Temukan operan dengan memulai dari posisi paling kiri. Proses ini akan menemukan karakter 'a'.
b. Jika karakter di sebelah kanan operan juga merupakan operan, maka karakter berikutnya harus berupa operator biner. Jika tidak, karakter di sebelah kanan operan adalah operator unary. Proses ini akan mendapatkan subekspresi ab + yang diubah menjadi bentuk 4-tupel menjadi: +, a, b, E (1) Bentuk awalan diubah menjadi: E (1) c *
c. Ulangi proses pengecekan di atas sampai semua sub-ekspresi telah dikonversi atau hanya tersisa E dalam bentuk postfix.

Proses ini akan menghasilkan bentuk 4-tupel kedua:
*, E(1), c, E(2)
Bentuk postfix diubah menjadi: E(2)

1. Proses menggambar struktur pohon E [1] = ~ aa.
2. Node yang terbentuk dan propertinya adalah sebagai berikut:



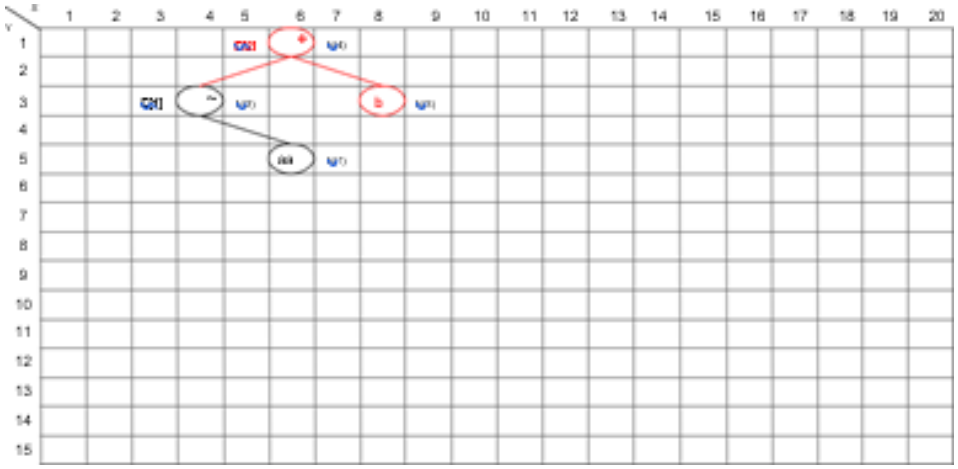
Gambar 2. Pohon Ekspresi E [1] = ~aa

Isi = '~'
Kiri = 0

doi.org/10.54209/jatilima. v2i2.40

Kanan = 1
Induk = 0

3. Proses menggambar struktur pohon E [2] = E [1] + b.



Gambar 3. Pohon Ekspresi E[2] = E[1] + b

Node yang terbentuk dan propertinya adalah sebagai berikut:

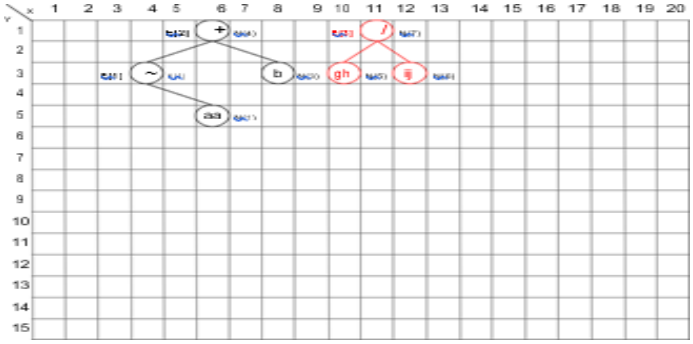
Node (1)
X = 6
Y = 5
Tag = 'E[2]'
Isi = 'aa'
Kiri = 0
Kanan = 0
Induk = 2

Node (2)
X = 4
Y = 3
Tag = 'E[2]'
Isi = '~'
Kiri = 0
Kanan = 1
Induk = 4

Node (3)
X = 8
Y = 3
Tag = 'E[2]'
Isi = 'b'
Kiri = 0
Kanan = 0
Induk = 4

Node (4)
X = 6
Y = 1
Tag = 'E[2]'
Isi = '+'
Kiri = 2
Kanan = 3
Induk = 0

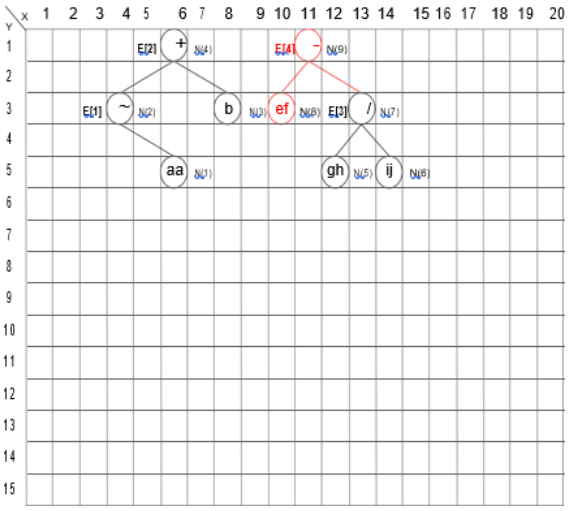
4. Proses menggambar struktur pohon E [3] = gh / ij.



Gambar 4. Pohon Ekspresi E [3] = gh / ij

Node yang terbentuk dan propertinya adalah sebagai berikut:

doi.org/10.54209/jatilima. v2i2.40



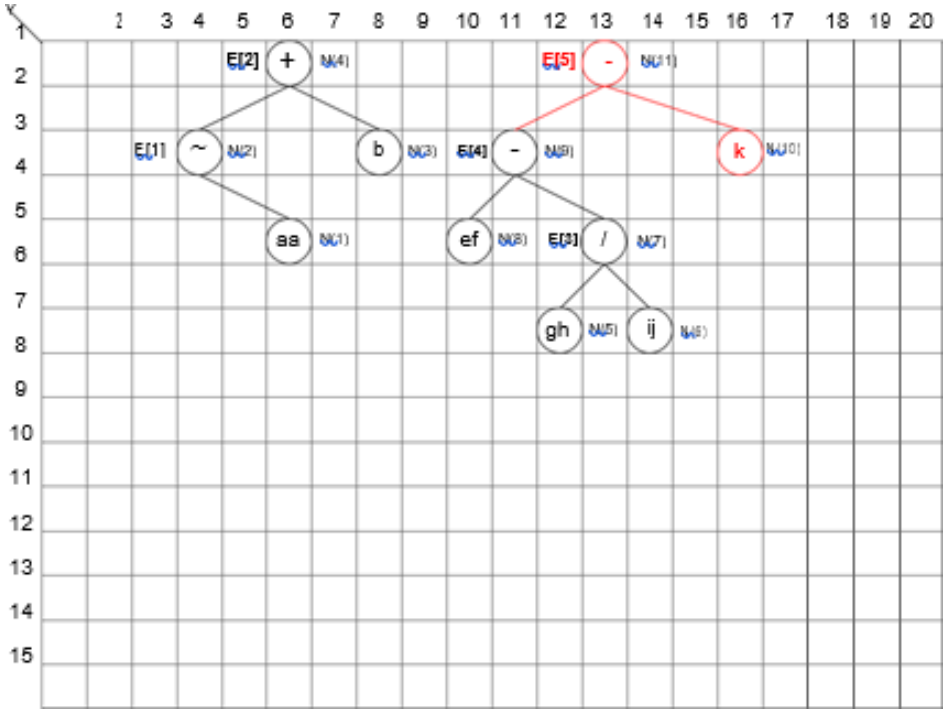
Gambar 5. Pohon Ekspresi E [4] = ef – E [3]

Node yang terbentuk dan propertinya adalah sebagai berikut:

Node (1)	Node (4)	Node (7)
X = 6	X = 6	X = 13
Y = 5	Y = 1	Y = 3
Tag = E [2]	Tag = E [2]	Tag = E [4]
Isi = 'aa'	Isi = '+'	Isi = '/'
Kiri = 0	Kiri = 2	Kiri = 5
Kanan = 0	Kanan = 3	Kanan = 6
Induk = 2	Induk = 0	Induk = 9
Node (2)	Node (5)	Node (8)
X = 4	X = 12	X = 10
Y = 3	Y = 5	Y = 3
Tag = E [2]	Tag = E [4]	Tag = E [4]
Isi = '~'	Isi = 'gh'	Isi = 'ef'
Kiri = 0	Kiri = 0	Kiri = 0
Kanan = 1	Kanan = 0	Kanan = 0
Induk = 4	Induk = 7	Induk = 9
Node (3)	Node (6)	Node (9)
X = 8	X=14	X=11
Y = 3	Y = 5	Y = 1
Tag = E [2]	Tag = E [4]	Tag = E [4]
Isi = 'b'	Isi = 'ij'	Isi = '-'
Kiri = 0	Kiri = 0	Kiri = 8
Kanan = 0	Kanan = 0	Kanan = 7
Induk = 4	Induk = 7	Induk = 0

6. Proses mendeskripsikan struktur pohon E [5] = E [4] - k.

doi.org/10.54209/jatilima. v2i2.40



Gambar 6. Pohon Ekspresi E [5] = E [4] – k

Node yang terbentuk dan propertinya adalah sebagai berikut:

Node (1)
X = 6
Y = 5
Tag = E[2]
Isi = 'aa'
Kiri = 0
Kanan = 0
Induk = 2

Node (2)
X = 4
Y = 3
Tag = E[2]
Isi = '~'
Kiri = 0
Kanan = 1
Induk = 4

Node (3)
X = 8
Y = 3
Tag = E[2]
Isi = 'b'
Kanan = 3
Induk = 0

Node (5)
X = 12
Y = 7
Tag = E [5]
Isi = 'gh'
Kiri = 0
Kanan = 0
Induk = 7

Node (6)
X=14
Y = 7
Tag = E [5]
Isi = 'ij'
Kiri = 0
Kanan = 0
Induk = 7

Node (7)
X = 13
Tag = E [5]
Isi = 'ef'
Kiri = 0
Kanan = 0
Induk = 9

Node (9)
X=11
Y = 3
Tag = E [5]
Isi = '-'
Kiri = 8
Kanan = 7
Induk = 11

Node (10)
X=16
Y = 3
Tag = E [5]
Isi = 'k'
Kiri = 0
Kanan = 0
Induk = 11

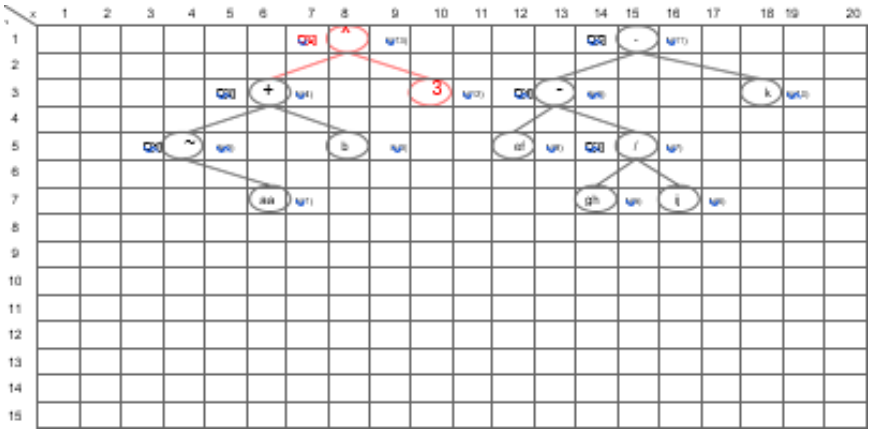
doi.org/10.54209/jatilima. v2i2.40

Kiri = 0
Kanan = 0
Induk = 4

Node (4)
X = 6
Y = 1
Tag = E[2]
Isi = '+'
Kiri = 2
Y = 5
Tag = E [5]
Isi = '/'
Kiri = 5
Kanan = 6
Induk = 9

Node (8)
X = 10
Y = 5
Node (11)
X = 13
Y = 1
Tag = E [5]
Isi = '-'
Kiri = 9
Kanan = 10
Induk = 0

7. Proses menggambar struktur pohon E [6] = E[2] ^ 3.



Gambar 7. Pohon Ekspresi E[6] = E[2]^3

Node yang terbentuk dan propertinya adalah sebagai berikut:

Node (1)
X = 6
Y = 7
Tag = E [6]
Isi = 'gh'
Kiri = 0
Kanan = 0
Induk = 7
Kanan = 7
Induk = 11
Node (10)
X=18
Y = 3
Tag = E [5]
Isi = 'k'
Kiri = 0
Kanan = 0
Induk = 11

Isi = 'aa'
Kiri = 0
Kanan = 0
Induk = 2

Node (2)
X = 4
Y = 5
Tag = E [6]
Isi = '~'
Kiri = 0
Kanan = 1
Induk = 4

Node (6)
X=16
Y = 7
Tag = E [5]
Isi = 'ij'
Kiri = 0
Kanan = 0
Induk = 7

Node (7)
X=15
Y = 5
Node (11)
X=15
Y = 1
Tag = E [5]
Isi = '-'
Kiri = 9

doi.org/10.54209/jatilima. v2i2.40

Node (3)
X = 8
Y = 5
Tag = E [6]
Isi = 'b'
Kiri = 0
Kanan = 0
Induk = 4

Node (4)
X = 6
Y = 3
Tag = E [6]
Isi = '+'
Kiri = 2
Kanan = 3
Induk = 13

Node (5)
X=14
Y = 7
Tag = E [5]
Isi = '/'
Kiri = 5
Kanan = 6
Induk = 9

Node (8)
X = 12
Y = 5
Tag = E [5]
Isi = 'ef'
Kiri = 0
Kanan = 0
Induk = 9

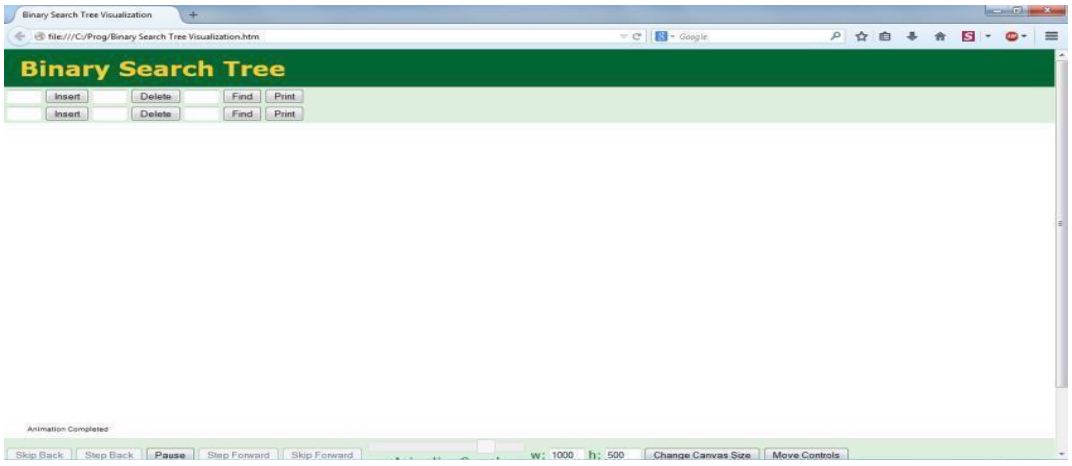
Node (9)
X = 13
Y = 3
Tag = E [5]
Isi = '-'
Kiri = 8
Kanan = 10
Induk = 0

Node (12)
X = 10
Y = 3
Tag = E [6]
Isi = '3'
Kiri = 0
Kanan = 0
Induk = 13

Node (13)
X = 8
Y = 1
Tag = E [6]
Isi = '^'
Kiri = 4
Kanan = 12
Induk = 0

Dalam setiap proses pengembangan sebuah aplikasi, ada dua komponen penting yang selalu dibutuhkan, yaitu perangkat keras dan perangkat lunak. Kedua komponen ini, satu dan lainnya saling mendukung sehingga membentuk suatu sistem. Berikut adalah hasil software pembelajaran yang telah dibangun.

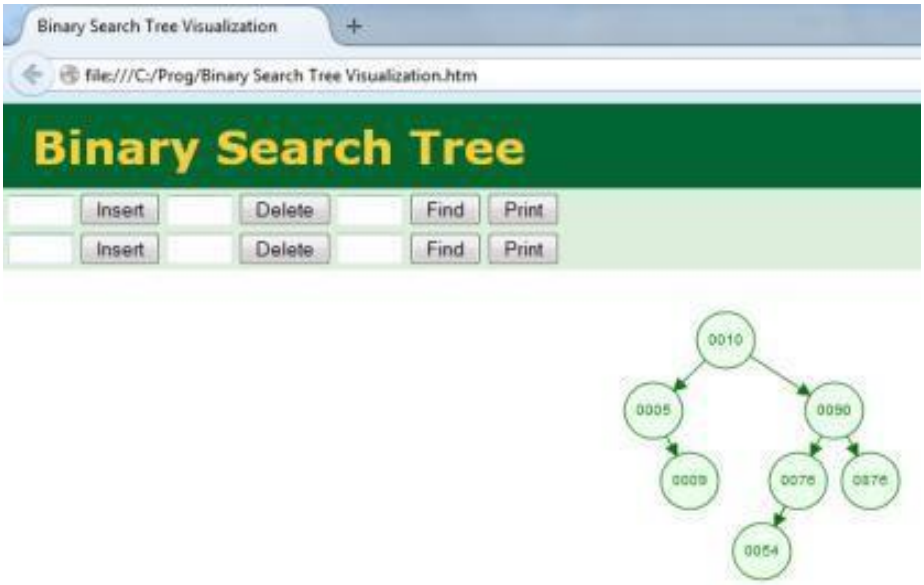
Gambar dibawah ini menjelaskan hasil implementasi pembelajaran binary search tree seperti gambar dibawah ini. Tampilan ini untuk mengimplementasikan pembelajaran struktur data.



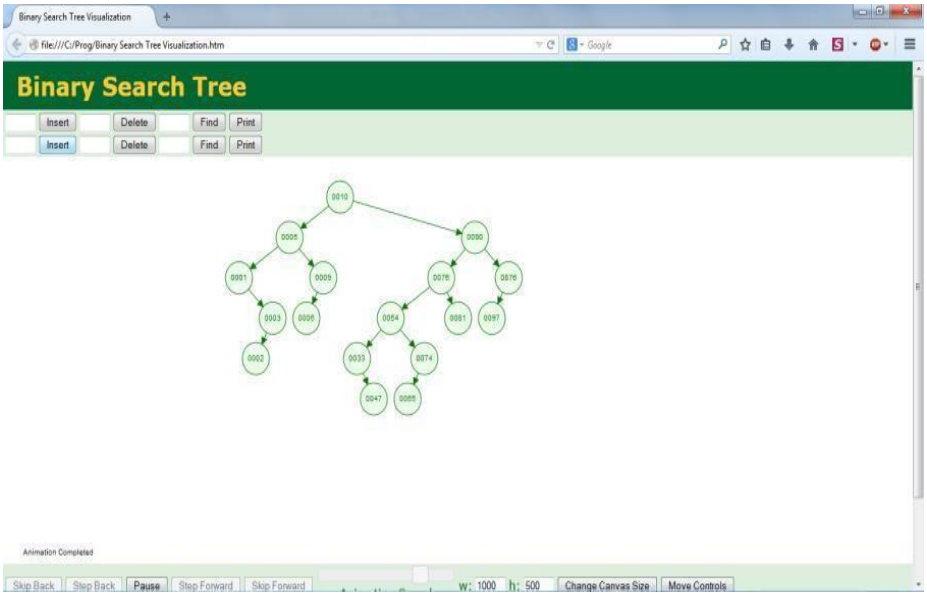
Gambar 8. Tampilan Menu

Tampilan di bawah ini menggambarkan hasil pengujian binary search tree pada struktur data seperti terlihat pada gambar di bawah ini. Hasil pencarian pada aplikasi ini akan memudahkan dalam mempelajari struktur data.

doi.org/10.54209/jatilima. v2i2.40



Gambar 9. Nomor Pengujian



Gambar 10. Kumpulan indeks

4. Kesimpulan

Berdasarkan hasil analisis yang penulis peroleh, penulis dapat menarik beberapa kesimpulan diantaranya: Penerapan metode Binary Search Tree dalam menentukan urutan indeks dapat berjalan dengan baik. Dengan menerapkan aplikasi sistem ini dapat memberikan pengetahuan tentang proses penyortiran. Dengan merancang antarmuka server yang mudah digunakan atau dipahami, pengguna dapat menggunakannya dengan mudah.

doi.org/10.54209/jatilima. v2i2.40

Referensi

[1] S. Kosasi, "Permainan Papan Strategi Menggunakan Algoritma Minimax," *STMIK Pontianak*, vol. 2, no. 372, 2014.

[2] G. Abdurrahman, "Clustering Data Kredit Bank Menggunakan Algoritma Agglomerative Hierarchical Clustering Average Linkage," *JUSTINDO (Jurnal Sist. dan Teknol. Inf. Indones.*, vol. 4, no. 1, 2019, doi: 10.32528/justindo.v4i1.2418.

[3] C. Baturu, "Brute Force Algorithm Implementation Of Dictionary Search," *J. Info Sains Inform. dan Sains*, vol. 10, no. 1, pp. 24–30, 2020.

[4] P. Pristiwanto, H. Sunandar, and B. Nadeak, "Analysis and Implementation of PlayFair Chipper Algorithm in Text Data Encoding Process," *J. Info Sains Inform. dan Sains*, vol. 10, no. 2, pp. 19–23, 2020.

[5] N. A. O. Saputri and M. P. Hannah, "Analisis Efektifitas Penggunaan Web-Based-Learning pada Matakuliah Praktikum Struktur Data," *JUST IT J. Sist. Informasi, Teknol. Inf. Dan Komput.*, vol. 8, no. 2, pp. 69–75, 2018.

[6] R. M. S. Anita Sindar and M. T. I. ST, "Struktur Data dan Algoritma."

[7] P. S. H. Hasugian and A. Simangunsong, "Implementation Of Least Significant Bit (LSB) Algorithm For Data Security In Digital Imagery," *J. Info Sains Inform. dan Sains*, vol. 10, no. 2, pp. 6–12, 2020.

[8] S. B. Sinaga, "Aplikasi Pembalajaran Struktur Data Dengan Index Model Splay Tree Dengan Menggunakan Metode Binary Search Treeeac," *J. Multimed. dan Teknol. Inf.*, vol. 2, no. 2, pp. 1–11, 2020.

[9] R. Harman, "Jurnal Ilmiah Informatika (JIF)."

[10] S. Lailiyah, K. Kusaeri, and W. Y. Rizki, "Identifikasi proses berpikir siswa dalam menyelesaikan masalah aljabar dengan menggunakan representasi graf," *J. Ris. Pendidik. Mat.*, vol. 7, no. 1, pp. 24–45, 2020.

[11] O. Suwanty, Suwanty; Pribadi, "Metode AVL Tree Untuk Penyeimbangan Tinggi Binary Tree," *J. TIMES*, vol. 4, no. 2, 2015.

[12] K. Khairunnisa and N. Wulan, "Perancangan Intelligent Tutoring System Sebagai Upaya Inovatif Pada Pembelajaran Algoritma dan Struktur Data," *Algoritm. J. ILMU Komput. DAN Inform.*, vol. 4, no. 2, 2020, doi: 10.30829/algoritma.v4i2.8515.