Aplikasi Algoritma String Matching pada Website Pencarian Hero Dota 2

Menggunakan Algoritma Knuth–Morris–Pratt

Muhammad Alfandavi Aryo Utomo 13519211
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13519211@std.stei.itb.ac.id

***Abstract*—Game Online memiliki banyak sekali update demi mendukung perkembangan game tersebut, salah satunya update hero khususnya Dota 2 yang memiliki jumlah hero sebanyak 121 (7.29). Untuk mencari nama hero yang diinginkan, digunakan algoritma knuth-morris-pratt dalam pencarian kita, agar selalu mengikuti perkembangan game Dota 2 ini dan tidak lupa dengan nama-namanya yang unik.**

Keywords—Dota 2, Game, hero, KMP, String Matching.

I. PENDAHULUAN

Dewasa ini, game online merupakan hobi bagi kalangan anak muda, tak terkecuali bagi orang dewasa yang bisa dibilang tidak sedikit, salah satu dari game online tersebut adalah Dota 2 yang merupakan singkatan dari (Defense Of The Ancients). Dota awalnya hanyalah sebuah mod dari game populer besutan blizzard entertainment (Warcraft 3) dan berkembang sangat pesat oleh dukungan komunitas yang sangatlah mendukung dikala itu. Singkat cerita, Dota berkembang menjadi salah satu cabang e-sport yang memiliki hadiah dan komunitas terbesar

yang pernah ada, dan perlombaan internasionalnya bernama The Internationals diadakan terakhir kali pada tahun 2019 (TI9) yang memiliki total hadiah sebesar $34.300.000 (34,3 juta dollar / 487 milliar rupiah) dan akan selalu meningkat setiap tahunnya. Dibutuhkan pengalaman dan kemampuan yang sangatlah tinggi dalam memenangkan satu permainan karena begitu kompleksnya Dota ini.

Dota 2 Memiliki jumlah hero sebanyak 121 (7.29 current patch) dan akan bertambah seiring waktu, oleh karena itu untuk membantu para pemula yang sering terlupa akan nama-nama hero dota yang unik, penulis membuat sebuah website dengan penerapan algoritma KMP dalam metode pencariannya.

II. LANDASAN TEORI

Landasan teori yang akan digunakan dalam aplikasi algoritma string matching dalam pencarian hero Dota 2:

A. *Algoritma String Matching*

Algoritma pencocokan string ini digunakan untuk mencari kesamaan suatu kata atau pattern tertentu didalam target nya (kata atau kumpulan kata bersambung), pemanfaatannya sangat banyak didalam kehidupan kita seperti pencarian dalam teks editor, mesin pencarian (google, bing, duckduckgo), analsis citra (digunakan dalam sidik jari dan forensik), bio informatika (dalam pencocokan rantai dna), dan lain-lain.

B. *Algoritma Brute Force*

Pencocokan dilakukan dengan mengecek pattern dari awal teks sampai akhir teks dengan langkah satu-per satu, misalkan kita memiliki Teks (" NOBODY NOTICED HIM ") dan Pattern (" NOT "), maka proses pencocokan akan dilakukan seperti berikut.

Teks: NOBODY NOTICED HIM
Pattern: NOT

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

Gambar 1. Algoritma Brute Force pada Pencocokan String

(Sumber: Materi kuliah Ir. Rinaldi Munir, M.T)

Proses diawali dengan mencocokkan pattern diawal teks, kemudian apabila tidak sama, akan bergerak kekanan dengan +1 setiap ditemukan ketidak samaan, proses akan berhenti apabila sudah ditemukan pattern yang sama didalam teks atau sudah mencapai akhir teks dan tidak ditemukan hasilnya.

Kompleksitas untuk algoritma ini adalah O(n), kasus terbaik apabila pattern ditemukan diawal kalimat, dan kasus terburuk apabila pettern ditemukan diakhir kalimat atau tidak sama sekali.

C. Algoritma Knuth-Morris-Pratt

Algoritma ini sama seperti brute force, tetapi lebih "pintar" dan efektif dalam melakukan pencariannya, dengan menyimpan informasi pergeseran yang perlu dilakukan sehingga tidak secara satu per satu seperti brute force dan terbukti lebih efektif dan cepat. Dikembangkan oleh Donald Knuth, James Morris, dan Vaughan Pratt pada 1970.

Penggeseran pada KMP seperti berikut, misalkan kita memiliki teks [i ... + n-1] dan diasumsikan mismatching pada [i + j] dengan pattern [j], dengan syarat 0 < j < n maka teks [i ... + j – 1] = pattern [0 .. j - 1] dan a = teks [i + j] tidak sama dengan b = pattern [j]. Saat dilakukan pergeseran, sangatlah mungkin jika awalan v dari patter mungkin bisa sama dengan sebagian akhiran u dari sebagian teks, sehingga dilakukan pergeseran pattern agar awalan v tersebut sama dengan akhiran u.

Sehingga KMP ini akan efektif apabila kita bisa menentukan seberapa banyak pergeseran yang harus dilakukan, prosesnya dilakukan dengan pembuatan tabel yang berisi next [j] yang mana posisi [j] setelah digeser, sehingga kita bisa menggeser pattern sebesar j – next [j] relatif terhadap teks tersebut.

Langkah-langkah prosesnya sebagai berikut:

1. KMP memulai proses pencocokan pada awal teks.

2. Pattern dicocokkan tiap karakter dengan teks, hingga memenuhi salah satu keadaan berikut

   a. Hingga akhir pattern semua karakter cocok dengan teks, maka akan menghasilkan kondisi dimana ditemukan kecocokan pada teks.

   b. Karakter pada pattern tidak cocok dengan teks.

3. Apabila terdapat ketidakcocokan dengan teks, maka dilakukan border function, seperti berikut.

Gambar 2. KMP Usage

(Sumber: Materi kuliah Ir. Rinaldi Munir, M.T)

Kompleksitas waktu untuk border function adalah O(m) dan pencocokannya O(n) sehingga ditotal kompleksitasnya O(m+n), lebih efektif daripada brute force.

Kelebihan dari algoritma ini diantaranya:

1. Datanya menjamin efisiensi kasus terburuk

2. Pemrosesan awal dan waktu selalu tepat telah ditentukan sebelumnya.

3. Tidak ada masukan kasus terburuk atau tidak disengaja.

4. Lebih disukai di mana string pencarian di ruang yang lebih besar lebih mudah dan lebih efisien dicari karena itu menjadi algoritma linier waktu.

5. Algoritme perlu bergerak mundur dalam teks masukan. Ini sangat menguntungkan dalam memproses file besar

Namun dibalik kelebihan pasti memiliki kekurangan, diantaranya :

1. Data tidak berfungsi dengan baik saat ukuran huruf meningkat.

2. Untuk memproses file yang sangat besar, itu juga membutuhkan sumber daya dalam bentuk prosesor sehingga agak sulit untuk diterapkan pada organisasi kecil.

D. Dota 2

Dota 2 merupakan sebuah permainan online bertipe MOBA (Multiplayer Online Battle Arena) yang mempertandingkan 2 buah tim untuk saling menghancurkan ancient musuh mereka , berikut gambar dari ancient masing-masing sisi (kiri : dire, kanan : radiant)

Gambar 3. Ancient Dota

Sumber :

https://cdnx.kincir.com/production/media/2018/desember/cerita-dota-2-ancient-ketiga/ancient-dota-2.jpg

yang mana tentunya tidaklah mudah untuk merebut ancient musuh karena dijaga oleh beberapa tower beserta hero yang siap mengorbankan nyawanya demi melindungi hal ini.

Jumlah hero di Dota 2 sebanyak 113 pada tahun 2016, 115 pada 2017, 116 pada 2018, 119 pada 2019, 120 pada 2020, dan 121 untuk 2021 (patch 7.29) dan jumlah

ini akan terus berkembangan seiring updates yang diberikan kepada komunitas Dota 2 ini.

Setiap hero meliki kemampuan dan atribut tersendiri, hal ini memungkinkan setiap pemain untuk mengkombinasikan kemampuan setiap hero yang dimainkan untuk mencapai kemenangan, atribut setiap hero dibagi menjadi 3 yaitu:

1. Strength Hero

Beberapa hero yang memiliki atribut ini yaitu Axe, Centaur Warrunner, Dragon Knight, Io, Alchemist, dll. Memiliki ciri-ciri atribut gain strength yang besar dan memiliki HP dengan jumlah tinggi, biasa digunakan sebagai pelindung/tanker didalam teamfight atau initiator didalam war dan tidak takut akan nyawanya, Strength hero biasa diletakkan pada role offlane ataupun mid, namun tidak menutup kemungkinan untuk diletakkan sebagai safelaner bahkan support 4,5. Beberapa player yang memiliki banyak signature hero di strength yaitu OG.Ceb, Secret.Zai,Eg.IceIceIce, dll.

2. Agility Hero

Beberapa hero yang memiliki atribut ini yaitu Phantom Lancer, Monkey King, Troll Warlord, Anti Mage, Arc Warden, dll. Memiliki ciri-ciri atribut gain agility yang besar dan memiliki armor dengan jumlah tinggi, biasa digunakan sebagai main attacker/dps didalam teamfight atau assassin yang membunuh support musuh dengan cepat didalam war, Agility hero biasa diletakkan pada role safelaner ataupun mid, namun tidak menutup kemungkinan untuk diletakkan sebagai offlaner bahkan support 4,5. Beberapa player yang memiliki banyak signature hero di agility yaitu OG.Ana, Secret.Matumbaman,Eg.RTZ, dll.

3. Intelligence Hero

Beberapa hero yang memiliki atribut ini yaitu Invoker, Tinker, Enigma, Leshrac, Puck, dll. Memiliki ciri-ciri atribut gain intelligence yang besar dan memiliki mana dengan jumlah tinggi, biasa digunakan sebagai main support didalam teamfight atau initator dan disabler didalam war, Intelligence hero biasa diletakkan pada role 4/5 ataupun mid, namun tidak menutup kemungkinan untuk diletakkan sebagai safelaner bahkan offlaner.Beberapa player yang memiliki banyak signature hero di intelligence yaitu OG.N0tail, OG.JerAx , OG.Saksa, Secret.Puppey, Nigma.Kuroky, dll.

III. Implementasi

Penulis membuat sebuah website dengan url https://alfandaviau.github.io/Dota2HeroFinder/ sebagai implementasi tugas makalah ini, pada website bisa dilakukan pencarian nama hero yang diinginkan, algoritma KMP digunakan pada main.js seperti berikut

```
function buildPatternTable(pattern) {
    const patternTable = [0];
    let prefixIndex = 0;
    let suffixIndex = 1;

    while (suffixIndex < pattern.length) {
      if (pattern[prefixIndex] === pattern[suffixIndex]) {
        patternTable[suffixIndex] = prefixIndex + 1;
        suffixIndex += 1;
        prefixIndex += 1;
      } else if (prefixIndex === 0) {
        patternTable[suffixIndex] = 0;
        suffixIndex += 1;
      } else {
        prefixIndex = patternTable[prefixIndex - 1];
      }
    }

    return patternTable;
  }

function KMP(text, word) {
    if (word.length === 0) {
      return 0;
    }

    let textIndex = 0;
    let wordIndex = 0;

    const patternTable = buildPatternTable(word);

    while (textIndex < text.length) {
      if (text[textIndex] === word[wordIndex]) {
        if (wordIndex === word.length - 1) {
          return (textIndex - word.length) + 1;
        }
        wordIndex += 1;
        textIndex += 1;
      } else if (wordIndex > 0) {
        wordIndex = patternTable[wordIndex - 1];
      } else {
        wordIndex = 0;
        textIndex += 1;
      }
    }

    return -1;
}
```

Gambar 4. Algoritma KMP yang digunakan

(Sumber : Penulis)

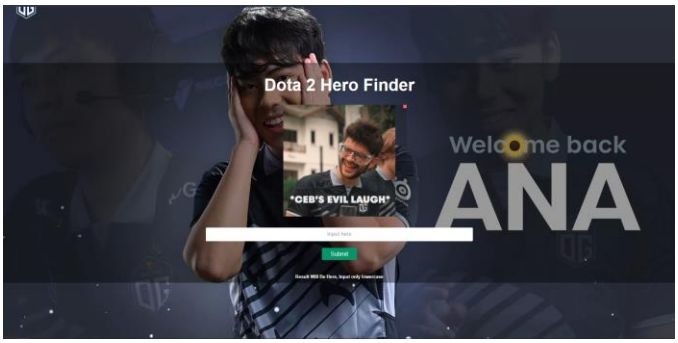
```
var heroDB = ['abaddon','alchemist','ancient apparition','anti-mage','arc warden
// console.log(KMP("dawnbreaker","nbre"));
function main(){
  var x = document.getElementById("hero").value;
  // console.log(x);
  // for (var i = 0; i < x.length; i++){
  //   if (KMP(heroDB[i],x) != 0){
  //     document.write(heroDB[i]);
  //     console.log(i);
  //   }
  // }
  var outputnya = "Result : <br>";
  for (var i = 0; i < heroDB.length; i++){
    if (KMP(heroDB[i],x) != -1){
      outputnya += (`${heroDB[i]} <br>`);
      // document.getElementById("res").innerHTML = "asu";

        // document.write(`${heroDB[i]} <br>`);
        // console.log(i);
    }
  }
  document.getElementById("res").innerHTML = outputnya;
}
```

Gambar 5. Struktur Data dan Fungsi Utama

(Sumber : Penulis)

Website dibuat dengan vanilla js dan tailwind css dalam prosesnya, dengan tampilan sebagai berikut.



Gambar 6. Tampilan Website

(Sumber : Penulis)

Input dari pengguna bisa dilakukan pada search bar tersebut dengan prasyarat huruf kecil semua dan asumsi input benar, kemudian tombol sumbit ditekan untuk menampilkan hasil berupa nama hero yang memiliki pola seperti input.

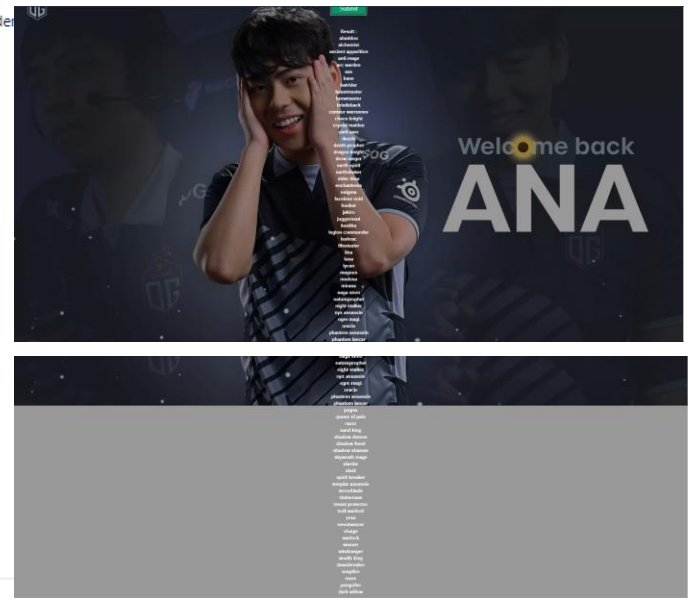
IV. PENGUJIAN DAN PEMBAHASAN

A. *Skenario Pengujian*

Pengujian dilakukan pada web browser biasa seperti mozilla, chrome, namun penulis menggunakan brave browser sebagai browser utama yang akan diuji. Algoritma akan diuji dengan input yang random dan dites apakah program mengeluarkan hasil yang diinginkan dan tidak terjadi kesalahan dalam prosesnya.

B. *Pengujian*

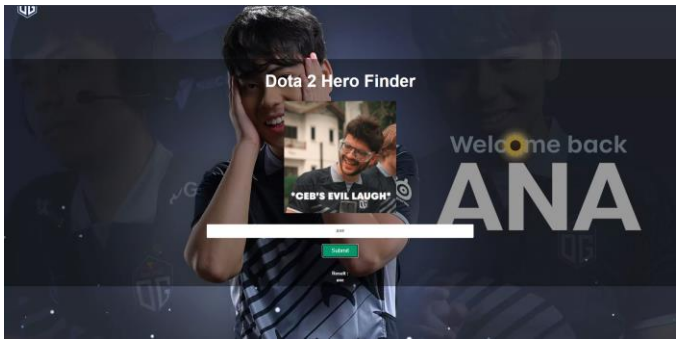
1. Pengujian dengan input “a”



Gambar 7 dan 8. Pengujian dengan input “a”

(Sumber : Penulis)

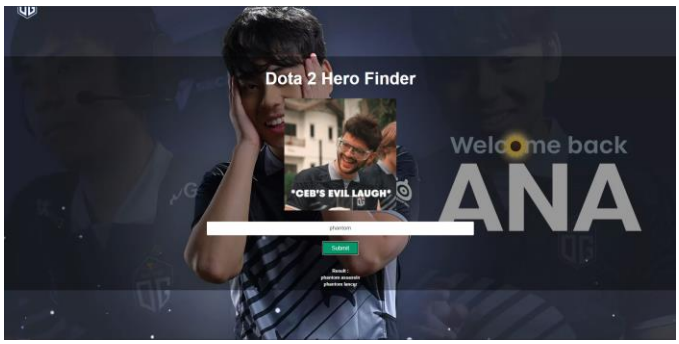
2. Pengujian dengan input “axe”



Gambar 9. Pengujian dengan input “axe”

(Sumber : Penulis)

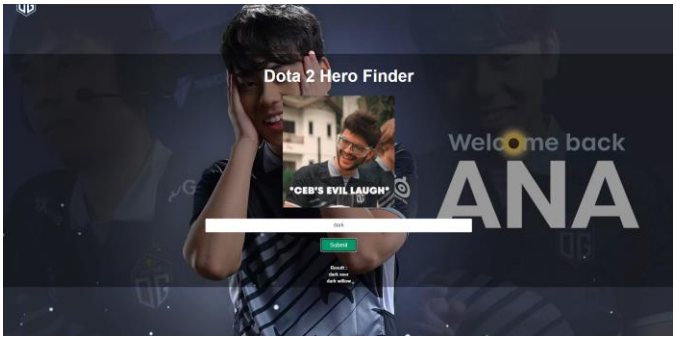
3. Pengujian dengan input “phantom”



Gambar 10. Pengujian dengan input “phantom”

(Sumber : Penulis)

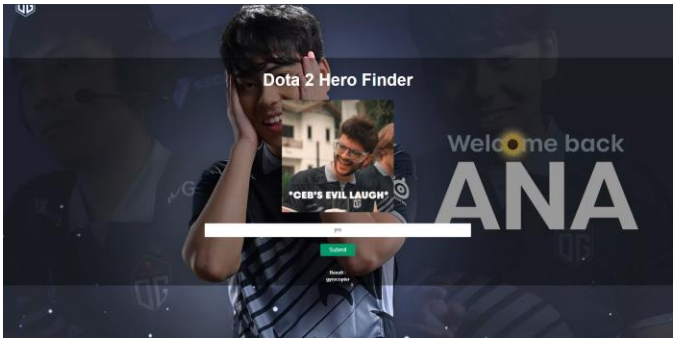
4. Pengujian dengan input “dark”



Gambar 11. Pengujian dengan input “dark”

(Sumber : Penulis)

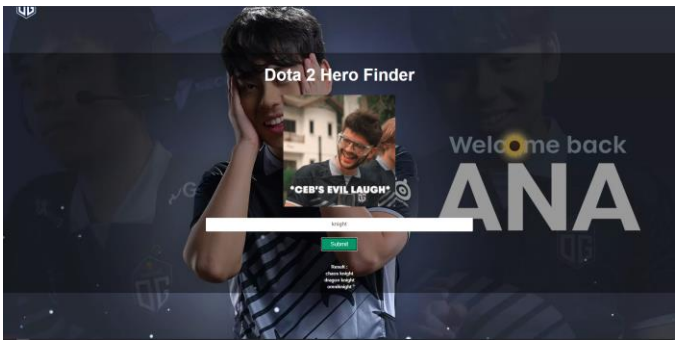
5. Pengujian dengan input “yro”



Gambar 12. Pengujian dengan input “yro”

(Sumber : Penulis)

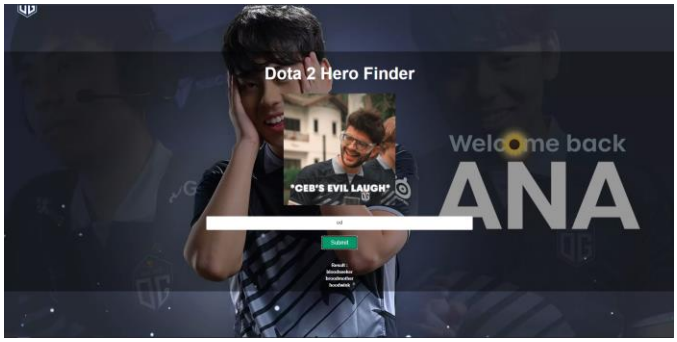
6. Pengujian dengan input “knight”



Gambar 13. Pengujian dengan input “knight”

(Sumber : Penulis)

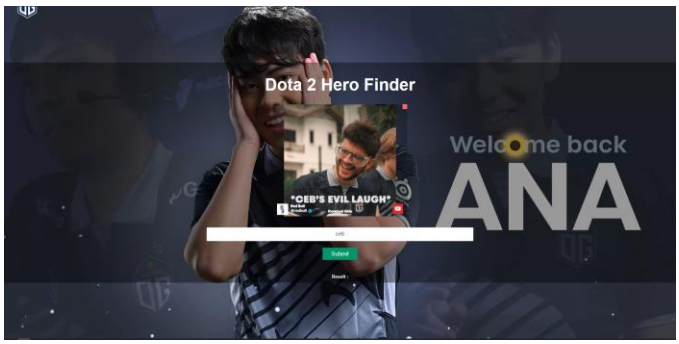
7. Pengujian dengan input “od”



Gambar 14. Pengujian dengan input “od”

(Sumber : Penulis)

8. Pengujian dengan input “ceb”



Gambar 15. Pengujian dengan input “ceb”

(Sumber : Penulis)

C. Pembahasan

Pada input pertama yaitu karakter “a”, dihasilkan semua hero yang memiliki karakter “a” pada nama mereka, mulai dari abaddon, axe,alchemist,dll hingga melebihi halaman wallpaper karena jumlah hero yang memiliki karakter “a” pada nama mereka terlampau banyak.

Kemudian pada input kedua yaitu “axe” untuk mencari nama “axe” secara spesifik, didapatkan hasil axe juga karena tidak ada hero lain selain axe yang memiliki karakter “axe” pada nama mereka.

Input ketiga yaitu “phantom” untuk mencari semua nama hero yang memiliki unsur phantom pada nama mereka, didapatkan phantom assassin dan phantom lancer sebagai hasilnya.

Input keempat yaitu “dark” untuk mencari semua nama hero yang memiliki unsur dark pada nama mereka, didapatkan dark seer dan dark willow sebagai hasilnya.

Input kelima yaitu “yro” untuk mencari semua nama hero yang memiliki unsur yro pada nama mereka, didapatkan g”yro”copter sebagai hasilnya.

Input keenam yaitu “knight” untuk mencari semua nama hero yang memiliki unsur knight pada nama mereka, didapatkan chaos knight, dragon knight, omniknight sebagai hasilnya.

Input ketujuh yaitu “od” untuk mencari semua nama hero yang memiliki unsur od pada nama mereka, didapatkan bloodseeker,broodmother,hoodwinksebagai hasilnya.

Kemudian input terakhir yaitu “ceb” yang mana merupakan nama player dan bukan nama hero, didapatkan hasil kosong karena tidak ditemukan nama hero yang mengandung unsur ceb didalam namanya.

Hasil ini sesuai dengan keinginan penulis dan tidak ada kesalahan yang ditemukan dan algoritma efektif dalam melakukan pencarian nama hero ini karena prosesnya cepat dan tidak ditemukan kesalahan yang signifikan.

REFERENCES

[1] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Pencocokan-string-2021.pdf

[2] https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2018-2019/String-Matching-dengan-Regex-2019.pdf

[3] https://www.geeksforgeeks.org/kmp-algorithm-for-pattern-searching/

[4] https://www-igm.univ-mlv.fr/~lecroq/string/node8.html

[5] https://www.dotafire.com/dota-2/heroes

[6] https://dota2.fandom.com/wiki/Heroes_by_release

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.

Malang, 11 Mei 2021

Muhammad Alfandavi Aryo Utomo 13519211