
Implementasi Algoritma Brute Force Pada Permainan 20 Solver Berbasis Web Statis Sederhana

Muhammad Afrian Suwandi – L0123087¹, Muhammad Rafli Werizky – L0123097²

^{1,2} Fakultas Teknologi Informasi dan Sains Data, Universitas Sebelas Maret

¹ srian744@gmail.com; ² rafliwrzky@gmail.com

Abstrak

20 Solver adalah permainan yang bertujuan untuk menemukan cara untuk memanipulasi bilangan–bilangan bulat dengan tujuan hasil perhitungan 20. Penggunaan algoritma yang tepat dapat membantu menemukan semua kombinasi perhitungan dari bilang bilangan tersebut dikarenakan dalam beberapa kasus penggunaan cara manual akan sangat sukar dilakukan. Salah satu algoritma yang dapat dilakukan adalah menggunakan algoritma Brute Force. Tulisan ini akan membahas prinsip algoritma Brute Force dan penerapan dalam permainan 20 Solver yang akan dituangkan dalam website statis sederhana untuk memudahkan pengguna.

Kata-kata kunci : 20 solver, brute force, website

Abstract

20 Solver is a game that aims to find a way to manipulate integers with the aim of calculating 20. Using the right algorithm can help find all combinations of calculations from these numbers because in some cases using the manual method will be very difficult to do. One algorithm that can be done is using the Brute Force algorithm. This article will discuss the principle of the Brute Force algorithm and its application in the 20 Solver game which will be poured into a simple static website to make it easier for users.

Keywords : 20 solver, brute force, website

I. PENDAHULUAN

20 Solver merupakan permainan yang sejenis dengan permainan kartu 24. Perbedaananya terletak pada total hasil perhitungan dan nilai untuk dilakukannya kombinasi perhitungan aritmatika. 20 Solver hanya menggunakan bilangan bulat dalam perhitungannya, sedangkan permainan kartu 24 selain bilangan bulat juga menggunakan kartu remi.

20 Solver merupakan jenis permainan yang mengasah otak dikarenakan diperlukannya daya pikir dan perhitungan yang kuat untuk menemukan kombinasi–kombinasi yang ada dari bilangan bulat dengan tujuan menghasilkan perhitungan 20. Operasi aritmatika yang dapat digunakan dalam permainan ini terbatas pada penjumlahan (+), pengurangan (–), perkalian (×), pembagian (÷), dan tanda kurung (()). Permainan ini juga harus menggunakan prioritas perhitungan bilangan aritmatika.

Permainan ini dapat dimainkan oleh siapapun dan usia berapapun tanpa memandang bulu. Namun kebanyakan, permainan sejenis ini dimainkan oleh para remaja sebagai hal untuk mengasah otak dan bersenang-senang dengan teman sebayanya. Tetapi tidak menutup kemungkinan usia-usia lainnya juga sering memainkan permainan 20 Solver atau permainan sejenisnya.

II. DASAR TEORI

A. Algoritma Brute Force

Algoritma adalah metode terbatas sekumpulan perintah untuk menyelesaikan masalah. Algoritma memiliki kriteria tertentu pada awal kondisi sebelum menjalankan algoritma dan akan berakhir jika semua kondisi awal memenuhi kriteria.

Brute Force adalah pendekatan langsung (*straight forward*) untuk memecahkan suatu masalah yang didasarkan pada pernyataan masalah (*problem statement*) dan definisi konsep. Algoritma ini memecahkan masalah dengan sangat sederhana langsung dan dengan cara yang jelas (*obvious way*).

Algoritma *Brute force* bekerja dengan cara mengecek setiap kedudukan string mulai dari karakter awal hingga karakter akhir. Setelah melakukan pengecekan karakter pertama, maka proses *shift* dilakukan dengan memindahkan string tepat satu posisi ke kanan dan seterusnya. Pembandingan karakter pada string dapat selesai pada posisi manapun selama tahap pencarian, sehingga Algoritma *Brute Force* tidak memerlukan tahap proses.

Karakter Algoritma *Brute Force* :

1. Umumnya tidak efektif dan efisien karena harus memeriksa seluruh kemungkinan yang ada sehingga waktu penyelesaiannya lama.

2. Lebih cocok untuk permasalahan yang memiliki ukuran kecil karena Algoritma *Brute Force* dapat diimplementasikan dengan mudah dan sederhana dibanding algoritma lainnya..
3. Menyelesaikan hampir seluruh permasalahan yang ada. Bahkan, beberapa permasalahan hanya bisa diselesaikan menggunakan Algoritma *Brute Force*.

Kelebihan Algoritma *Brute Force* :

1. Menyelesaikan sebagian permasalahan yang ada.
2. Sederhana dan mudah dipahami.
3. Menghasilkan algoritma yang layak untuk beberapa permasalahan.
4. Menghasilkan algoritma standar untuk beberapa komputasi.

Kekurangan Algoritma *Brute Force* :

1. Algoritma tidak efektif dan efisien.
2. Lambat saat bertemu permasalahan berukuran besar.
3. Tidak kreatif algoritma penyelesaian masalah lainnya.

B. Permainan 20 Solver

Permainan *20 solver* adalah permainan matematika yang melibatkan manipulasi angka menggunakan operasi aritmatika dasar, yaitu penjumlahan, pengurangan, perkalian, pembagian, dan penggunaan tanda kurung, untuk mencapai nilai 20. Permainan ini menggunakan bilangan bulat untuk membentuk kombinasi 4 angka.

C. Website

Menurut Lukmanul Hakim (2004), Website merupakan fasilitas internet yang menghubungkan dokumen dalam lingkup lokal maupun jarak jauh. Dokumen dalam website disebut dengan webpage dan link dalam website dapat digunakan oleh pengguna untuk beralih dari satu halaman ke halaman (*hyertext*) lain baik antar halaman yang disimpan di server yang sama maupun dalam server yang ada di seluruh dunia. Halaman (*page*) dapat di akses atau di baca melalui browser.

Berdasarkan teknologi, web terbagi menjadi 2 jenis, yaitu web statis dan web dinamis. Web statis adalah web dimana pengguna tidak dapat mengubah website secara langsung melalui browser, sedangkan web dinamis adalah situs web yang kontennya dapat diperbaharui secara berkala dengan mudah. Sederhananya jika sebuah *website* tidak memiliki *database* dan pengguna tidak dapat berinteraksi dengan pemilik *website* maka itu disebut web statis. Akan tetapi, jika *website* memiliki *database* dan pengguna dapat berinteraksi dengan pemilik *website* maka itu disebut web dinamis.

III. PEMBAHASAN

Berikut adalah hasil akhir website beserta contoh-contoh outputnya dan pembahasan algoritma *brute force* yang dipakai

A. Algoritma *Brute Force*

Pada laporan ini, aturan permainan 20 solver yang digunakan adalah menggunakan 4 input angka dari *user* atau random.

a. Jumlah kemungkinan solusi

Algoritma *Brute Force* diimplementasikan dengan menggunakan looping yang melakukan iterasi pada seluruh kombinasi kartu yang ada. Jumlah kombinasi kartu dapat ditentukan dengan menggunakan permutasi. Banyak input yang digunakan pada permainan ini adalah 4. Namun, memungkinkan jika tidak semua input berbeda. Kombinasi tersebut memungkinkan 4 input memiliki nilai yang sama. Jumlah kombinasi angka yang mungkin adalah :

4	3	2	1
---	---	---	---

Atau,

$$4!$$

Sehingga, jumlah kombinasi angka yang mungkin = $4 \times 3 \times 2 \times 1 = 24$.

Setiap operasi memiliki 4 kemungkinan yang dapat digunakan sehingga :

4	4	4
---	---	---

Jumlah kombinasi operator = $4 \times 4 \times 4 = 64$.

Banyak kemungkinan operasi yang dilakukan oleh Algoritma *Brute Force* berjumlah $24 \times 64 = 1536$.

b. Implementasi *Brute Froce*

Algoritma *Brute Force* yang diimplementasikan pada laporan ini, menggabungkan *for looping* dengan pola operasi tertentu. Pola tersebut diperoleh dari kombinasi penggunaan tanda kurung sebagai prioritas perhitungan. Berikut adalah kombinasi penggunaan tanda kurung

```
//a $ (b $ (c $ d))
`$a$ $op1$ ($b$ $op2$ ($c$ $op3$ $d))` ,
```

Gambar 3.1. Kombinasi kurung 1
Sumber: Penulis

```
//a $ ((b $ c) $ d)
`$a$ $op1$ ((b$ $op2$ $c)$ $op3$ $d)` ,
```

Gambar 3.2. Kombinasi kurung 2
Sumber: Penulis

```
(((a $ b) $ c) $ d)
`(($a$ $op1$ $b)$ $op2$ $c)$ $op3$ $d)` ,
```

Gambar 3.3. Kombinasi kurung 3
Sumber: Penulis

```
((a $ (b $ c)) $ d)
`(($a$ $op1$ ($b$ $op2$ $c)) $op3$ $d)` ,
```

Gambar 3.4. Kombinasi kurung 4
Sumber: Penulis

```
//(a $ b) $ (c $ d)
`(${a} ${op1} ${b}) ${op2} (${c} ${op3} ${d})`,
```

Gambar 3.5. Kombinasi kurung 5
Sumber: Penulis

Dari operasi tersebut, diperoleh kombinasi-kombinasi sebagai berikut :

- Gambar 3.1. :a \$ (b \$ (c \$ d))
- Gambar 3.2. :a \$ ((b \$ c) \$ d)
- Gambar 3.3. :((a \$ b) \$ c) \$ d
- Gambar 3.4. :(a \$ (b \$ c)) \$ d
- Gambar 3.5. :(a \$ b) \$ (c \$ d)

Kombinasi-kombinasi tersebut akan digunakan pada Algoritma *Brute Force* untuk mencari solusi dari permainan 20 solver.

B. Program

Penyelesaian permainan 20 solver dengan Algoritma *Brute Force* diimplementasikan menggunakan bahasa pemrograman JavaScript dan dituangkan ke dalam website menggunakan HTML dan CSS.

a. UI Website

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>20 Solver</title>
    <link rel="stylesheet" href="style.css" />
  </head>
  <body>
    <!-- Fonts -->
    <link rel="preconnect" href="https://fonts.googleapis.com" />
    <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
    <link href="https://fonts.googleapis.com/css2?family=Open+Sans:ital,wght@0,300,0,400,0,600,0,800&display=swap" rel="stylesheet" />
    <!-- Styles -->
    <style>
      input::-webkit-outer-spin-button,
      input::-webkit-inner-spin-button {
        -webkit-appearance: none;
        margin: 0;
      }
      input[type="number"] {
        appearance: textfield;
        -moz-appearance: textfield;
      }
    </style>
  </body>
  <div class="container">
    <h1>20 Solver</h1>
    <div class="input-container">
      <input type="number" id="num1" min="1" max="30" required />
      <input type="number" id="num2" min="1" max="30" required />
      <input type="number" id="num3" min="1" max="30" required />
      <input type="number" id="num4" min="1" max="30" required />
    </div>
    <div class="button-container">
      <button id="solveBtn" class="solve-btn">Solve</button>
    </div>
    <div class="small-button-container">
      <button id="resetBtn" class="small-btn-reset">Reset</button>
      <button id="randomBtn" class="small-btn-random">Random</button>
    </div>
    <div id="result"></div>
  </div>
  <script src="algoritma20Solver.js"></script>
</html>
```

Gambar 3.6. HTML
Sumber: Penulis

Kode diatas merupakan html *file* yang menjadi struktur atau kerangka utama dari halaman *website* yang kami gunakan. Pada file html ini terdiri dari beberapa *tag* html, yaitu <div>, <h1>, <link>, dan lain-lain.

Dalam *file* html ini, dalam tag <style>, kami menggunakan cara agar masukan dari pengguna yang berupa bilangan bulat tidak memiliki panah ke atas dan ke bawah tujuannya tidak lain dan tidak bukan untuk memperindah tampilan website kami.

```
* {
  font-family: 'pixel';
  scrollbar-width: thin;
  scrollbar-color: rgba(155, 155, 155, 0.5) transparent;
}

@font-face {
  font-family: 'pixel';
  src: url('fonts/PressStart2P-vav7.ttf');
}

body {
  background: #27374D;
  height: 100vh;
  margin: 0;
  display: flex;
  justify-content: center;
  align-items: center;
}

.container {
  background-color: #DDE6ED;
  border-radius: 15px;
  padding: 30px;
  box-shadow: 0 10px 20px rgba(0, 0, 0, 0.2);
  max-width: 500px;
  width: 90%;
}

h1 {
  font-family: "pixel";
  font-size: 40px;
  text-align: center;
  color: #27374D;
}

.input-container {
  display: flex;
  justify-content: space-around;
  margin-bottom: 20px;
  margin-top: 30px;
}

input {
  width: 60px;
  height: 60px;
  font-size: 24px;
  text-align: center;
  border: 2px solid #526082;
  border-radius: 10px;
}

.button-container {
  display: flex;
  justify-content: center;
  margin-bottom: 20px;
}
```

Gambar 3.7. CSS
Sumber: Penulis

Kode diatas merupakan bagian dari *css file* yang digunakan untuk tampilan dari *website* yang kami gunakan. *File* css ini memiliki berbagai hal yang membuat tampilan *website* menjadi lebih enak dipandang dan *user-friendly*.

Kode diatas memuat *font* unduhan dan dimasukkan ke dalam *folder* yang bernama fonts. Kemudian, *font* unduhan tersebut kami gunakan untuk menjadi font utama bagi website yang kami buat. Alasan pemilihan *font* tersebut dikarenakan merepresentasikan sebuah permainan dengan aksien pixel.

Beberapa tombol yang ada di *website* kami rancang sedemikian rupa agar memudahkan pengguna dalam penggunaannya. Untuk tema warna kami menggunakan warna biru sebagai warna dasar dan biru muda sebagai tambahan.

Dalam *website*, kami juga menggunakan beberapa detail kecil seperti mengecilkan *scroll bar* untuk memperindah tampilan dan juga menggunakan hover pada setiap *button*.

b. JavaScript

Berikut merupakan dokumentasi kodenya :

```
1 let hasil = new Set();
2 const op = ["+", "-", "*", "/"];
3
4 function algoritma20Solver(angka) {
5   hasil.clear();
6   for (let i = 0; i < angka.length; i++) {
7     for (let j = 0; j < angka.length; j++) {
8       if (j !== i) {
9         for (let k = 0; k < angka.length; k++) {
10          if (k !== i && k !== j) {
11            for (let l = 0; l < angka.length; l++) {
12              if (l !== i && l !== j && l !== k) {
13                for (let op1 = 0; op1 < op.length; op1++) {
14                  for (let op2 = 0; op2 < op.length; op2++) {
15                    for (let op3 = 0; op3 < op.length; op3++) {
16                      cekBentukOperasi(
17                        angka[i],
18                        angka[j],
19                        angka[k],
20                        angka[l],
21                        op[op1],
22                        op[op2],
23                        op[op3]
24                      );
25                    }
26                  }
27                }
28              }
29            }
30          }
31        }
32      }
33    }
34  }
35 }
36
37 function cekBentukOperasi(a, b, c, d, op1, op2, op3) {
38   const bentukOperasi = [
39     `a $ (b $ (c $ d))`,
40     `a $ ((b $ c) $ d)`,
41     `a $ (op1) ((b $ (op2 $ (c $ (op3 $ d))))`,
42     `a $ (op1) ((b $ (op2 $ (c $ (op3 $ d))))`,
43     `((a $ b) $ c) $ d`,
44     `((a $ b) $ c) $ (op1) $ (op2) $ (op3) $ d`,
45     `((a $ b) $ c) $ d`,
46     `((a $ b) $ (op1) $ (op2) $ (op3) $ d`,
47     `((a $ b) $ (c $ d))`,
48     `((a $ b) $ (op1) $ (op2) $ (c $ (op3 $ d))`,
49   ];
50
51   bentukOperasi.forEach((bentukOperasi) => {
52     try {
53       if (eval(bentukOperasi) === 20) {
54         hasil.add(bentukOperasi);
55       }
56     } catch (e) {}
57   });
58 }
```

Gambar 3.9. Kode program

Sumber: Penulis

Kode di atas adalah bagian utama dari algoritma *Brute Force*. Kode tersebut akan melakukan iterasi ke semua kombinasi angka yang ada dan mengecek hasil operasinya. Jika hasil operasi bernilai 20, program akan menambahkan bentukOperasi ke set hasil yang nantinya dijadikan sebagai output. Fungsi algoritma20solver akan melakukan operasi perhitungan aritmatika. Output juga akan memberikan banyaknya bentuk operasi yang tersedia dengan *method* size untuk set yang bernama hasil.

Kode selengkapnya dapat diakses pada laman berikut <https://github.com/raafiii/20-solver>

```
60 function solve() {
61   const num1 = document.getElementById("num1").value;
62   const num2 = document.getElementById("num2").value;
63   const num3 = document.getElementById("num3").value;
64   const num4 = document.getElementById("num4").value;
65
66   if (isNaN(num1) || isNaN(num2) || isNaN(num3) || isNaN(num4)) {
67     alert("Masukkan Angka (Integer) yang Valid");
68     return;
69   }
70
71   const angka = [num1, num2, num3, num4];
72   algoritma20Solver(angka);
73
74   const resultDiv = document.getElementById("result");
75   if (hasil.size === 0) {
76     resultDiv.innerHTML = "<p>Tidak ada solusi</p>";
77   } else {
78     resultDiv.innerHTML = `<p>Terdapat solusi sebanyak : ${hasil.size}</p>`;
79     hasil.forEach((element) => {
80       resultDiv.innerHTML += `<p>${element}</p>`;
81     });
82   }
83 }
84
85 function reset() {
86   document.getElementById("num1").value = "";
87   document.getElementById("num2").value = "";
88   document.getElementById("num3").value = "";
89   document.getElementById("num4").value = "";
90   document.getElementById("result").innerHTML = "";
91 }
92
93 function generateRandom() {
94   document.getElementById("num1").value = Math.floor(Math.random() * 30) + 1;
95   document.getElementById("num2").value = Math.floor(Math.random() * 30) + 1;
96   document.getElementById("num3").value = Math.floor(Math.random() * 30) + 1;
97   document.getElementById("num4").value = Math.floor(Math.random() * 30) + 1;
98 }
99
100 document.addEventListener("DOMContentLoaded", function () {
101   document.getElementById("solveBtn").addEventListener("click", solve);
102   document.getElementById("resetBtn").addEventListener("click", reset);
103   document
104     .getElementById("randomBtn")
105     .addEventListener("click", generateRandom);
106 });
```

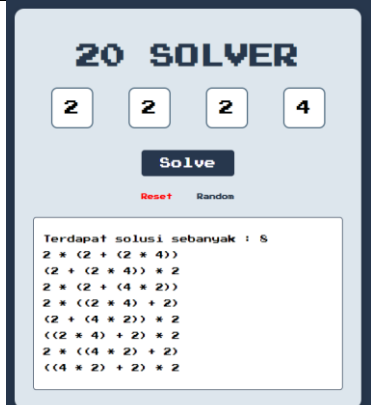
Gambar 3.10. Kode Program (2)

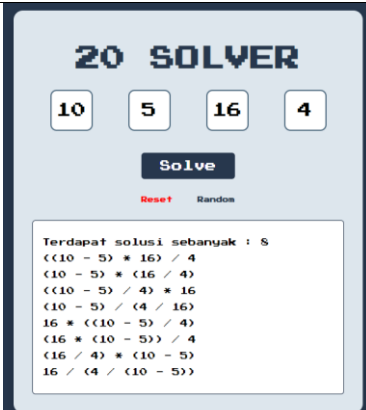
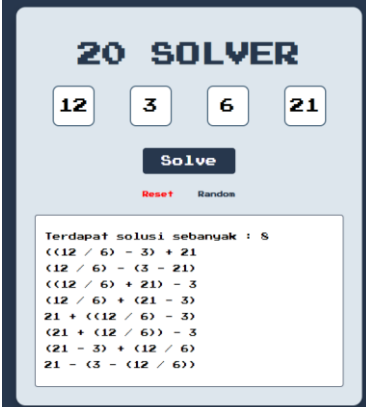
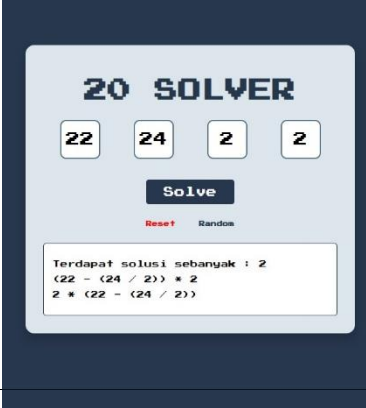
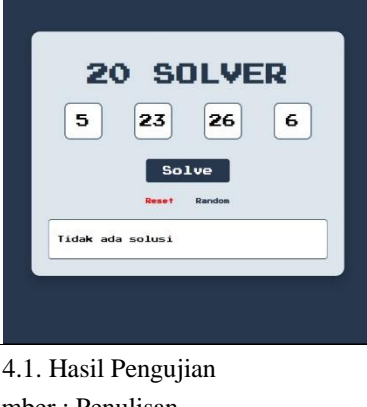
Sumber: Penulis

Kode diatas adalah lanjutan dari kode pada gambar sebelumnya. Kode tersebut digunakan untuk memberi masukan untuk angka pertama hingga ke-empat lalu dilakukan operasi sesuai dengan kode sebelumnya. Lalu, hasilnya akan dihitung sesuai jumlah kemungkinan yang akan ditampilkan pada output. Fungsi reset dilakukan untuk mengosongkan input yang ada. Fungsi random digunakan untuk membuat input random dengan angka maksimal 30. Juga, ditambahkan fitur solve untuk mengerjakan 20 solver.

IV. HASIL PENELITIAN

Dibawah ini merupakan hasil output sampel dari berbagai kombinasi angka dan kemudian akan mengeluarkan segala kemungkinan dari angka-angka tersebut dalam bentuk *website*.

Case	Input	Output
1	2, 2, 2, 4	 <p>The screenshot shows a web application titled "20 SOLVER". It has four input fields containing the numbers 2, 2, 2, and 4. Below these fields is a "Solve" button, and further down are "Reset" and "Random" buttons. The output area displays the message "Terdapat solusi sebanyak : 8" followed by a list of 8 mathematical expressions that evaluate to 20 using the numbers 2, 2, 2, and 4 with various operators and parentheses.</p>

Case	Input	Output
2	10, 5, 16, 4	
3	12, 3, 6, 21	
4	22, 24, 2, 2	
5	5, 23, 26, 6	

Tabel 4.1. Hasil Pengujian

Sumber : Penulisan

Berdasarkan hasil pengujian pada tabel 4.1. terbukti bahwa algoritma *brute force* dapat digunakan untuk menyelesaikan permainan 20 Solver. Pada *case* pertama dengan input (2, 2, 2, 4) akan mengeluarkan 8 total bentuk operasi yang menghasilkan total 20. Pada tes

case pertama juga dapat dilihat walaupun masukan angkanya sama tidak mempengaruhi outputnya dan berjalan dengan normal. Pada *case* 2, 3, dan 4 dapat dengan masukan angka (10, 5, 16, 4), (12, 3, 6, 21), (22, 24, 2, 2) akan menghasilkan total bentuk operasi berturut-turut 8, 8, dan 2. Pada *case* 1, 2, dan 3 dapat dilihat meskipun masukan angkanya berbeda akan tetap memungkinkan menghasilkan keluaran yang sama. Kemudian, pada *case* 5 dengan masukan angka (5, 23, 26, 6) akan menghasilkan keluaran “Tidak ada solusi” dikarenakan seluruh kombinasi dari masukannya tidak ada yang menghasilkan nilai 20.

V. KESIMPULAN DAN SARAN

Dari hasil penelitian dan analisis yang telah dilakukan, didapatkan bahwa Algoritma Brute Force bisa digunakan untuk memperoleh solusi dengan mengecek semua kemungkinan yang ada pada permainan 20 solver. Permainan ini masih tergolong mudah dan sederhana untuk diterapkan algoritma karena worst case yang dibuat dengan Algoritma *Brute Force* untuk 20 solver hanya 1536 iterasi. Oleh karena itu, tidak dianjurkan untuk menggunakan Algoritma *Brute Force* jika permasalahannya kompleks dan memiliki ukuran yang besar karena tidak efektif dan efisien.

VI. PENUTUP

Puji dan syukur kami panjatkan kepada Tuhan Yang Maha Esa atas berkat rahmat dan karunia-Nya, kami dapat menyelesaikan laporan ini dengan lancar. Kami mengucapkan terima kasih kepada Bapak Fajar Muslim, S.T., M.T., atas bimbingannya selama masa perkuliahan sehingga kami dapat menyelesaikan laporan ini dengan baik. Kami juga ingin mengucapkan terimakasih atas materi dan referensi yang telah diberikan. Semoga dengan kemurahan hati Bapak, kami diberikan nilai A untuk Mata Kuliah Design dan Analisis Algoritma.

VII. LAMAN TERKAIT

Link repository GitHub :

<https://github.com/raaflii/20-solver>

Link website :

<https://20-solver-six.vercel.app/>

VIII. DAFTAR PUSTAKA

- [1] “MUNIR, RINALDI. “ALGORITMA BRUTE FORCE BAGIAN 1, ” URL: [https://informatika.stel.itb.ac.id/~RINALDI.MUNIR/STMIK/2021-2022/ALGORITMA-BRUTE-FORCE-\(2022\)-BAG1.PDF](https://informatika.stel.itb.ac.id/~RINALDI.MUNIR/STMIK/2021-2022/ALGORITMA-BRUTE-FORCE-(2022)-BAG1.PDF) DIAKSES PADA 18 SEPTEMBER 2024
- [2] “MUNIR, RINALDI. “ALGORITMA BRUTE FORCE BAGIAN 2, ” URL: [https://informatika.stel.itb.ac.id/~RINALDI.MUNIR/STMIK/2021-2022/ALGORITMA-BRUTE-FORCE-\(2022\)-BAG2.PDF](https://informatika.stel.itb.ac.id/~RINALDI.MUNIR/STMIK/2021-2022/ALGORITMA-BRUTE-FORCE-(2022)-BAG2.PDF) DIAKSES PADA 18 SEPTEMBER 2024
- [3] “IMPLEMENTASI ALGORITMA BRUTE FORCE DALAM PENCARIAN MENU PADA APLIKASI PEMESANAN COFFEE (STUDI KASUS : TANAMERA COFFEE)” URL: <https://ejournal.sisfokomtek.org/index.php/jikom/article/download/129/130#:~:text=ALGORITMA%20BRUTE%20FORCE%20ADALAH%20ALGORITMA,ALGORITMA%20AKAN%20>

- [4] "NGODING WEB DINAMIS ATAU STATIS, APA PERBEDAANNYA?" URL:
[HTTPS://WWW.DICODING.COM/BLOG/NGODING-WEB-DINAMIS-ATAU-STATIS/](https://www.dicoding.com/blog/ngoding-web-dinamis-atau-statis/) DIAKSES PADA 20 SEPTEMBER 2024

IX. DAFTAR TUGAS

Nama	Pembagian Tugas
Muhamad Afrian Suwandi L0123087	-Membuat HTML dan CSS untuk Website. -Membuat algoritma <i>brute force</i> menggunakan python. -Membuat laporan. -Membuat video demo. -Deploy web
Muhammad Rafli Werizky L0123097	-Membuat HTML dan CSS untuk Website. -Membuat algoritma <i>brute force</i> menggunakan javascript. -Membuat laporan. -Membuat video demo. -Mengupload di github. -Deploy web