Pemanfaatan Algoritma *Backtracking* Dalam Pencarian Alamat Secara Manual

Serta Perbandingan Efektivitasnya Dengan Algoritma *Brute Force*

Christopher Chandrasaputra 13519074
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519074@std.stei.itb.ac.id

***Abstract*—Mencari alamat merupakan persoalan yang sering kali ditemukan di dunia nyata. Penggunaan teknologi membuat penggunanya dapat melakukan pencarian secara otomatis. Akan tetapi, tanpa penggunaan alat elektronik, pencarian alamat dapat dilakukan secara manual dengan menggunakan algoritma *backtracking*.**

***Keywords—backtracking*, pencarian, alamat**

I. PENDAHULUAN

Pada masa yang penuh teknologi ini, orang dapat mencari alamat menggunakan alat yaitu sebuah GPS. Hal ini tentu saja memudahkan pencarian alamat. Hanya dengan menuliskan informasi lokasi lalu melakukan pencarian rute, didapat lokasi dari alamat yang dimaksud secara otomatis. Akan tetapi, penggunaan teknologi ini secara berlebih dapat memanjakan penggunanya dalam mencari lokasi. Jika terjadi masalah pada teknologi ini, pengguna yang selalu mencari lokasi dengan cara tersebut akan mengalami kesulitan dalam mencari suatu lokasi secara manual tanpa menggunakan alat.

Pada makalah ini akan dibahas cara untuk mencari alamat dengan memanfaatkan algoritma *backtracking*. Hal ini ditujukan untuk keadaan yang mengharuskan pencarian alamat secara manual tanpa menggunakan alat elektronik. Selain itu, akan dibahas alasan penggunaan algoritma *backtracking* pada pencarian alamat dan akan dilakukan juga pembandingan dengan menggunakan algoritma lain dalam pencarian alamat yaitu algoritma *brute force*.

II. LANDASAN TEORI

A. Brute Force

Algoritma *brute force* merupakan algoritma yang sangat sederhana. Hal ini merupakan sederhana karena cara kerja algoritma *brute force* memiliki cara yang jelas dan tidak rumit. Karena cara kerjanya yang jelas dan tidak rumit, algoritma ini memiliki karakteristik yaitu tidak cerdas. Algoritma *brute force* dapat disebut juga algoritma naif.

Berikut adalah kelebihan dan kekurangan yang dimiliki dari algoritma *brute force*.

Kelebihan :

- Dapat diaplikasikan ke berbagai macam persoalan karena algoritma ini tidak memiliki batasan tipe persoalan
- Sederhana karena bentuknya yang tidak rumit sehingga memiliki cara yang jelas
- Pasti dapat menghasilkan solusi karena dilakukan pencocokan untuk semua kemungkinan yang ada
- Mengasilkan algoritma baku untuk beberapa kasus persoalan

Kekurangan :

- Tidak mangkus karena pencocokan solusi persoalan dilakukan untuk semua kemungkinan yang ada sehingga algoritma ini lambat
- Didasarkan pada kekuatan sistem komputer yang artinya algoritma ini akan lebih kuat seiring meningkatnya kekuatan sistem komputer
- Tidak kreatif karena penggunaan cara yang sederhana tanpa pembatasan solusi

Kelebihan dan kekurangan algoritma ini menjadikan algoritma *brute force* sebuah algoritma yang cocok untuk mecari solusi dari persoalan dengan skala yang kecil. Selain itu, algoritma *brute force* juga dapat dijadikan landasan untuk pembandingan algoritma lainnya.

Penggunaan brute force dapat menggunakan teknik *exhaustive search*. Teknik ini digunakan unuk mencari solusi dari persoalan kombinatorik. Cara kerja dari teknik ini adalah dengan mengenumenarsikan seluruh kemungkinan yang ada dan diambil yang terbaik.

Exhaustive search dapat diperbaiki menggunakan sebuah teknik yaitu teknik heuristik. Penggunaan teknik heurisktik ini digunakan untuk mengeliminasi kemungkinan yang tidak mungkin terjadi. Hasil yang dihasilkan dari penggunaan teknik ini belum tentu menghasilkan hasil yang optimal.

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

Gambar 1. Pencocokan *string* menggunakan algoritma *brute force*
Sumber :
https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Brute-Force-(2021)-Bag1.pdf, diakses pada 10 Mei 2021 pukul 10.43 (GMT+7)

B. Backtracking

Algoritma *backtracking* merupakan perbaikan dari *exhautive search*. Perbaikan yang dilakukan pada teknik tersebut adalah tidak melanjutkan evaluasi untuk solusi yang tidak mengarah ke solusi terbaik.

Properti umum algoritma *backtracking* :

- Solusi Persoalan

  Vektor n-tuple merupakan bentuk dari solusi

  Bentuk : X = $(x_1,x_2,x_3,..,x_n)$

- Fungsi Pembangkit

  Predikat T() digunakan untuk membangkitkan nilai dari x_k,

  Bentuk : T(x[1], x[2], x[3],...,x[k-1])

- Fungsi Pembatas

  Predikat B() untuk menentukan untuk melanjutkan solusi atau dibuang

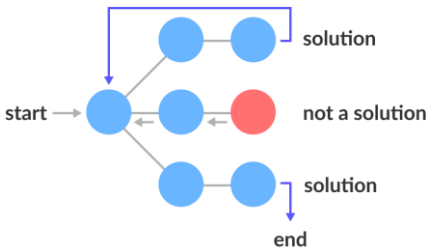
  Bentuk : B(x_1, x_2, x_3,.., x_k)

Cara kerja algoritma ini mengambil konsep dari teknik *exhaustive search* yang mengeksplorasi semua kemungkinan. Tetapi pada algoritma ini, fungsi pembatas akan membatasi eksplorasi yang dilakukan, jika fungsi pembatas menghasilkan nilai yang benar, maka akan dilakukan pembangkitan pada nilai yang akan ditelusuri selanjutnya. Hal ini mengakibatkan peningkatan efektivitas penyelesaian persoalan.

Langkah penyelesaian menggunakan algoritma *backtracking* adalah sebagai berikut :

1. Lakukan pengecekan pada nilai menggunakan fungsi pembatas.
2. a. Jika nilai memenuhi fungsi pembatas, bangkitkan semua nilai yang berasal dari nilai tersebut.

   b. Jika nilai tidak memenuhi fungsi pembatas, matikan nilai (tidak dilakukan pembangkitan pada nilai tersebut) dan kembali ke nilai yang membangkitkan.
3. Lakukan langkah 1 dan 2 pada setiap nilai yang dibangkitkan hingga tidak semua solusi sudah ditemukan (jika ada)



Gambar 2. Ilustrasi algoritma *backtracking*
Sumber : https://www.programiz.com/dsa/backtracking-algorithm, diakses pada 10 Mei 2021 pukul 11.16 (GMT+7)

Berikut adalah kelebihan dan kekurangan yang dimiliki oleh algoritma *backtracking*.

Kelebihan :

- Memiliki fungsi pembatas yang dapat menentukan lanjutan pengembangan solusi sehingga membuang solusi yang tidak sesuai dengan tujuan utama.
- Langkah demi langkah yang jelas sehingga mudah untuk dimengerti.
- Mudah untuk dicari kesalahan dalam pengaplikasiannya

Kekurangan :

- Merupakan algoritma yang hanya dapat dilakukan untuk persoalan tertentu
- Penyelesaian persoalan dapat memerlukan waktu yang cukup lama

III. Pencarian Alamat Menggunakan Algoritma *Backtracking*

3.1. Penggunaan Algoritma Backtracking

Algoritma *backtracking* akan digunakan untuk mencari sebuah alamat pada suatu lokasi. Pencarian alamat dilakukan dengan membandingkan nama dari alamat yang akan dicari.

Pada umumnya algoritma *backtracking* akan kembali mencari solusi lain yang memenuhi solusi persoalan. Akan tetapi, pada makalah ini akan dilakukan modifikasi pada algoritma backtracking sehingga pencarian akan selesai jika sudah ditemukan alamat yang bersesuaian.

Solusi persoalan algoritma *backtracking* pada persoalan ini memiliki banyak macam karena kemungkinan perbedaan format alamat yang dimliki daerah tertentu. Untuk persoalan ini akan digunakan dua macam solusi persoalan yaitu :

- Alamat = (Kode_Lokasi1, Kode_Lokasi2, Nomor_Lokasi)

Solusi persoalan ini digunakan untuk penomoran menggunakan blok rumah (contoh : Blok F1 nomor 7)

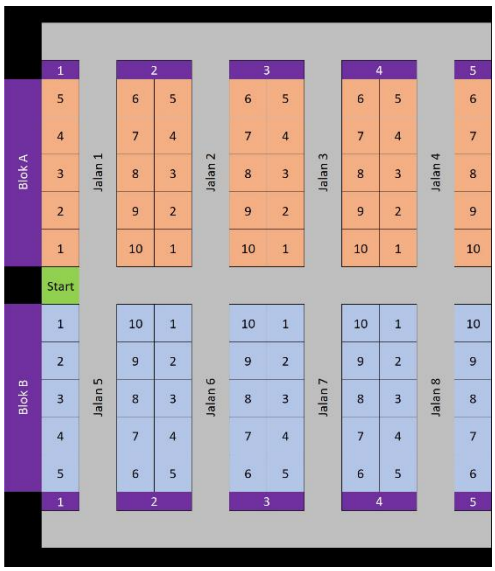
- Alamat = (Nama_Jalan, Nomor_jalan, Nomor_Lokasi)

  Solusi persoalan ini digunakan untuk penomoran menggunakan jalan (contoh : Jalan Strategi Algoritma V no. 2)

Fungsi pembatas yang digunakan adalah pengecekan kesesuaian nilai dari x_1 hingga x_i pada solusi persoalan. x_i adalah nilai ke-i pada tuple solusi persoalan. Jika kesesuaian bertambah, maka fungsi pembatas tidak akan mematikan percabangan.

3.1.1. Alamat Menggunakan Blok

Untuk melakukan pencarian alamat yang menggunakan blok, akan digunakan gambar dibawah ini.



Gambar 3. Peta Letak Bangunan Dengan Menggunakan Blok
Sumber : Ilustrasi Penulis

Pada gambar 3, penelusuran dimulai dari *start* yaitu lokasi yang ditandai dengan warna hijau. Penulisan alamat menggunakan penomoran bangunan yang berwarna oranye dan biru. Blok A dan Blok B menyatakan semua yang berada di baris horizontal yang sama merupakan blok tersebut. Penomoran blok dapat dilihat dari angka yang berada di atas atau dibawah blok bangunan tersebut. Sebagai contoh, kotak yang berada di atas tulisan *start* memiliki alamat Blok A1 nomor 1. Jalan pada kasus ini hanya akan digunakan untuk memperjelas penjelasan lokasi.

Pencarian alamat akan dilakukan menyesuaikan urutan alfabet dan nomor dari nomor terkecil dan dimulai dari jalan terkecil.

Langkah pencarian untuk kasus ini adalah sebagai berikut :

1. Bangkitkan seluruh kemungkinan kode pertama, dalam kasus ini akan dibangkitkan kode lokasi blok A dan blok B.
2. Bandingkan kode pertama dengan solusi.
3. a. Jika kode pertama tidak memenuhi solusi maka matikan percabangan tersebut.

   b. Jika kode pertama memenuhi solusi maka bangkitkan seluruh kode kedua yang berkaitan dengan kode pertama lokasi.
4. Bandingkan kode kedua dengan solusi.
5. a. Jika kode kedua tidak memenuhi solusi maka matikan percabangan tersebut.

   b. Jika kode kedua memenuhi solusi maka bangkitkan seluruh nomor lokasi yang berada pada kode lokasi yang bersangkutan.
6. Bandingkan nomor lokasi dengan solusi.
7. a. Jika nomor lokasi tidak memenuhi solusi maka matikan percabangan tersebut.

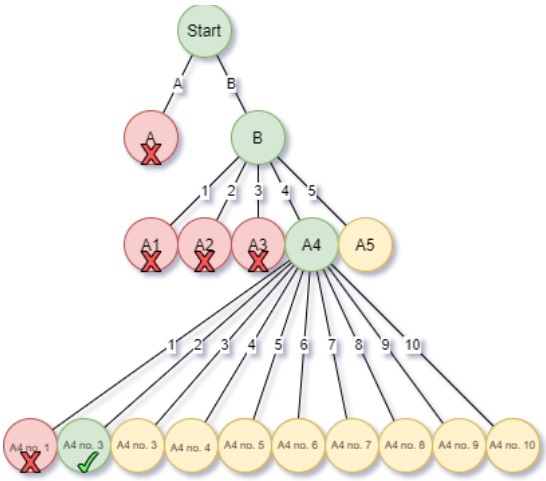
   b. Jika nomor lokasi sesuai dengan solusi persoalan, pencarian selesai.
8. Jika alamat lokasi tidak ditemukan, maka alamat lokasi yang diberikan memiliki kesalahan.

Sebagai uji coba, akan dicari 2 buah alamat menggunakan algoritma *backtracking* yaitu :

- Uji Pencarian 1.1 : Blok B4 nomor 2

  Solusi Persoalan = (B, 4, 2)

  1. Bangkitkan kode blok A dan blok B
  2. Kode blok A tidak memenuhi syarat, maka matikan percabangan ke blok A.
  3. Kode blok B memenuhi syarat, maka bangkitkan seluruh kode kedua dari blok B.
  4. Kode nomor blok 1, 2, dan 3 tidak memenuhi syarat, maka matikan percabangan ke nomor blok 1, 2, dan 3.
  5. Kode nomor blok 4 memenuhi syarat, maka bangkitkan seluruh nomor dari blok B4.
  6. Nomor bangunan 1 pada blok B2 tidak memnuhi syarat, maka matikan percabangan ke blok B4 nomor 1.
  7. Nomor bangunan 2 pada blok B2 memenuhi syarat, maka pencarian alamat selesai.

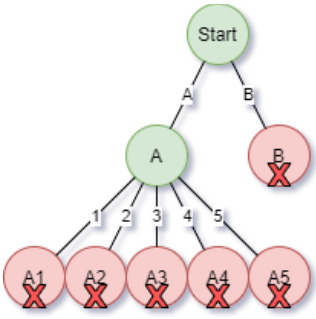


Gambar 4. Penggambaran hasil uji pencarian 1.1
Sumber : Ilustrasi Penulis

- Uji pencarian 1.2 : Blok A6 nomor 4

  Solusi Persoalan = (A, 6, 4)

  1. Bangkitkan kode blok A dan blok B
  2. Kode blok A memenuhi syarat, maka bangkitkan seluruh kode kedua dari blok A.
  3. Kode nomor blok 1, 2, 3, 4, dan 5 tidak memenuhi syarat, maka matikan percabangan ke nomor blok 1, 2, 3, 4, dan 5.
  4. Kembali ke hasil pembangkitan kode di awal yaitu kode blok B.
  5. Kode blok B tidak memenuhi solusi persoalan, maka matikan percabangan kode blok B.
  6. Tidak ada yang dapat dibandingkan dan di ekspansi lagi, maka pencarian selesai dengan tidak menemukan solusi.



Gambar 5. Penggambaran hasil uji pencarian 1.2
Sumber : Ilustrasi Penulis

3.1.2. Alamat Menggunakan Penomoran Bangunan Berdasarkan Jalan

Untuk melakukan pencarian alamat yang menggunakan penomoran bangundan berdasarkan jalan, akan digunakan gambar dibawah ini.

Gambar 6. Peta letak bangunan menggunakan penomoran jalan
Sumber : Ilustrasi Penulis

Pada gambar 6, penelusuran dimulai dari *start* yaitu lokasi yang ditandai dengan warna hijau. Penulisan alamat menggunakan penomoran bangunan yang berwarna biru. Jalan yang lebih besar akan terpecah menjadi jalan yang lebih kecil, pada contoh di gambar, Jalan Strategi dapat dipecah menjadi lebih kecil Jalan Strategi I dan Jalan Strategi II.

Pencarian alamat akan dilakukan dari jalan di utara yaitu dari Jalan Algoritma lalu Jalan Strategi. Selanjutnya dari Barat ke Timur, yaitu Jalan nomor I dan jalan nomor II. Pencarian Nomor bangunan akan dilakukan terurut dari nilai terkecil yaitu nol.

Langkah pencarian untuk kasus ini adalah sebagai berikut :

1. Bangkitkan seluruh kemungkinan nama jalan, dalam kasus ini akan dibangkitkan Jalan Algoritma dan Jalan Strategi.
2. Bandingkan nama jalan dengan solusi.
3. a. Jika nama jalan tidak memenuhi solusi maka matikan percabangan tersebut.

   b. Jika nama jalan memenuhi solusi maka bangkitkan seluruh nomor jalan yang berkaitan dengan nama jalan.
4. Bandingkan nomor jalan dengan solusi.
5. a. Jika nomor jalan tidak memenuhi solusi maka matikan percabangan tersebut.

   b. Jika nomor jalan memenuhi solusi maka bangkitkan seluruh nomor bangunan yang berada pada jalan yang bersangkutan.
6. Bandingkan nomor bangunan dengan solusi.
7. a. Jika nomor bangunan tidak memenuhi solusi maka matikan percabangan tersebut.

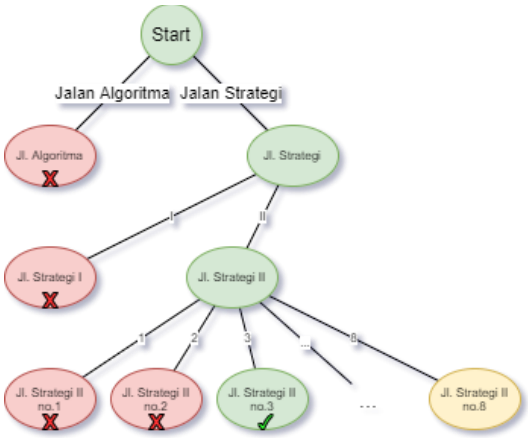
   b. Jika nomor bangunan sesuai dengan solusi persoalan, pencarian selesai.
8. Jika alamat lokasi tidak ditemukan, maka alamat lokasi yang diberikan memiliki kesalahan.

Sebagai uji coba, akan dicari 2 buah alamat menggunakan algoritma *backtracking* yaitu :

- Uji pencarian 2.1 : Jalan Strategi II nomor 3

  Solusi Persoalan = (Jalan Strategi, II, 3)

  1. Bangkitkan Jalan Strategi dan Jalan Algoritma.
  2. Jalan Algoritma tidak memenuhi syarat, maka matikan percabangan ke Jalan Algoritma.
  3. Jalan Strategi memenuhi syarat, maka bangkitkan seluruh nomor dari Jalan Strategi.
  4. Jalan Strategi I tidak memenuhi syarat, maka matikan percabangan ke Jalan Strategi I.
  5. Jalan Strategi II memenuhi syarat, maka bangkitkan seluruh nomor bangunan pada Jalan Strategi II.
  6. Nomor bangunan 1 dan 2 pada Jalan Strategi II tidak memnuhi syarat, maka matikan percabangan nomor bangunan 1 dan 2 pada Jalan Strategi II.
  7. Nomor bangunan 3 pada Jalan Strategi II memenuhi syarat, maka pencarian alamat selesai.

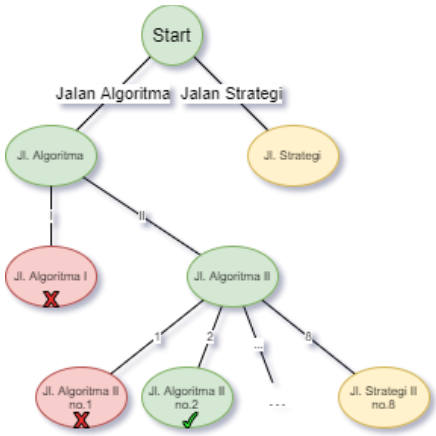


Gambar 7. Penggambaran hasil uji pencarian 2.1
Sumber : Ilustrasi Penulis

- Uji pencarian 2.2 : Jalan Algoritma II nomor 2

  Solusi Persoalan = (Jalan Algoritma, II, 2)

  1. Bangkitkan Jalan Strategi dan Jalan Algoritma.
  2. Jalan Algoritma memenuhi syarat, maka bangkitkan seluruh nomor dari Jalan Algoritma.
  3. Jalan Algoritma I tidak memenuhi syarat, maka matikan percabangan ke Jalan Algoritma I.
  4. Jalan Algoritma II memenuhi syarat, maka bangkitkan seluruh nomor bangunan pada Jalan Algoritma II.
  5. Nomor bangunan 1 pada Jalan Algoritma II tidak memnuhi syarat, maka matikan percabangan nomor bangunan 1 pada Jalan Algoritma II.
  6. Nomor bangunan 2 pada Jalan Algoritma II memenuhi syarat, maka pencarian alamat selesai.



Gambar 8. Penggambaran Hasil Uji Pencarian 2.2
Sumber : Ilustrasi Penulis

3.2. Perbandingan Dengan Algoritma *Brute Force*

Penggunaan algoritma *brute force* dalam mencari alamat dapat dilakukan dengan cara mencocokan setiap alamat pada alamat yang dicari.

Untuk mencari alamat yang menggunakan blok, hasil dari alamat yang dapat dihasilkan adalah Blok A1 nomor 1, Blok A 1 nomor 2, Blok A1 nomor 3, dan seterusnya hingga Blok B5 nomor 10. Total alamat yang dihasilkan berjumlah 80.

Untuk mencari alamat yang menggunakan penomoran jalan, hasil dari alamat yang dapat dihasilkan adalah Jalan Algoritma I nomor 1, Jalan Algoritma I nomor 2, Jalan Algoritma I nomor 3, dan seterusnya hingga Jalan Strategi II nomor 8. Total alamat yang dihasilkan berjumlah 32.

Jika akan dicari alamat yaitu Blok B5 nomor 9, maka diperlukan pencocokan solusi sebanyak 79 kali. Apabila alamat yang dicari adalah Jalan Strategi I nomor 3, maka diperlukan pencocokan solusi sebanyak 19 kali.

Sebagai pembanding hasil uji yang telah dilakukan sebelumnya berikut perbandingan efektivitas algoritma *brute force* dengan algoritma *backtracking*.

Test Uji	Jumlah Pencocokan	
	Brute Force	*Backtracking*
1.1	67	8
1.2	80	7
2.1	27	7
2.4	10	7

Tabel 1. Perbandingan jumlah pencocokan alamat menggunakan algoritma *brute force* dan algoritma *backtracking*

Dari jumlah pencocokan yang dilakukan oleh kedua algoritma, didapatkan bahwa penggunaan algoritma *backtracking* dalam mencari alamat lebih baik dibandingkan menggunakan algoritma *brute force*. Hal ini disebabkan oleh kemampuan algoritma *backtracking* yang dapat membedakan pola. Pola yang dimaksud pada hal ini adalah nama jalan atau kode pertama blok dan nomor jalan atau kode kedua blok. Pada algoritma *brute force* hanya dilakukan pencocokan nama secara keseluruhan tanpa memperhatikan pola penamaan yang terbentuk dari letak bangunan.

IV. Kesimpulan

Penggunaan algoritma *backtracking* dapat dilakukan dalam pencarian alamat tanpa menggunakan bantuan alat elektronik. Hal ini merupakan salah satu bukti bahwa pembelajaran strategi algoritma dapat diaplikasikan pada berbagai bidang. Pengaplikasian strategi yang didapat dari pelajaran strategi algoritma juga tidak terbatas pada perangkat elektronik.

Pada persoalan yang dibahas dalam makalah, meskipun algoritma *brute force* dapat menghasilkan hasil yang diinginkan, penggunaan algoritma *backtracking* memiliki efektivitas yang lebih baik.

Langkah sederhana dalam persoalan yang ada pada makalah ini adalah melakukan perbandingan pada setiap nilai pada alamat, dimulai dari yang paling umum yaitu jalan atau kode blok, kemudian nomor jalan atau kode blok, kemudian dilanjut dengan membandingkan nomor bangunan. Jika terdapat perbedaan saat pembandingan dari hal yang ter-umum maka ubah nilai yang umum tersebut.

V. Ucapan Terima Kasih

Penulis memberi ucapan syukur kepada Tuhan Yang Maha Esa oleh karena-Nya, penulis dapat menyelesaikan makalah ini. Penulis mengucapkan terima kasih kepada Dosen yang dengan senantiasa membimbing dan mengajarkan penulis dalam mata kuliah Strategi Algoritma, Dr. Nur Ulfa Maulidevi, ST., M.Sc. Penulis bersyukur dan juga ingin mengucapkan terima kasih kepada keluarga penulis yang telah memberikan dukungan kepada penulis. Tidak lupa penulis berterima kasih untuk para pembaca yang telah membaca makalah ini.

Penulis memohon maaf jika makalah ini memiliki banyak kekurangan dan jika terdapat kesalahan kata. Harapan penulis untuk makalah ini adalah ilmu yang didapat dari makalah ini dapat digunakan dengan baik dan dapat dikembangkan lebih baik lagi kedepannya.

Referensi

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Brute-Force-(2021)-Bag1.pdf, diakses pada 10 Mei 2021 pukul 10.29 (GMT+7)

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Brute-Force-(2021)-Bag2.pdf, diakses pada 10 Mei 2021 pukul 10.30 (GMT+7)

[3] https://www.geeksforgeeks.org/brute-force-approach-and-its-pros-and-cons/, diakses pada 10 Mei 2021 pukul 10.31 (GMT+7)

[4] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-backtracking-2021-Bagian1.pdf, diakses pada 10 Mei 2021 pukul 11.11 (GMT+7)

[5] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-backtracking-2021-Bagian2.pdf,, diakses pada 10 Mei 2021 pukul 11.12 (GMT+7)

[6] https://medium.com/geekculture/backtracking-algorithm-95622dcb6ac8, diakses pada 11 Mei 2021 pukul 09.22 (GMT+7)

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, 11 Mei 2021

Christopher Chandrasaputra 13519074