APLIKASI METODE
PENCARIAN LINIER, BINER, DAN INTERPOLASI

Hotlan Maroli Tua Lase[1], Jeremia Siregar[2]
Mahasiswa Prodi Teknik Informatika Fakultas Teknologi Industri, Institut Sains dan Teknologi TD.Pardede[1]
Dosen Prodi Teknik Informatika Fakultas Teknologi Industri, Institut Sains dan Teknologi TD.Pardede[2].
Email: [1]Hotlan.marolitua98@gmail.com. [2]jeremiasiregar@istp.ac.id.
JL. DR. TD.PARDEDE No.8 Medan 20153

ABSTRAK

Pencarian (searching) merupakan suatu pekerjaan yang sering dikerjakan dalam kehidupan sehari-hari. Ada kalanya pencarian dilakukan dengan tujuan hanya untuk mengetahui apakah data tersebut ada dalam sekumpulan data atau tidak, atau mungkin dilain waktu posisi dari data yang dicari tersebut dibutuhkan untuk keperluan tertentu, atau jika kemunculan data lebih dari satu kali maka semua posisi dan frekuensi kemunculannya ingin ditampilkan. Suatu yang dikerjaan bisa selesai dengan menggunakan algoritma yang tidak sama dengan kumpulan instruksi (*set of instructions*) yang berbeda dengan perbedaan waktu akses, efisiensi tempat, usaha dan sebagainya. Permasalahan dalam pencarian algoritma suatu data dapat diselesaikan dengan metoda pencarian linier, biner dan interpolasi. Dimana metoda linier (*Linier/Sequensial Search*) dapat dilakukan dengan pencarian tidak terurut (*random*), terurut menaik (*ascending*) dan terurut menurun (*descending*) sedangkan metoda biner dan interpolasi dapat dilakukan dengan pencarian terurut menaik (*ascending*) dan terurut menurun (*descending*). Perangkat lunak pembelajaran ini akan menampilkan tahapan-tahapan secara random, ascending, dan descending. Perangkat lunak pembelajaran juga menyediakan fasilitas ‘copy to clipboard’ untuk menyimpan algoritma pencarian dari metode linier, biner, dan interpolasi.

Kata kunci : *Pencarian, Linear, Biner, Interpolasi*

1. PENDAHULUAN

Pencarian adalah suatu pekerjaan yang sering dikerjakan dalam kehidupan sehari-hari. Ada kalanya kita mencari sesuatu dengan tujuan hanya untuk mengetahui apakah data tersebut ada dalam sekumpulan data atau tidak, sementara di lain waktu mungkin kita menginginkan posisi dari data yang dicari tersebut.

Dalam ilmu komputer terdapat bermacam-macam algoritma untuk metode pencarian yang pernah di pelajari adalah metode pencarian linear (*Linear* / *Sequential Search*), pencarian biner (*binary search)* dan pencarian interpolasi (*Iinterpolasi Search).* Masing-masing algoritma memiliki prasyarat dan cara serta waktu pelaksanaan yang berbeda. Pemilihan atas metode pencarian dilakukan berdasarkan keadaan dan keinginan pengguna metode yang biasanya tergantung pada data, jenis data dan struktur yang digunakan. Metode linier (Linear Search) merupakan teknik pencarian data secara urut di mulai dari depan ke belakang atau dari awal-akhir secara Sequen. Pencarian biner (Binary Search) adalah sebuah teknik untuk menemukan nilai tertentu dalam sebuah larik (array) , Prinsip pencarian Biner yaitu dengan membagi data atas dua bagian kemudian membandingkannya sedangkan pencarian interpolasi (Interpolation Search) adalah menentukan posisi yang diestimasi dari sisa rekaman yang belum diperiksa.

Algoritma adalah hal yang mendasar untuk komputer dalam proses informasi. Sebagaimana yang dikatakan oleh Thompson Susabda Ngoen [1] bahwa algortima merupakan langkah komputasi yang mengubah masukan menjadi keluaran yang benar.

Abu Abdullah Muhammad bin Musa Al-Khawarizmi hidup pada abad ke-9 dan berkebangsaan persia dan juga seorang ahli matematika yang menemukan pertama kali kata algoritma. Pada awalnya, kata *algorism* diartikan sebagai aturan-aturan untuk melakukan proses aritmatika menggunakan numerik Arab. Kata *algorism* diubah menjadi kata *algorithm* pada abad ke-18.

Ada Byron's notes on the analytical engine merupakan Penerapan pertama dari algoritma yang ditulis untuk sebuah komputer yang ditulis pada tahun 1842, dimana Ada Byron dianggap oleh kebanyakan orang sebagai *programmer* pertama di dunia. Walaupun sejak Charles Babbage tidak menyelesaikan analytical engine-nya, dan algoritma tidak pernah di implementasikan lagi.

Tujuan penelitian ini adalah untuk merancang suatu perangkat lunak yang mampu untuk melakukan analisa terhadap algoritma dari Metode pencarian linier, biner dan interpolasi untuk data terurut menaik (*ascending*) dan terurut menurun (*descending*) secara tahap demi tahap. Manfaat dari penelitian untuk Membantu pembelajaran algoritma dari Metode pencarian linier, biner dan interpolasi dan Sebagai fasilitas pendukung dalam proses belajar-mengajar

II. LANDASAN TEORI

II.1 Pengenalan Algoritma

Abu Abdullah Muhammad bin Musa Al-Khawarizmi hidup pada abad ke-9 dan berkebangsaan persia dan juga seorang ahli matematika yang menemukan pertama kali kata algoritma. Pada awalnya kata *algorims* di artikan sebagai aturan-aturan melakukan proses aritmatika dengan menggunakan numerik arab. Kata *algorism* diubah menjadi kata *algorithm* pada abad ke-18. Sekarang, penertiannya mencakup semua prosedur terhingga untuk menyelesaikan problea/melakukan pekerjaan.

Ada Byron's notes on the analytical engine' adalah penerapan pertama dari algoritma yang dituliskan sebuah komputer yang ditulis pada tahun 1842, dan Ada Byron di anggap programer pertama di dunia. Walaupun sejak Charles Babbage tidak menyelesaikan analytical engine-nya, dan algoritma tidak pernah di implementasikan lagi.

Algoritma bisa diimplementasikan dalam program komputer. Kesalahan dalam merancang algoritma untk menyelesaikan suatu problema dapat menyebabkan program gagal dam implementasinya. Konsep suatu algoritma sering diilustrasikan dengan mengambil contoh sebuah resep,.Walaupun banyak algoritma yang jahu lebih kompleks. Algoritma sering memiliki beberapa langkah perulangan (iterasi) atau memerlukan pengambilan keputusan seperti logika (*logic*) atau perbandingan

II.2 Pencarian

Pencarian (*searching*) merupakan suatu pekerjaan yang sering dikerjakan dalam kehidupan sehari – hari. Ada kalanya pencarian dilakukan dengan tujuan hanya untuk mengetahui apakah data tersebut ada dalam sekumpulan data atau tidak, atau mungkin di lain waktu posisi dari data yang dicari tersebut dibutuhkan untuk keperluan tertentu, atau jika kemunculan data lebih dari satu kali maka semua posisi dan frekuensi kemunculannnya ingin ditampilkan.

Pencarian paling sederhana dapat digambarkan sebagai berikut,
Misalkan suatu barisan data A[1] … A[n], maka yang menjadi perproblemaan adalah apakah X (sembarang data dengan tipe data sama dengan tipe data yang ada dalam barisan A dan biasanya di-*input* atau sudah diketahui terlebih dahulu) ada di antara A[1] … A[n] atau apakah X ada di dalam atau tidak dengan hasil *Boolean* benar / salah atau sukses / gagal.

macam-macam algoritma dalam ilmu komputer untuk metode pencarian (searching). Metode pencrian di bagi menjadi 2 bagian secara garis besar yaitu.

1. Metode pencarian data tanpa penempatan data seperti,
a) Metode pencarian Linier (linier/ Sequential search.
b) Metode Pencrian Biner (Binary search).
c) Metode Pencrian interpolasi (interpolation search).
2. Metode pencarian data dengan penempatan data seperti,
a) Metode pencarian langsung (direct search).
b) Metode Pencarian relatif (Hash search).

Masing – masing algoritma pencarian memiliki prasyarat dan cara serta waktu pelaksanaan yang berbeda – beda. Pemilihan atas Metode pencarian dilakukan berdasarkan keadaan dan keinginan pengguna Metode yang biasanya

tergantung pada jumlah data, jenis data data dan struktur data yang digunakan.

Sesuai dengan topik tugas akhir (skripsi) yang diambil, maka pembahasan Metode pencarian hanya terbatas pada Metode pencarian Linier (*Linear / Sequential Search*), Metode pencarian Biner (*Binary Search*) dan Metode pencarian Interpolasi (*Interpolation Search*).

II.2.1 Metode Pencarian Linier (Linear / Sequential Search)

Pencarian linier dilakukan dengan cara membandingkan data yang dicari (X) dengan data dalam barisan A[1]…A[n] dengan dimulai dari data elemen pertama pada barisan A. Jika perbandingan sama nilainya, maka pencarian dihentikan dan dinyatakan sukses. Sedangakn apabila perbandingan tidak bernilai sama maka;

1. Jika data akan dilanjutkan ke data selanjutnya.
2. Jika data terurut secara menaik (asecnding, maka pencarian hanya akan dilanjutkan ke data selanjutnya yang berada sisebelah kanan data yang sedang dibandingkan apabila data yang dicari (X) > data yangsedang di bandingkan sekarang.
3. Jika data terurut menurun (descending), maka pencarian hanya akan di lanjutkan ke data selanjutnya yang berada disebelah kanan data yang sedang dibandingkan apabila data yang dicari (X) < data yang sedang dibandingkan sekarang.

Jika syarat – syarat di atas dipenuhi, maka pencarian data akan dilakukan sampai data yang dicari (X) ditemukan sehingga pencarian dinyatakan sukses atau sampai elemen terakhir dari barisan A dan tidak ada elemen A yang sama dengan data yang dicari (X) sehingga pencarian dinyatakan gagal.

II.2.2 Metode Pencarian Biner (Binary Search)

Pencarian Biner hanya dapat dilakukan pada barisan bilangan yang telah diurutkan baik secara menaik (*ascending*) maupun menurun (*descending*). Pencarian Biner melakukan pencarian data X dalam barisan A[1] … A[n] dengan dimulai dari data tengah pada barisan A. Jika nilai data X sama dengan nilai data tengah barisan A, maka pencarian dihentikan dan dinyatakan sukses. Sedangkan jika tidak sama maka,

1. Untuk data yang diurutkan secara menaik (*ascending*), pencarian akan dilanjutkan ke ½ bagian kiri apabila nilai data X lebih kecil daripada nilai data tengah pada barisan A. Sedangkan apabila nilai data X lebih besar daripada nilai data tengah pada barisan A, maka pencarian akan dilanjutkan ke ½ bagian kanan.
2. Untuk data yang diurutkan secara menurun (*descending*), pencarian akan dilanjutkan ke ½ bagian kiri apabila nilai data X lebih besar daripada nilai data tengah pada barisan A. Sedangkan apabila nilai data X lebih kecil daripada nilai data tengah pada barisan A, maka pencarian akan dilanjutkan ½ bagian kiri.

Pencarian akan dihentikan dan dinyatakan gagal apabila ½ bagian kiri atau ½ bagian kanan berupa sebuah data tunggal dan data tersebut tidak sama dengan data X yang sedang dicari.

II.2.3 Metode Pencarian Interpolasi (Interpolation Search)

Pencarian Interpolasi mencari data dengan cara menebak (*guess*) posisi data yang dicari dengan menggunakan rumus tertentu. Pencarian interpolasi hanya dapat dilakukan pada barisan bilangan yang telah diurutkan secara menaik (Asecending) mauoun menurun (descending). Posisi yang diduga sebagai posisi data yang dicari (X) tersebut dapat dihitung dengan menggunakan rumus berikut,

Pos = ceiling [BB + ((X – A[BB]) / (A[BA] – A[BB])) * (BA – BB)]

Keterangan :

Pos = posisi yang diduga

BB = Batas Bawah; mula-mula sama dengan 1

BA = Batas Atas; mula-mula sama dengan jumlah data (n)

X = Data yang dicari

Ceiling = operasi matematika untuk pembulatan angka ke atas

Apabila data pada posisi tersebut sama dengan nilai data yang dicari (X) maka pencarian dihentikan dan dinyatakan sukses. Sedangkan apabila nilai data pada posisi yang diduga tidak sama dengan nilai data yang dicari (X), maka :

1. Untuk data yang diurutkan secara menaik (ascending), apabila nilai data yang dicari (X) < dibandingkan dengan nilai data pada posisi yang diduga, maka pencarian dilanjutkan dengan mengubah batas atas kawasan pencarian. Sedankan apabila nilai yang dicari (X) > dibandingkan dengan nilai data pada posisi yang diduga, maka pencrian dilanjutkan dengan mengubah batas bawah kawasan pencarian.
2. Untuk data yang diurutkan secara menurun (*descending*), apabila nilai data yang dicari (X) lebih besar dibandingkan dengan nilai data pada posisi yang diduga, maka pencarian dilanjutkan dengan mengubah batas atas kawasan pencarian. Sedangkan apabila nilai data yang di cari (X) < dibandingkan dengan nilai data pada posisi yang diduga, maka pencarian dilanjutkan dengan mengubah batas bawah kawasan pencarian.

Proses pencarian akan dihentikan dan dinyatakan gagal apabila terjadi kondisi – kondisi berikut ini,

a. Nilai Pos lebih kecil dari nilai batas bawah.
b. Nilai Pos lebih besar dari nilai batas atas.
c. Nilai posisi dugaan berturut – turut sama.

II.2.4 Metode Pencarian Relatif (Hash Search)

Metode pencarian Relatif (Hash Search) ini hampir mirip dengan Metode pencarian langsung (*Direct Search*), yaitu dengan menggunakan rumus tertentu baik pada saat penempatan maupun pencarian data. Pencarian Relatif (*Hash Search*) memiliki efisiensi penggunaan tempat yang lebih baik daripada pencarian langsung (*Direct Search*). Fungsi (I – 1) yang digunakan oleh Metode pencarian langsung (*Direct Search*) memiliki efisiensi penggunaan tempat yang buruk, sehingga Metode pencarian Relatif (*Hash Search*) memperbaikinya dengan menggunakan fungsi operasi modulo (mod). Fungsi operasi modulo(mod) sering disebut fungsi Hash dan tempat penampungan data di sebut tabel Hash. Fungsi *Hash* bukan merupakan fungsi satu – satu seperti fungsi bagi dari Metode pencarian Langsung (*Direct Search*) sehingga ada kemungkinan beberapa data memiliki hasil fungsi yang sama. Hal ini mengakibatkan terjadinya tebarakan (collision) pada saat penempatan data ke dalam tabel sehingga diperlukan stragtegi untuk mengatasi tabrakan (collision) ini. Strategi untuk mengatasi tabrakan(collision) ini ada bermacam-macam dan masing-masing memilki kelebihan dan kekurangannya masing-masing.

Fungsi hash ini menyebabkan data yang tersimpan dalam tabel hash memilki 2 jenis alamagt (address) yaitu:

1. *Home Address*, adalah lokasi (*address*) yang diperoleh dengan menggunakan Fungsi *Hash*.
2. *Real (Physical) Address*, adalah lokasi (*address*) dimana data tersimpan dalam tabel.

Metode pencarian Relatif (*Hash Search*) terdiri dari 2 macam yaitu,

1. Hash Tertutup(Closed Hash).
2. Hash terbuka (Open Hash)

III. HASIL DAN PEMBAHASAN

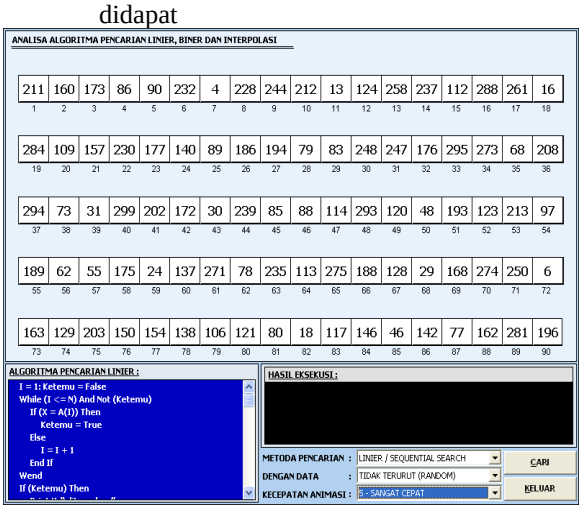
III.1 Algoritma Simulasi dan Analisa Pencarian

Pada tahap simulasi dan analisa, terdapat prosedur ‘SimulasiCari’, yang berfungsi untuk mensimulasikan tahapan – tahapan pencarian dan menampilkan hasil analisis dan prosedur kerja program dalam bentuk *report*. Prosedur S*imulasi Cari* terdiri atas 3 prosedur utama yaitu: *Linear Search, Binary Search Dan interpolation Search.* Prosedur Linear Search untuk mensimulasikan tahapan pencarian dengan algoritma pencarian linier dan menampilkan hasil analisis dan prosedur kerja program dalam bentuk report. Prosedir Bianry search untuk mensimulasikan tahapan pencarian dengan algoritma pencarian biner dan menampilkan hasil analisis dan prosedur kerja program dalam bentuk report. Prosedur interpolation search untuk mensimulasikan tahapan pencarian dengan algoritma pencarian interpolasi dan menampilkan hasil analisis dan prosedur kerja program dalam bentuk report.

Prosedur ‘SimulasiCari’ mempunyai 4 buah parameter yaitu, variabel *integer* pnAlgo (bernilai 0 untuk algoritma linier, 1 untuk algoritma biner, 2 untuk algoritma interpolasi), variabel *integer* pnUrut (bernilai 0 untuk data tidak terurut, 1 untuk data

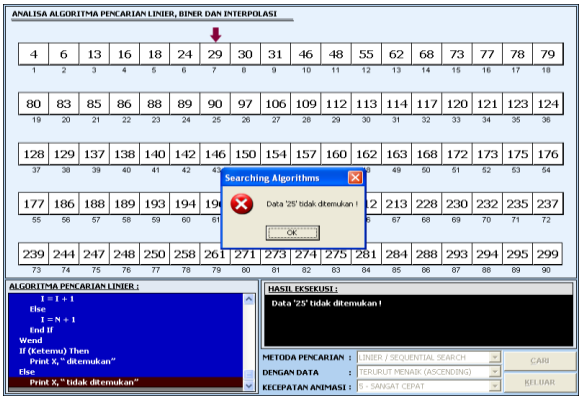
terurut menaik, 2 untuk data terurut menurun), variabel *integer* pnSpeed (sebagai ukuran kecepatan animasi) dan variabel *integer* pnSearch (sebagai variabel data yang ingin dicari).

- Proses penempatan barisan data pada tabel, didapat



Gambar 4.1 Tabel Penempatan Data

- Pencarian Data '25' dengan Metode pencarian linier (*sequential search*) dan data terurut menaik (*ascending*), didapat



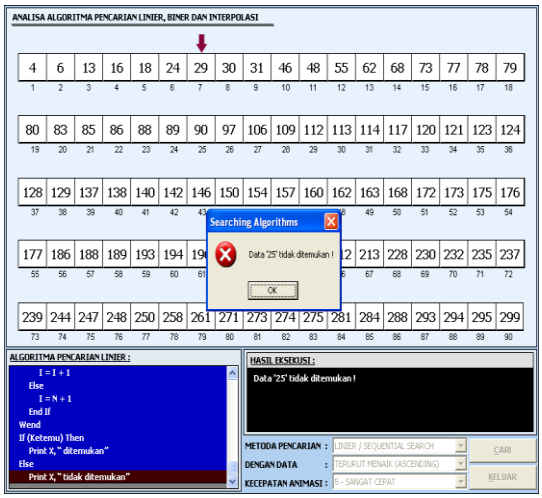
Hasil analisa algoritma yang dihasilkan program :

Data : 4, 6, 13, 16, 18, 24, 29, 30, 31, 46, 48, 55, 62, 68, 73, 77, 78, 79, 80, 83, 85, 86, 88, 89, 90, 97, 106, 109, 112, 113, 114, 117, 120, 121, 123, 124, 128, 129, 137, 138, 140, 142, 146, 150, 154, 157, 160, 162, 163, 168, 172, 173, 175, 176, 177, 186, 188, 189, 193, 194, 196, 202, 203, 208, 211, 212, 213, 228, 230, 232, 235, 237, 239, 244, 247, 248, 250, 258, 261, 271, 273, 274, 275, 281, 284, 288, 293, 294, 295, 299

X = Data yang dicari
I = Posisi data yang dibandingkan terhadap X
N = Jumlah data
Ketemu = variable boolean (apakah data sudah ditemukan atau belum)
A(I) = Data dengan posisi ke-I

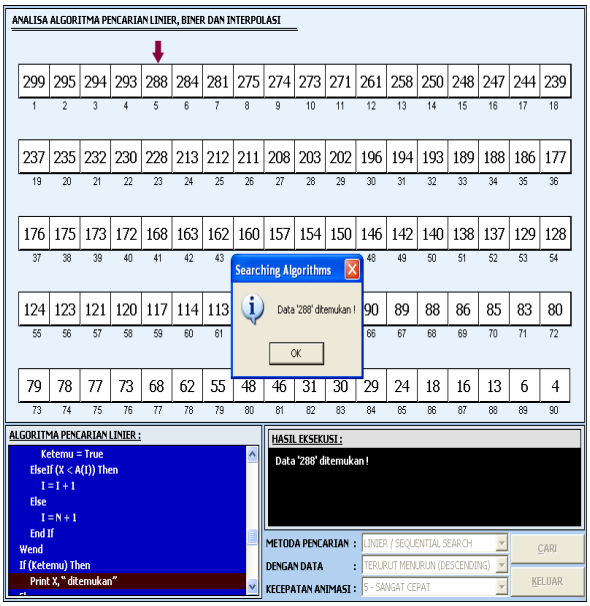
X = 25, I = 1, N = 90, Ketemu = False, A(1) = 4
While (1 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (25 = 4) -> False
(25 > 4) -> True
I = I + 1 = 1 + 1 = 2
X = 25, I = 2, N = 90, Ketemu = False, A(2) = 6
While (2 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (25 = 6) -> False
(25 > 6) -> True
I = I + 1 = 2 + 1 = 3
X = 25, I = 3, N = 90, Ketemu = False, A(3) = 13
While (3 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (25 = 13) -> False
(25 > 13) -> True
I = I + 1 = 3 + 1 = 4
X = 25, I = 4, N = 90, Ketemu = False, A(4) = 16
While (4 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (25 = 16) -> False
(25 > 16) -> True
I = I + 1 = 4 + 1 = 5
X = 25, I = 5, N = 90, Ketemu = False, A(5) = 18
While (5 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (25 = 18) -> False
(25 > 18) -> True
I = I + 1 = 5 + 1 = 6
X = 25, I = 6, N = 90, Ketemu = False, A(6) = 24
While (6 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (25 = 24) -> False
(25 > 24) -> True
I = I + 1 = 6 + 1 = 7
X = 25, I = 7, N = 90, Ketemu = False, A(7) = 29
While (7 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (25 = 29) -> False
(25 > 29) -> False
I = N + 1 = 90 + 1 = 91
X = 25, I = 91, N = 90, Ketemu = False
While (91 <= 90) And Not (False) -> False, Pencarian tidak dilanjutkan.

Ketemu = False
Data '25' tidak ditemukan !
Pencarian data '25' dilakukan hingga perbandingan sebanyak 7 kali.



Gambar 4.3 Hasil Pencarian Data '25' dengan Metode pencarian linier (sequential search) dan data terurut menaik (ascending)

- Pencarian Data '288' dengan Metode pencarian linier (*sequential search*) dan data terurut menurun (*descending*), didapat



Gambar 4.4 Hasil Pencarian Data '288' dengan Metode pencarian linier (sequential search) dan data terurut menurun (descending)

Hasil analisa algoritma yang dihasilkan program :

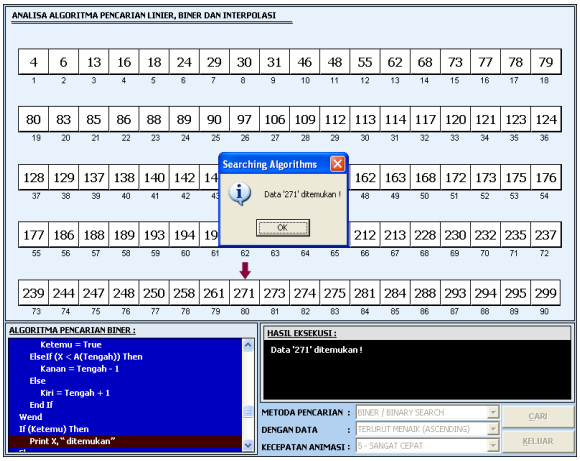
Data : 299, 295, 294, 293, 288, 284, 281, 275, 274, 273, 271, 261, 258, 250, 248, 247, 244, 239, 237, 235, 232, 230, 228, 213, 212, 211, 208, 203, 202, 196, 194, 193, 189, 188, 186, 177, 176, 175, 173, 172, 168, 163, 162, 160, 157, 154, 150, 146, 142, 140, 138, 137, 129, 128, 124, 123, 121, 120, 117, 114, 113, 112, 109, 106, 97, 90, 89, 88, 86, 85, 83, 80, 79, 78, 77, 73, 68, 62, 55, 48, 46, 31, 30, 29, 24, 18, 16, 13, 6, 4

X = Data yang dicari
I = Posisi data yang dibandingkan terhadap X
N = Jumlah data
Ketemu = variable boolean (apakah data sudah ditemukan atau belum)
A(I) = Data dengan posisi ke-I

X = 288, I = 1, N = 90, Ketemu = False, A(1) = 299
While (1 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (288 = 299) -> False
(288 < 299) -> True
I = I + 1 = 1 + 1 = 2
X = 288, I = 2, N = 90, Ketemu = False, A(2) = 295
While (2 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (288 = 295) -> False
(288 < 295) -> True
I = I + 1 = 2 + 1 = 3
X = 288, I = 3, N = 90, Ketemu = False, A(3) = 294
While (3 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (288 = 294) -> False
(288 < 294) -> True
I = I + 1 = 3 + 1 = 4
X = 288, I = 4, N = 90, Ketemu = False, A(4) = 293
While (4 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (288 = 293) -> False
(288 < 293) -> True
I = I + 1 = 4 + 1 = 5
X = 288, I = 5, N = 90, Ketemu = False, A(5) = 288
While (5 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
If (288 = 288) -> True
Ketemu = True
X = 288, I = 5, N = 90, Ketemu = True, A(5) = 288
While (5 <= 90) And Not (True) -> False, Pencarian tidak dilanjutkan.
Ketemu = True
Data '288' ditemukan !

Pencarian data '288' dilakukan hingga perbandingan sebanyak 5 kali.

- Pencarian Data ‘271’ dengan Metode pencarian biner (*binary search*) dan data terurut menaik (*ascending*), didapat



Gambar 4.5 Hasil Pencarian Data ‘271’ dengan Metode pencarian biner (binary search) dan data terurut menaik (ascending)

Hasil analisa algoritma yang dihasilkan program :

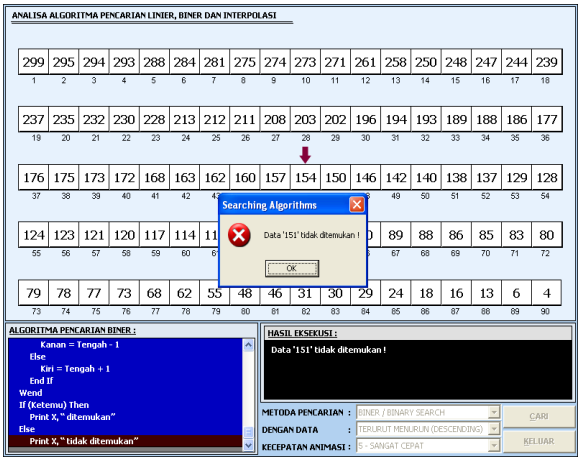
Data : 4, 6, 13, 16, 18, 24, 29, 30, 31, 46, 48, 55, 62, 68, 73, 77, 78, 79, 80, 83, 85, 86, 88, 89, 90, 97, 106, 109, 112, 113, 114, 117, 120, 121, 123, 124, 128, 129, 137, 138, 140, 142, 146, 150, 154, 157, 160, 162, 163, 168, 172, 173, 175, 176, 177, 186, 188, 189, 193, 194, 196, 202, 203, 208, 211, 212, 213, 228, 230, 232, 235, 237, 239, 244, 247, 248, 250, 258, 261, 271, 273, 274, 275, 281, 284, 288, 293, 294, 295, 299

----------X

= Data yang dicari
N = Jumlah data
Kiri = Nilai variable kiri dari algoritma biner
Kanan = Nilai variable kanan dari algoritma biner
Tengah = Nilai variable tengah dari algoritma biner
Ketemu = variable boolean (apakah data sudah ditemukan atau belum)
A(Tengah) = Data dengan posisi ke- <nilai tengah>

X = 271, Kiri = 1, Kanan = 90, Tengah = 0, Ketemu = False
While (1 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (1 + 90) \ 2 = 45
If (271 = 154) -> False
(271 < 154) -> False
Kiri = Tengah + 1 = 45 + 1 = 46
X = 271, Kiri = 46, Kanan = 90, Tengah = 45, Ketemu = False, A(45) = 154
While (46 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (46 + 90) \ 2 = 68
If (271 = 228) -> False
(271 < 228) -> False
Kiri = Tengah + 1 = 68 + 1 = 69
X = 271, Kiri = 69, Kanan = 90, Tengah = 68, Ketemu = False, A(68) = 228
While (69 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (69 + 90) \ 2 = 79
If (271 = 261) -> False
(271 < 261) -> False
Kiri = Tengah + 1 = 79 + 1 = 80
X = 271, Kiri = 80, Kanan = 90, Tengah = 79, Ketemu = False, A(79) = 261
While (80 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (80 + 90) \ 2 = 85
If (271 = 284) -> False
(271 < 284) -> True
Kanan = Tengah - 1 = 85 - 1 = 84
X = 271, Kiri = 80, Kanan = 84, Tengah = 85, Ketemu = False, A(85) = 284
While (80 <= 84) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (80 + 84) \ 2 = 82
If (271 = 274) -> False
(271 < 274) -> True
Kanan = Tengah - 1 = 82 - 1 = 81
X = 271, Kiri = 80, Kanan = 81, Tengah = 82, Ketemu = False, A(82) = 274
While (80 <= 81) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (80 + 81) \ 2 = 80
If (271 = 271) -> True
Ketemu = True
X = 271, Kiri = 80, Kanan = 81, Tengah = 80, Ketemu = True, A(80) = 271
While (80 <= 81) And Not (True) -> False, Pencarian tidak dilanjutkan.
Ketemu = True
Data '271' ditemukan !
Pencarian data '271' dilakukan hingga perbandingan sebanyak 6 kali.

- Pencarian Data ‘151’ dengan Metode pencarian biner (*binary search*) dan data terurut menurun (*descending*), didapat



Gambar 4.6 Hasil Pencarian Data ‘151’ dengan Metode pencarian biner (binary search) dan data terurut menurun (descending)

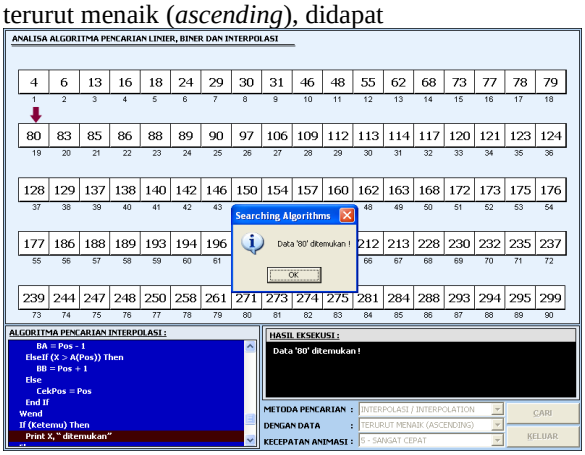
Hasil analisa algoritma yang dihasilkan program :

Data : 299, 295, 294, 293, 288, 284, 281, 275, 274, 273, 271, 261, 258, 250, 248, 247, 244, 239, 237, 235, 232, 230, 228, 213, 212, 211, 208, 203, 202, 196, 194, 193, 189, 188, 186, 177, 176, 175, 173, 172, 168, 163, 162, 160, 157, 154, 150, 146, 142, 140, 138, 137, 129, 128, 124, 123, 121, 120, 117, 114, 113, 112, 109, 106, 97, 90, 89, 88, 86, 85, 83, 80, 79, 78, 77, 73, 68, 62, 55, 48, 46, 31, 30, 29, 24, 18, 16, 13, 6, 4

X = Data yang dicari
N = Jumlah data
Kiri = Nilai variable kiri dari algoritma biner
Kanan = Nilai variable kanan dari algoritma biner
Tengah = Nilai variable tengah dari algoritma biner
Ketemu = variable boolean (apakah data sudah ditemukan atau belum)
A(Tengah) = Data dengan posisi ke- <nilai tengah>

X = 151, Kiri = 1, Kanan = 90, Tengah = 0, Ketemu = False
While (1 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (1 + 90) \ 2 = 45
If (151 = 157) -> False
(151 > 157) -> False
Kiri = Tengah + 1 = 45 + 1 = 46
X = 151, Kiri = 46, Kanan = 90, Tengah = 45, Ketemu = False, A(45) = 157
While (46 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (46 + 90) \ 2 = 68
If (151 = 88) -> False
(151 > 88) -> True
Kanan = Tengah - 1 = 68 - 1 = 67
X = 151, Kiri = 46, Kanan = 67, Tengah = 68, Ketemu = False, A(68) = 88
While (46 <= 67) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (46 + 67) \ 2 = 56
If (151 = 123) -> False
(151 > 123) -> True
Kanan = Tengah - 1 = 56 - 1 = 55
X = 151, Kiri = 46, Kanan = 55, Tengah = 56, Ketemu = False, A(56) = 123
While (46 <= 55) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (46 + 55) \ 2 = 50
If (151 = 140) -> False
(151 > 140) -> True
Kanan = Tengah - 1 = 50 - 1 = 49
X = 151, Kiri = 46, Kanan = 49, Tengah = 50, Ketemu = False, A(50) = 140
While (46 <= 49) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (46 + 49) \ 2 = 47
If (151 = 150) -> False
(151 > 150) -> True
Kanan = Tengah - 1 = 47 - 1 = 46
X = 151, Kiri = 46, Kanan = 46, Tengah = 47, Ketemu = False, A(47) = 150
While (46 <= 46) And Not (False) -> True, Pencarian dilanjutkan.
Tengah = (Kiri + Kanan) \ 2 = (46 + 46) \ 2 = 46
If (151 = 154) -> False
(151 > 154) -> False
Kiri = Tengah + 1 = 46 + 1 = 47
X = 151, Kiri = 47, Kanan = 46, Tengah = 46, Ketemu = False, A(46) = 154
While (47 <= 46) And Not (False) -> False, Pencarian tidak dilanjutkan.
Ketemu = False
Data '151' tidak ditemukan !
Pencarian data '151' dilakukan hingga perbandingan sebanyak 6 kali.

- Pencarian Data ‘80’ dengan Metode pencarian interpolasi (*interpolation search*) dan data terurut menaik (*ascending*), didapat



Gambar 4.7 Hasil Pencarian Data ‘80’ dengan Metode pencarian interpolasi (interpolation search) dan data terurut menaik (ascending)

Hasil analisa algoritma yang dihasilkan program :

Data : 4, 6, 13, 16, 18, 24, 29, 30, 31, 46, 48, 55, 62, 68, 73, 77, 78, 79, 80, 83, 85, 86, 88, 89, 90, 97, 106, 109, 112, 113, 114, 117, 120, 121, 123, 124, 128, 129, 137, 138, 140, 142, 146, 150, 154, 157, 160, 162, 163, 168, 172, 173, 175, 176, 177, 186, 188, 189, 193, 194, 196, 202, 203, 208, 211, 212, 213, 228, 230, 232, 235, 237, 239, 244, 247, 248, 250, 258, 261, 271, 273, 274, 275, 281, 284, 288, 293, 294, 295, 299

X = Data yang dicari
N = Jumlah data
BB = Nilai Batas Bawah
BA = Nilai Batas Atas
Pos = Posisi data yang dibandingkan
CekPos = Nilai Variable CekPos
Ketemu = variable boolean (apakah data sudah ditemukan atau belum)
A(Pos) = Data dengan posisi ke- <nilai pos>

X = 80, BB = 1, BA = 90, Pos = 0, CekPos = 0, Ketemu = False
While (1 <= 90) And Not (False) -> True, Pencarian dilanjutkan.
Temp = BB + ((X - A(BB)) / (A(BA) - A(BB))) * (BA - BB)
Temp = 1 + ((80 - 4) / (299 - 4)) * (90 - 1) = 23.928814
Pos = Int(Temp) = Int(23.928814) = 23
If (23.928814 - 23 > 0) -> True
Pos = Pos + 1 = 23 + 1 = 24
If (0 = 24) -> False
(80 = 89) -> False
(80 < 89) -> True
BA = Pos - 1 = 24 - 1 = 23
X = 80, BB = 1, BA = 23, Pos = 24, CekPos = 0, Ketemu = False, A(24) = 89
While (1 <= 23) And Not (False) -> True, Pencarian dilanjutkan.
Temp = BB + ((X - A(BB)) / (A(BA) - A(BB))) * (BA - BB)
Temp = 1 + ((80 - 4) / (88 - 4)) * (23 - 1) = 20.904762
Pos = Int(Temp) = Int(20.904762) = 20
If (20.904762 - 20 > 0) -> True
Pos = Pos + 1 = 20 + 1 = 21
If (0 = 21) -> False
(80 = 85) -> False
(80 < 85) -> True
BA = Pos - 1 = 21 - 1 = 20

X = 80, BB = 1, BA = 20, Pos = 21, CekPos = 0, Ketemu = False, A(21) = 85
While (1 <= 20) And Not (False) -> True, Pencarian dilanjutkan.
Temp = BB + ((X - A(BB)) / (A(BA) - A(BB))) * (BA - BB)
Temp = 1 + ((80 - 4) / (83 - 4)) * (20 - 1) = 19.278481
Pos = Int(Temp) = Int(19.278481) = 19
If (19.278481 - 19 > 0) -> True
Pos = Pos + 1 = 19 + 1 = 20
If (0 = 20) -> False
(80 = 83) -> False
(80 < 83) -> True
BA = Pos - 1 = 20 - 1 = 19

X = 80, BB = 1, BA = 19, Pos = 20, CekPos = 0, Ketemu = False, A(20) = 83
While (1 <= 19) And Not (False) -> True, Pencarian dilanjutkan.
Temp = BB + ((X - A(BB)) / (A(BA) - A(BB))) * (BA - BB)
Temp = 1 + ((80 - 4) / (80 - 4)) * (19 - 1) = 19
Pos = Int(Temp) = Int(19) = 19
If (19 - 19 > 0) -> False
If (0 = 19) -> False
(80 = 80) -> True
Ketemu = True

X = 80, BB = 1, BA = 19, Pos = 19, CekPos = 0, Ketemu = True, A(19) = 80
While (1 <= 19) And Not (True) -> False, Pencarian tidak dilanjutkan.
Ketemu = True
Data '80' ditemukan !

Pencarian data '80' dilakukan hingga perbandingan sebanyak 4 kali.

IV. KESIMPULAN DAN SARAN

IV.1 Kesimpulan

Setelah melalui proses penyelesaian tugas akhir (skripsi) yang berjudul "Perancangan Perangkat Lunak Untuk Menganalisa Algoritma Dari Metode Pencarian Linier, Biner, dan Interpolasi", penulis menarik kesimpulan sebagai berikut:

1. Membantu pembelajaran algoritma dari Metode pencarian linier, biner dan interpolasi yang dapat menampilkan cara kerja dari setiap algoritma secara tahap demi tahap.
2. Perangkat lunak pembelajaran ini menampilkan langkah-langkah penyesuaian algoritma untuk proses pencarian linier, biner dan interpolasi yang dapat dicopy hasil penyesuaian algoritmanya.

V.2 Saran

Penulis ingin memberikan beberapa saran yang mungkin berguna untuk pengembangan lebih lanjut pada Perancangan Perangkat Lunak Untuk Menganalisa Algoritma Dari Metode Pencarian Linier, Biner dan Interpolasi yaitu:

1. Perangkat lunak dapat dikembangkan agar dapat digabungkan dengan pembelajaran untuk metode pencarian linier, biner dan interpolasi yang lain.
2. Perangkat lunak dapat ditambahkan fasilitas *multimedia* agar lebih menarik

DAFTAR PUSTAKA

1. Ngoen, Thompson Susabda (2000) **Pengantar Algoritma**, Salemba Teknika, Jakarta.
2. Retno Hendrawati, Ir., M.T. (2000) **Logika Matematika**, Informatika, Bandung.
3. Dr. Suarga, M.Sc., M.Math., Ph.D. (2012) **Algoritma dan Pemrograman,** Yogyakara:C.V Andi Offset.
4. Lipschutz Seymor. (1997) **Discrete Mathematics**, Schaum Series, Mc.Graw-Hill, New York.
5. Donalt F. Starat, David F. Mc. Allister. (1977) **Discrete Mathematics in Computer Science**, Prentice Hall, Inc., USA.
6. Kenneth A.Ross, Charles R.B.Wright, (1999) **Discrete Mathematics**, Prentice Hall, Inc., USA.
7. Hadi, Rahadian. (2021) **Pemrograman Microsoft Visual Basic**, PT. Elex Media Komputindo, Jakarta.
8. Kurniawan, J. (2004) **Kriptografi, Keamanan Internet dan Jaringan Komunikasi**, Informatika, Bandung.
9. Stallings, W. (2003) ***Cryptography and Network Security*** *Third Edition, Prentice Hall*.
10. Putra, R. 2005 ***The Best Source Code Visual Basic***, PT.Elex Media Komputindo, Jakarta.