Penerapan Algoritma Binary Search dan String Matching Pada Pencarian Buku di Perpustakaan

Ferdy Santoso 13517116
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517116@std.stei.itb.ac.id

***Abstraksi*—Perpustakaan memiliki banyak sekali buku dan tidak mungkin jika kita memaksakan diri untuk mencari satu persatu buku yang kita ingin baca mulai dari depan perpustakaan, maka dari itu kita dapat menggunakan algoritma dalam mencari buku tersebut. Algoritma yang digunakan di makalah ini adalah algoritma searching binary search dan algoritma brute force string matching. Dua algoritma tersebut dapat mencari buku di perpustakaan lebih baik daripada mencari satu-satu (naif brute force searching).**

Kata kunci—binary, search, string, matching, algoritma, buku, perpustakaan

I. PENDAHULUAN

Perpustakaan adalah sarana bagi semua orang. Baik pagi siswa, mahasiswa, maupun orang pada umumnya, banyak sekali orang yang membutuhkan perpustakaan baik untuk keperluan akademik maupun non-akademik. Untuk mencari buku yang ada di perpustakaan umumnya disediakan sebuah atau lebih komputer di perpustakaan tersebut sehingga orang dapat mencari buku yang mereka ingin baca dan dapat menemukan tempat di mana buku tersebut berada.

Program yang terdapat di dalam komputer tersebut yang digunakan untuk melakukan pencarian pasti menggunakan algoritma di dalamnya untuk melakukan pencarian buku, baik judul buku maupun nama pengarang buku. Untuk mempercepat pencarian buku di program tersebut dibutuhkan algoritma yang cukup mangkus.

Algoritma yang dapat digunakan untuk melakukan pencarian adalah algoritma decrease and conquer. Algoritma decrease and conquer yang dapat digunakan di dalam pencarian buku di perpustakaan ini adalah algoritma binary search. Untuk melakukan pencocokan string judul buku atau nama pengarah buku dapat menggunakan algoritma string matching, yang akan saya gunakan di sini adalah algoritma brute force karena judul buku tidak terlalu panjang.

II. DASAR TEORI

1. Decrease and Conquer

Decrease and conquer adalah sebuah algoritma yang bekerja dengan cara mereduksi sebuah permasalahan menjadi berbagai sub-persoalan. Sub-persoalan yang diproses hanyalah satu sub-persoalan saja, berbeda dengan divide and conquer yang memroses semua sub-persoalan yang ada dan melakukan penggabungan.

Decrease and conquer memiliki 2 tahap :

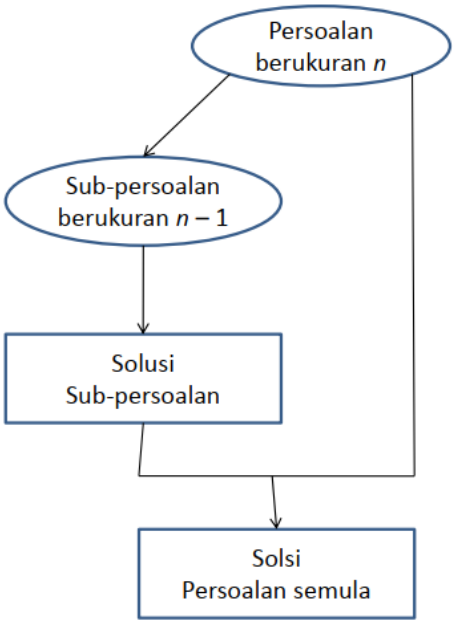
- Decrease : Tahap mereduksi persoalan menjadi beberapa sub-persoalan
- Conquer : Memproses satu sub-persoalan secara rekursif

Dalam algoritma decrease and conquer tidak ada tahap combine seperti di algoritma divide and conquer.

Algoritma decrease and conquer memiliki 3 varian berbeda, yakni :

1.1 Decrease by a constant

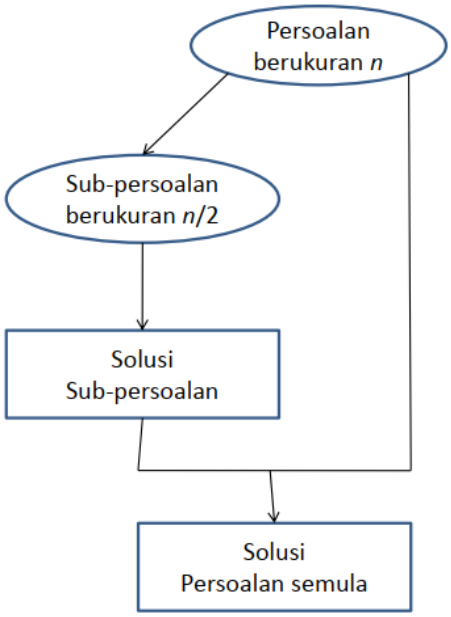
Ukuran instans persoalan direduksi sebesar konstanta yang sama setiap iterasi algoritma. Biasanya konstanta = 1. (Contoh : Persoalan perpangkatan a^n , Selection Sort).



Gambar 1. Skema algoritma decrease and conquer dengan decrease by a constant. (*sumber: Slide kuliah IF2211 Strategi Algoritma. Tanggal akses : 25 April 2019*)

1.2 Decrease by a constant factor

Ukuran instans persoalan direduksi sebesar faktor konstanta yang sama setiap iterasi algoritma. Biasanya faktor konstanta = 2. (Contoh : Binary Search, Interpolation Search, Mencari koin palsu).



Gambar 2. Skema algoritma decrease and conquer dengan decrease by a constant factor. (*sumber: Slide kuliah IF2211 Strategi Algoritma. Tanggal akses : 25 April 2019*)

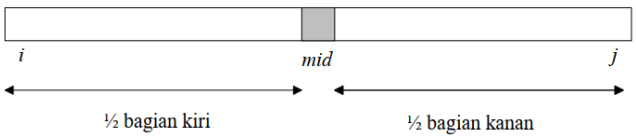
1.3 Decrease by a variable size

Ukuran instans persoalan direduksi bervariasi pada setiap iterasi algoritma. (Contoh : Menghitung median dan selection problem).

2. Binary Search

Binary Search adalah salah satu algoritma Decrease and Conquer varian kedua yakni decrease by a constant factor, dengan faktor konstanta = 2.

Kondisi awal : larik A sudah sudah terurut menaik K adalah nilai yang dicari



Gambar 3. Representasi larik dalam algoritma Binary Search. (*sumber: Slide kuliah IF2211 Strategi Algoritma. Tanggal akses : 25 April 2019*)

Jika elemen tengah (mid) != k, maka pencarian dilakukan hanya pada setengah bagian larik (kiri atau kanan).

Ukuran persoalan selalu berkurang setengah ukuran semula. Hanya setengah bagian yang diproses, setengah bagian lagi tidak.

```
procedure bin_search(input A : ArrayOfInteger;
          input i, j : integer; input K : integer; output idx : integer)
Deklarasi
     mid : integer

Algoritma:
    if i > j then   { ukuran larik sudah 0}
       idx ← -1     { k tidak ditemukan }
    else
       mid ← (i + j)/2
       if A(mid) = k then   { k ditemukan }
          idx ← mid         { indeks elemen larik yang bernilai = K }
       else
          if K > A(mid  then
             bin_search(A, mid + 1, j, K, idx)
          else
             bin_search(A, i, mid - 1 , K, idx)
          endif
       endif
 endif
```

Gambar 4. Skema umum algoritma Binary Search. (*sumber: Slide kuliah IF2211 Strategi Algoritma. Tanggal akses : 25 April 2019*)

T(n) untuk n = 0 adalah 0, dan T(n) untuk n > 0 adalah 1 + T(n/2).

Relasi rekursif tersebut diselesaikan sebagai berikut :

$T(n) = 1 + T(n/2)$

$= 1 + (1 + T(n/4)) = 2 + T(n/4)$

$= 2 + (1 + T(n/8)) = 3 + T(n/8)$

$= ... = j + T(n/2^j)$

Asumsi: $n = 2^j \rightarrow j = {}^2\log n$

$T(n) = {}^2\log n + T(1) = {}^2\log n + (1 + T(0)) = 1 + {}^2\log n = O({}^2\log n)$. Pada umumnya kompleksitas algoritma dari binary search direpresentasikan sebagai O(log n).

3. String Matching

Algoritma string matching adalah sebuah algoritma untuk pencarian substring di dalam sebuah string. Algoritma string matching ada yang bersifat exact match dan ada yang bersifat tidak exact match. Exact match berarti bahwa string yang ingin dicari di dalam sebuah string lain harus 100% sama, sementara algoritma yang tidak exact match berarti string yang ingin dicari di dalam sebuah string lain tidak harus 100% sama namun boleh hanya sebagai substring dari string lain tersebut.

Ada 2 komponen utama dalam algoritma string matching :

1. Teks : string yang di dalamnya akan dicari sebuah string pattern.
2. Pattern : string yang ingin dicari di dalam string text, dapat berupa substring dari string Teks.

Algoritma string matching yang dasar ada 3, yakni :

3.1 The Brute Force Algorithm

Algoritma brute-force atau yang biasa disebut sebagai algoritma yang naif memiliki cara kerja yang sederhana. Kita mengecek setiap posisi karakter di Text untuk mengetahui apakah ada Pattern yang mulai di posisi tersebut. Jika benar, keluarkanlah indexnya, jika salah tambah posisi karakter Text dengan 1, dan lakukan hal yang sama sampai ditemukan Pattern di dalam Text atau sampai Text sudah habis diperiksa.

Contoh :

Pattern : NOT

Text : NOBODY NOTICED HIM

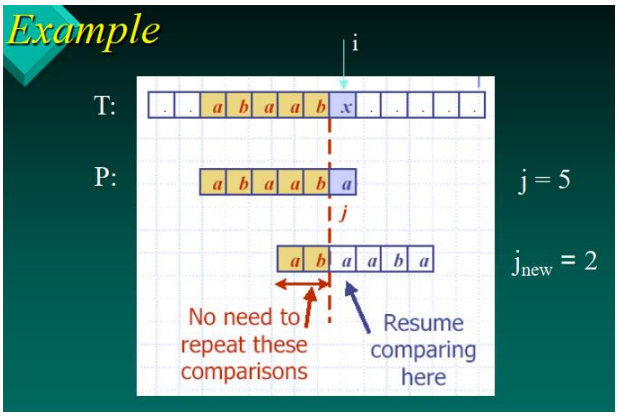
```
  NOBODY NOTICED HIM
1 NOT
2  NOT
3   NOT
4    NOT
5     NOT
6      NOT
7       NOT
8        NOT
```

Maka pattern NOT ditemukan di index ke-8 dari Text NOBODY NOTICED HIM.

3.2 The Knuth-Morris-Pratt Algorithm

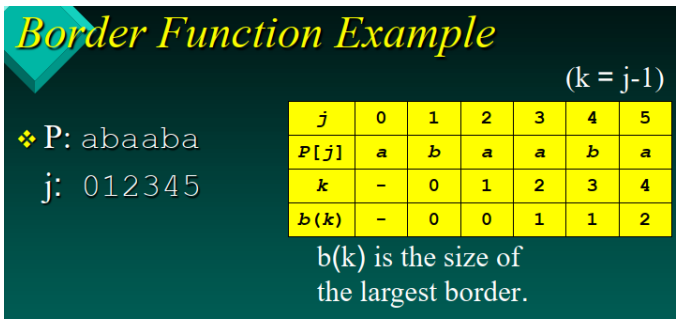
Algoritma Knuth-Morris-Pratt menggeser Pattern dari kiri ke kanan hampir sama seperti algoritma brute-force, namun algoritma Knuth-Morris-Pratt menggeser pattern dengan lebih cerdas dan tidak naif.

Teknik menggeser di algoritma Knuth-Morris-Pratt ini adalah sebagai berikut. Kita mencari apakah ada prefix di pattern yang sama dengan suffix di tempat berhenti pattern. Dengan teknik ini string pattern akan tergeser lebih cepat jika dibandingkan dengan teknik brute-force.



Gambar 5. Contoh penggunaan algoritma Knuth-Morris-Pratt. (*sumber: Slide kuliah IF2211 Strategi Algoritma. Tanggal akses : 25 April 2019*)

Untuk mengetahui letak di mana prefix di Pattern sama dengan suffix di posisi Pattern tertentu dibutuhkan pre-processing, dengan cara membuat larik yang menyatakan letak prefix yang sama dengan suffix dan berapa jumlahnya, fungsi membuat larik ini dinamakan Border Function.



j	0	1	2	3	4	5
P[j]	a	b	a	a	b	a
k	-	0	1	2	3	4
b(k)	-	0	0	1	1	2

Gambar 6. Contoh penggunaan Border Function. (*sumber: Slide kuliah IF2211 Strategi Algoritma. Tanggal akses : 25 April 2019*)

3.3 The Boyer-Moore Algorithm

Algoritma Boyer-Moore mengecek string Pattern apakah ada di string Text dengan urutan dari belakang ke depan. Ada 2 teknik yang digunakan di Boyer-Moore Algorithm ini :

1. The *looking-glass* Technique

Temukan Pattern di Text dengan cara menelusuri dari belakang ke depan di Pattern.

2. The *character-jump* Technique

Melakukan "lompat" ke karakter lain saat ada karakter di Pattern yang tidak cocok dengan yang ada di Text.

Algoritma yang saya gunakan untuk pengerjaan tugas ini adalah algoritma brute-force dengan tipe match Exact Match.

III. Implementasi dan Penjelasan Algoritma Program

Sebelum memulai algoritma pencarian, akan dilakukan inisialisasi array buku dan array pengarang buku terlebih dahulu. Hal ini dilakukan supaya ada hal yang dicari oleh algoritma. Untuk menginisialisasi array tersebut, saya telah mengisi 10 buku di array books, dan 4 pengarang di array authors. Saya mengalokasikan buku sebanyak 10 saja dan pengarang sebanyak 4 saja supaya algoritma saya dapat saya jelaskan secara cukup ringkas tahap per tahap di makalah ini.

Setelah dilakukan inisialisasi array buku dan array pengarang, yang dilakukan selanjutnya adalah melakukan sorting ke array buku dan array pengarang. Hal ini dilakukan karena algoritma binary search hanya dapat bekerja jika array yang dipakai telah terurut, dalam kasus ini array saya urutkan dalam urutan dari kecil ke besar.

Daftar tabel buku dan tabel pengarang dapat dilihat di bawah ini :

Index	Judul Buku	Letak Buku
0.	Algoritma Pemrograman Dasar	Lantai 1 Rak 2
1.	Game of Thrones	Lantai 3 Rak 2
2.	Hunger Games	Lantai 3 Rak 1
3.	Kalkulus Dasar	Lantai 1 Rak 1
4.	Kimia Anorganik	Lantai 4 Rak 5
5.	Kimia Organik	Lantai 2 Rak 1
6.	Kumpulan Puisi	Lantau 2 Rak 2
7.	Operating System Fundamentals	Lantai 2 Rak 6
8.	Strategi Algoritma	Lantai 1 Rak 5
9.	Struktur Data	Lantai 4 Rak 1

Tabel 1. Tabel Judul Buku dan Letak Buku yang sudah terurut berdasarkan Judul Buku dari alfabet terkecil ke terbesar.

Index	Nama Pengarang	Buku yang Telah Dikarang
0.	Ferdy Santoso	['Strategi Algoritma', 'Struktur Data', 'Algoritma Pemrograman Dasar']
1.	Jan Meyer	['Operating System Fundamentals', 'Kalkulus Dasar', 'Kumpulan Puisi']
2.	Nixon Andhika	['Hunger Games', 'Game of Thrones']
3.	Yoel Susanto	['Kimia Organik', 'Kimia Anorganik']

Tabel 2. Tabel Nama Pengarang dan Buku yang telah Dikarang yang sudah terurut berdasarkan Nama Pengarang dari alfabet terkecil ke terbesar.

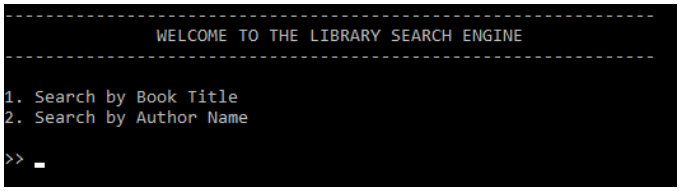
Secara garis besar, program saya bekerja seperti ini :

1. Pengguna memasukkan pilihan, apakah mereka ingin melakukan pencarian judul buku atau nama pengarang.
2. Pengguna memasukkan query sesuai dengan pilihan mereka.
3. Algoritma binary search dan string matching dijalankan.
4. Pengguna mendapatkan hasil dari query mereka.

Saya akan membahas lebih detail lagi mengenai implementasi dan cara kerja algoritma Binary Search dan String matching saya. Algoritmanya berjalan sebagai berikut :

1. Menjalankan algoritma bianry search.
2. Ambil nilai array di index tengah dari array buku/pengarang (index tengah adalah index awal + index terakhir dilakukan divisi dengan 2).
3. Lakukan string matching brute-force exact match antara nilai array di index tengah dengan string pattern.
4. Jika ditemukan nilainya sama, maka keluarkan posisi buku di perpustakaan (jika melakukan search buku), atau keluarkan buku apa saja yang pernah dikarang oleh suatu pengarang (jika melakukan search nama pengarang).
5. Jika tidak ditemukan, cek apakah string pattern bernilai lebih besar (secara alfabet) dari string di array index tengah, jika lebih besar ulangi langkah 2 dengan index awal yakni nilai tengah + 1 dan index akhir tetap, jika lebih kecil, ulangi langkah 2 dengan index awal tetap dan index akhir yakni index tengah – 1.
6. Jika sudah dilakukan pengulangan sampai index awal > index akhir (tidak ditemukan string pattern), maka keluarkan pesan bahwa buku/pengarang tidak ditemukan.

Saya telah membuat program pencarian buku/pengarang di perpustakaan sederhana, tampilan utamnya adalah sebagai berikut :



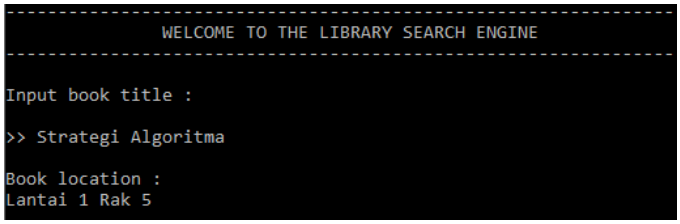
Gambar 7. Tampilan utama program pencarian buku sederhana

Jika pengguna memberi masukan 1, maka pengguna akan dialihkan ke menu pencarian buku, jika pengguna memberi masukan 2, maka pengguna akan dialihkan ke menu pencarian nama pengarang. Kedua menu menggunakan mekanisme binary search dan string matching yang sama, hanya output yang dihasilkan berbeda. Hasil output untuk menu pencarian buku adalah lokasi buku di perpustakaan (nomor lantai dan rak), sementara hasil output untuk menu pencarian nama pengarang adalah buku apa saja yang telah dikarang oleh pengarang tersebut.

Untuk lebih memperjelas algoritma saya, akan saya berikan uji kasus di bab selanjutnya.

IV. UJI KASUS

4.1 Uji Kasus 1



Gambar 8. Tampilan pencarian buku berdasarkan judul buku di program pencarian buku sederhana, dengan input “Strategi Algoritma”

Pengguna memilih pilihan 1 (pencarian berdasarkan judul buku). Langkah-langkah yang terjadi untuk mendapatkan jawaban tersebut adalah :

4.1.1 Pengguna memberikan input judul buku, dalam kasus ini adalah “Strategi Algoritma”.

4.1.2 Algoritma binary search dijalankan, cek apakah input user sama dengan buku yang ada di index tengah array ((0 + 9) div 2) (lihat Tabel 1) dengan menggunakan algoritma brute-force string matching.

4.1.3 String matching dilakukan antara judul buku “Strategi Algoritma” dan “Kimia Anorganik”. Huruf pertama sudah berbeda, karena exact match maka pencocokan string gagal.

4.1.4 Judul buku “Strategi Algoritma” jika dibandingkan dengan “Kimia Anorganik” bernilai lebih besar, maka kembali dilakukan binary search di sub-array bagian “atas” (bagian yang lebih besar) yakni index 5 - 9.

4.1.5 cek apakah input user sama dengan buku yang ada di index tengah array ((5 + 9) div 2) dengan menggunakan algoritma brute-force string matching.

4.1.6 String matching dilakukan antara judul buku “Strategi Algoritma” dan “Operating System Fundamentals”. Huruf pertama sudah berbeda, karena exact match maka pencocokan string gagal.

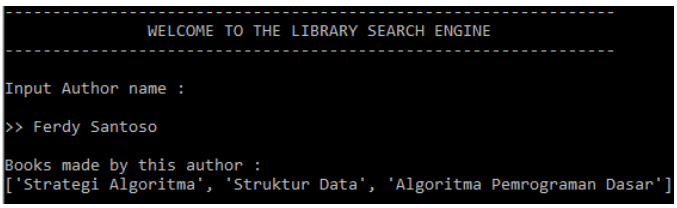
4.1.7 Judul buku “Strategi Algoritma” jika dibandingkan dengan “Operating System Fundamentals” bernilai lebih besar, maka kembali dilakukan binary search di sub-array bagian “atas” (bagian yang lebih besar) yakni index 8 - 9.

4.1.8 cek apakah input user sama dengan buku yang ada di index tengah array ((8 + 9) div 2) dengan menggunakan algoritma brute-force string matching.

4.1.9 String matching dilakukan antara judul buku “Strategi Algoritma” dan “Strategi Algoritma”. Dengan melakukan algoritma brute force exact match didapatkan hasil yang benar.

4.1.10 Lokasi buku “Strategi Algoritma” yakni “Lantai 1 Rak 5” (Lihat Tabel 1) ditampilkan ke user, algoritma binary search berhenti.

4.2 Uji Kasus 2



Gambar 9. Tampilan pencarian buku berdasarkan nama pengarang di program pencarian buku sederhana, dengan input “Ferdy Santoso”

Pengguna memilih pilihan 2 (pencarian berdasarkan nama pengarang). Langkah-langkah yang terjadi untuk mendapatkan jawaban tersebut adalah :

4.2.1 Pengguna memberikan input nama pengarang, dalam kasus ini adalah “Ferdy Santoso”.

4.2.2 Algoritma binary search dijalankan, cek apakah input user sama dengan pengarang yang ada di index tengah array ((0 + 3) div 2) (lihat Tabel 2) dengan menggunakan algoritma brute-force string matching.

4.2.3 String matching dilakukan antara nama pengarang “Ferdy Santoso” dan “Jan Meyer”. Huruf pertama sudah berbeda, karena exact match maka pencocokan string gagal.

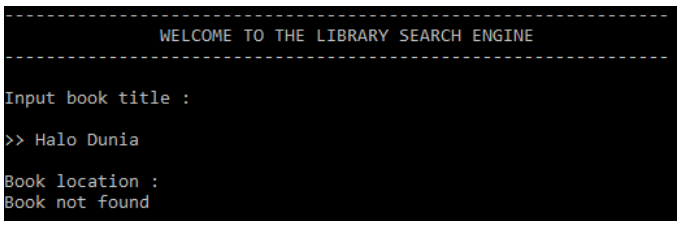
4.2.4 Nama pengarang “Ferdy Santoso” jika dibandingkan dengan “Jan Meyer” bernilai lebih kecil, maka kembali dilakukan binary search di sub-array bagian “bawah” (bagian yang lebih kecil) yakni index 0.

4.2.5 cek apakah input user sama dengan buku yang ada di index tengah array (0) dengan menggunakan algoritma brute-force string matching.

4.2.6 String matching dilakukan antara judul buku “Ferdy Santoso” dan “Ferdy Santoso”. Dengan melakukan algoritma brute force exact match didapatkan hasil yang benar.

4.2.7 Buku-buku hasil karangan “Ferdy Santoso” yakni ['Strategi Algoritma', 'Struktur Data', 'Algoritma Pemrograman Dasar'] (Lihat Tabel 2) ditampilkan ke user, algoritma binary search berhenti.

4.3 Uji Kasus 3



Gambar 10. Tampilan pencarian buku berdasarkan judul buku di program pencarian buku sederhana, dengan input "Halo Dunia"

Pengguna memilih pilihan 1 (pencarian berdasarkan judul buku). Langkah-langkah yang terjadi untuk mendapatkan jawaban tersebut adalah :

4.3.1 Pengguna memberikan input judul buku, dalam kasus ini adalah "Halo Dunia".

4.3.2 Algoritma binary search dijalankan, cek apakah input user sama dengan buku yang ada di index tengah array ((0 + 9) div 2) (lihat Tabel 1) dengan menggunakan algoritma brute-force string matching.

4.3.3 String matching dilakukan antara judul buku "Halo Dunia" dan "Kimia Anorganik". Huruf pertama sudah berbeda, karena exact match maka pencocokan string gagal.

4.3.4 Judul buku "Halo Dunia" jika dibandingkan dengan "Kimia Anorganik" bernilai lebih kecil, maka kembali dilakukan binary search di sub-array bagian "bawah" (bagian yang lebih kecil) yakni index 0 - 3.

4.3.5 cek apakah input user sama dengan buku yang ada di index tengah array ((0 + 3) div 2) dengan menggunakan algoritma brute-force string matching.

4.3.6 String matching dilakukan antara judul buku "Halo Dunia" dan "Game of Thrones". Huruf pertama sudah berbeda, karena exact match maka pencocokan string gagal.

4.3.7 Judul buku "Halo Dunia" jika dibandingkan dengan "Game of Thrones" bernilai lebih besar, maka kembali dilakukan binary search di sub-array bagian "atas" (bagian yang lebih besar) yakni index 2 - 3.

4.3.8 cek apakah input user sama dengan buku yang ada di index tengah array ((2 + 3) div 2) dengan menggunakan algoritma brute-force string matching.

4.3.9 String matching dilakukan antara judul buku "Halo Dunia" dan "Hunger Games". Huruf pertama benar, namun huruf kedua salah, karena exact match maka pencocokan string gagal.

4.3.10 Judul buku "Halo Dunia" jika dibandingkan dengan "Hunger Games" bernilai lebih kecil, maka kembali dilakukan binary search di sub-array bagian "bawah" (bagian yang lebih kecil) yakni index 2 sampai 1.

4.3.11 Dapat kita lihat dari langkah 4.3.10, kata-kata "index 2 sampai 1" menyatakan bahwa nilai j (range atas) sudah lebih kecil dari i (range bawah), maka algoritma sudah mencapai basis dan algoritma binary search berhenti, serta buku dinyatakan tidak ditemukan. Pesan "Book not Found" yang menyatakan bahwa buku tidak ditemukan ditampilkan ke user.

KESIMPULAN

Perpustakaan memiliki banyak sekali buku dan diperlukan algoritma yang mangkus untuk dapat mencari daftar buku dan lokasi buku yang tersedia di perpustakaan tersebut. Ada banyak sekali algoritma di dunia komputasi dan pemrograman, algoritma yang saya pilih adalah algoritma Binary Search dan Brute-force string matching.

Algoritma Binary Search dan Brute-force dapat bekerja dengan cukup efektif. Binary search sangat efektif dalam melakukan pencarian pada larik yang telah terurut, maka perlu diadakan pre-process terlebih dahulu sebelum melakukan pencarian.

REFERENSI

[1] Levitin, A. 2012. "The Design and Analysis of Algorithms 3rd edition". New Jersey: Pearson.

[2] Slide Kuliah IF2211 Strategi Algoritma. (http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/stima18-19.htm) Tanggal akses : 25 April 2019.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 26 April 2019

Ferdy Santoso
13517116