Aplikasi Algoritma Knuth-Morris-Pratt Pada Permainan "Number Search"

Inka Anindya Riyadi / 13518038
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13518038@std.stei.itb.ac.id

***Abstract*—Algoritma adalah deretan instruksi yang disusun untuk menyelesaikan suatu masalah. Salah satunya dalah penyelesaian permainan Number Search dengan menggunakan pencocokan string. Algoritma pencocokan string yang digunakan adalah Algoritma Knuth-Morris-Pratt.**

***Keywords*—Algoritma KMP, Algoritma Knuth-Morris-Pratt, string, Number Search, *string matching*, pencocokan string**

I. PENDAHULUAN

Algoritma adalah deretan instruksi yang disusun untuk memecahkan suatu masalah, yaitu untuk memperoleh keluaran yang diinginkan. Dalam pemograman,algoritma mempunyai peran yang sangat penting karena algoritma merupakan dasar dari keberjalanan program tersebut. Untuk menjalankan suatu program, terdapat beberapa algoritma yang umum digunakan , diantaranya adalah algoritma *brute force* dan algoritma *greedy.* Masing-masing algoritma mempunyai penyelesaian serta pengaplikasian yang berbeda. Dengan adanya algoritma, banyak persoalan-persoalan yang terdapat dalam kegiatan sehari-hari yang dapat diselesaikan.

Permainan adalah bentuk aktivitas yang menyenangkan yang dilakukan semata-mata untuk aktivitas itu sendiri, bukan karena ingin memperoleh sesuatu yang dihasilkan dari aktivitas tersebut. Sampai sekarang, banyak sekali permainan yang ada, mulai dari permainan yang menyenangkan hingga permainan yang menggunakan kemampuan logika dalam menyelesaikannya. Salah satunya adalah permainan Number Search. Permainan ini merupakan sebuah permainan yang terdapat sebuah tabel beisi angka, dan terdapat beberapa kata yang harus ditemukan dalam tabel tersebut. Permainan ini mempunyai kesamaan dengan permainan Word Search, namun perbedaannya adalah permainan Word Search berisi dengan alfabet.

Algoritma untuk menyelesaikan permainan Number Search adalah algoritma pencocokan string. Algoritma pencocokan string terbagi menjadi 4, algoritma *brute force,* Knuth-Morris-Pratt, Booyer-Moore, dan *regular expression* atau yang biasa dikenal dengan *regex.* Keempat algoritma ini mempunyai karakteristiknya masing-masing. Algoritma yang paling umum digunakan adalah algoritma *brute force*. Namun, pada makalah ini penulis akan menggunakan algoritma Knuth-Morris-Pratt untuk menyelesaikan permainan Number Search.

II. LANDASAN TEORI

A. Pencocokan String (*String Matching*)

String adalah sebuah tipe data yang digunakan untuk menyimpan barisan karakter. Dalam beberapa bahasa pemrograman, string merupakan sekumpulan karakter/ *array of character*.

Apabila kita mempunyai sebuah string S dengan ukuran/panjang M, maka :

$$S = X_0,X_1,\ldots,X_{M-1}$$

dimana X[i], $0 \le i \le m-1$, merupakan karakter pada posisi i. Contoh string adalah "konsep" dimana M merupakan panjang string yaitu 6 dan X_0 = 'k'.

Sebuah string mempunyai *prefix* dan *suffix*. *Prefix* adalah huruf atau kelompok huruf yang ditempatkan di awal kata. Sedangkan, *suffix* adalah huruf atau kelompok huruf yang ditempatkan di belakang kata. Apabila kita mempunyai suatu string S dengan panjang m, maka prefix dari string tersebut adalah S[0..i] dan suffix dari string tersebut adalah S[i…m-1] dimana m merupakan panjang string dan i merupakan indeks diantara 0 dan m-1, $0 \le i \le m-1$.

Persoalan pencocokan string dirumuskan sebagai berikut:

1. Diberikan sebuah teks, yaitu sebuah string/teks T dengan panjang N
2. Diberikan sebuah *pattern* P, yaitu sebuah string dengan panjang M dimana M<N yang akan dicari dalam teks

B. Algoritma Knuth-Morris-Pratt

Algoritma Knuth-Morris-Pratt merupakan salah satu algoritma penyelesaian persoalan pencocokan string. Algoritma ini awalnya disusun oleh James H. Morris dan ditemukan secara independen oleh Donald Ervin Knuth yang merupakan seorang *computer scientist* dan *Professor Emeritus* di Stanford University. Algoritma ini bisa dibilang mempunyai kesamaan seperti algoritma *brute force* yang mengecek semua posisi di sebuah teks T dan melihat apakah cocok dengan sebuah *pattern* P yang dimulai pada posisi tersebut. Algoritma *brute force*

memiliki kompeksitas rata-rata O(m+n) dimana m merupakan panjang *pattern* dan n merupakan panjang teks.

Algoritma Knuth-Morris-Pratt merupakan bentuk efisiensi dari algoritma *brute force* dengan cara melakukan pergeseran yang pintar sehingga jumlah pengecekan teks bisa lebih efektif. Algoritma ini mengecek sebuah teks dimulai kiri dan bergeser ke kanan. Untuk menyelesaikan persoalan ini dalam algoritma Knuth-Morris-Pratt, kita mengasumsikan bahwa terdapat teks T dengan pattern P dimana M merupakan panjang pattern dan N merupakan panjang teks.

Dalam algoritma ini, terdapat fungsi pinggiran KMP atau yang biasa disebut KMP *Border Function / Failure Function*. Fungsi pinggiran ini melakukan pengecekan pada *pattern* untuk menemukan kecocokan *prefix* dari pattern dengan *pattern* itu sendiri. b(k) merupakan fungsi pinggiran dimana ukuran terbesar dari suatu prefix pattern P[0..k] yang juga *suffix* dari pattern P[1..k], dimana k adalah posisi sebelum terjadi ketidakcocokan yaitu j-1, dimana j merupakan posisi terjadi ketidakcocokan.

Contoh apabila kita mempunyai sebuah pattern “ababd”, dapat didefinisikan sebagai berikut:

j	0	1	2	3	4
P[j]	a	b	a	b	d
k	-	0	1	2	3

Tabel 1 : Tabel pattern “ababd”
Sumber : Penulis

Untuk menghitung fungsi pembatas, maka dimulai dari j=0, yaitu k tidak dapat didefinisikan sehingga nilai fungsi pembatas juga tidak dapat definisikan.

Fungsi pembatas selanjutnya pada j=1, sehingga k = 0. *Prefix* dari pattern adalah P[0..0] yaitu “a”. *Suffix* dari pattern adalah [1..1] yaitu “b”. Karena *suffix* tidak sama dengan *prefix* , maka nilai fungsi pembatas adalah 0 atau b(0) = 0.

Fungsi pembatas selanjutnya pada j=2, sehingga k=1. *Prefix* dari pattern adalah P[0..1] yaitu “a”, dan “ab”. *Suffix* dari *pattern* adalah P[1..2] yaitu “a”, dan “ba”. Karena terdapat kesamaan pada *prefix* dan *suffix* yang berpanjang 1 yaitu “a”, maka nilai fungsi pembatas adalah 1 atau b(1) = 1.

Fungsi pembatas selanjutnya pada j=3, sehingga k=2. *Prefix* dari *pattern* adalah P[0..2] yaitu “a”, “ab”, dan “aba”. *Suffix* dari *pattern* adalah P[1..3] yaitu “b”, “ab”, dan “bab”. Karena terdapat kesamaan pada *prefix* dan *suffix* yang berpanjang 2 yaitu “ab”, maka nilai fungsi pembatas adalah 2 atau b(2) = 2.

Fungsi pembatas selanjutnya pada j=4, sehingga k=3. *Prefix* dari *pattern* adalah P[0..3] yaitu “a”,”ab”,”aba”, dan ”abab”. *Suffix* dari *pattern* adalah P[1..4] yaitu “d”, “bd”. “abd”, dan “babd”. Karena tidak terdapat kesamaan pada *prefix* dan *suffix* maka nilai fungsi pembatas adalah 0 atau b(3) = 0.

Dengan itu, bisa didefinisikan fungsi pembatas dari string “ababd” adalah sebagai berikut:

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

Tabel 2 : Tabel fungsi pembatas pattern “ababd”
Sumber : Penulis

Pergeseran yang dilakukan pada algoritma Knuth-Morris-Pratt adalah berdasarkan fungsi pembatas yang sudah didefinisikan dari pattern. Persyaratannya adalah sebagai berikut:

1. Apabila terjadi ketidakcocokan karakter pada pattern P[j] yaitu P[j]!=T[i], dan k=j-1, maka nilai j menjadi b(k)
2. Apabila terjadi kecocokan karakter pada P[j] yaitu P[j]=T[i], maka nilai i menjadi i+1, dan nilai j menjadi j+1

Berikut merupakan contoh pergeseran dengan algoritma Knuth-Morris-Pratt dengan pattern P “ababd” dan teks T “ababcabcabababd”

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
T[j]	a	b	a	b	c	a	b	c	a	b	a	b	a	b	d

Tabel 3 : Tabel teks “ababcabcabababd”
Sumber : Penulis

Pengecekan string dimulai dari i=0 dan j=0, dimana ketika i=0 dan j=0 terjadi kecocokan, sehingga nilai i dan j masing-masing bertambah 1. Ketika i=1 dan j=1 terjadi kecocokan kembali sehingga i dan j masing-masing bertambah 1.

Pengecekan selanjutnya berlanjut sampai terjadi ketidakcocokan pada i=4 dan j=4 dimana T[i] = “c” dan P[j] = “d”. Karena terjadi ketidakcocokan maka kita melihat tabel fungsi pembatas ketika j=4 yaitu 0, sehingga mengulangi pengecekan pada i=4 dan j=0.

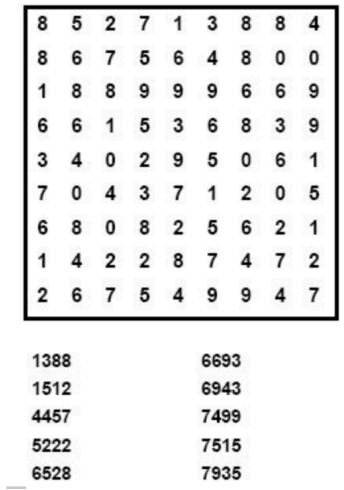
Namun, pada selanjutnya terjadi ketidakcocokan kembali sehingga fungsi pembatas dari j=0 yaitu tidak terdefinisi. Karena tidak terdefinisi, maka nilai ditambah 1 dan j=0. Pengecekan pada i=5 dan j=0 terjadi kecocokan, sehingga terjadi penambahan i dan j masing-masing 1. Pengecekan selanjutnya pada i=6 dan j=1 dimana terjadi kecocokan sehingga terjadi penambahan i dan j masing-masing 1. Pengecekan selanjutnya pada i=7 dan j=2 terjadi ketidakcocokan, sehingga melakukan pergeseran dimana fungsi pembatas ketika j=2 yaitu b(1) = 1.

Pengecekan selanjutnya terjadi ketika i=7 dan j=1 dimana terjadi ketidakcocokan sehingga terjadi pergeseran kembali. Fungsi pembatas ketika j=1 yaitu b(0) = 0. Pengecekan selanjutnya yaitu i=7 dan j=0 dimana terjadi ketidakcocokan

kembali sehingga terjadi pergeseran. Fungsi pembatas ketika j=0 yaitu tidak terdefinisi, sehingga i ditambah 1. Pengecekan selanjutnya yaitu i=8 dan j=0 dan terjadi kecocokan. Ketika i=12 dan j=3 terjadi kecocokan sehingga harus melakukan pergeseran. Fungsi pembatas ketika j=4 yaitu b(3) = 0. Pengecekan selanjutnya yaitu i=12 dan j=2. Pengecekan terus cocok hingga indeks terakhir yaitu i=14. Sehingga terdapat sebuah pattern “ababd” pada string/teks “abababcabcababd”.

C. Algoritma penyelesaian game “Number Search”

Number Search merupakan sebuah permainan dimana terdapat suatu tabel yang berisi kumpulan angka, dan terdapat beberapa string yang terdapat pada tabel tersebut. String dapat ditemukan secara horizontal, vertical, maupun diagonal.



Gambar 1: Number Search
Sumber : http://www.childrenspuzzles.net/ns127/3.pdf

Menyelesaikan persoalan ini dibagi menjadi 4 tahap, yaitu:

1. Memilih suatu pattern
2. Menghitung fungsi pembatas dari pattern
3. Menentukan string yang ingin diproses
4. Melakukan pengecekan pattern pada suatu string tersebut

Pertama, program akan memilih suatu pattern yang ingin diproses. Pattern ini dipilih dari kata-kata yang tertera pada tabel kata. Pemilihan ini dipilih secara iterative.

Kemudian, setelah pemilihan pattern, maka dapat dihitung fungsi pembatas dari pattern tersebut. Untuk menghitung fungsi pembatas, maka dapat digunakan suatu algoritma yaitu sebagai berikut:

Algoritma yang penulis gunakan untuk menghitung fungsi pinggiran dalam bahasa Python. Algoritma ini akan mengembalikan sebuah array berisi fungsi pembatas dari sebuah pattern. Berikut merupakan algoritma fungsi pembatas :

```
def borderfunction(pattern):
    m = len(pattern)
    lps = [0 for i in range (m)]
    substring = [0 for i in range (m)]
    prefix = ['0' for i in range (m)]
    suffix = ['0' for i in range (m)]
    lps[0] = 0
    i = 1
    while i<m:
        maks = 0
        ctrsuffix = i
        for j in range (i):
            print("j = " + str(j))
            ctrprefix = 0

            maks1 =0
            for k in range(j+1):
             if(pattern[k]==pattern[ctrsuffix+k]):
                    maks1+=1
             else:
                    maks1=0
                    break;
            if(maks1>maks):
                maks = maks1
            ctrsuffix -= 1
        lps[i] = maks
        i+=1
    print(lps)
    return lps
```

Kemudian, apabila fungsi pembatas sudah ditemukan, proses selanjutnya adalah menentukan string yang ingin diproses. Untuk mencari suatu string pada game Number Search, maka proses pencarian akan dibagi menjadi 6, yaitu :

1. Secara horizontal dari kiri ke kanan
2. Secara vertical dari atas ke bawah
3. Secara diagonal dari kiri ke kanan
4. Secara horizontal dari kanan ke kiri
5. Secara vertical dari bawah ke atas
6. Secara diagonal dari kanan ke kiri

Pencarian dimulai dari proses pertama yaitu secara horizontal dari kiri ke kanan dan akan terus melakukan pencarian sampai bertemu dengan pattern yang sesuai

Langkah terakhir adalah melakukan pengecekan pattern tersebut terhadap string yang dipilih. Algoritma untuk mencari suatu pattern P dalam suatu string/teks T dalam bahasa Python adalah sebagai berikut.:

```
def kmp(pattern, text):
    len_pat = len(pattern)
    len_txt = len(text)

    lps = [0 for i in range (len_pat)]
    i = 0 #index untuk txt
    j = 0 #index untuk pattern
    lps = borderfunction(pattern)
    while i<len_txt:
        if(text[i]==pattern[j]):
            i+=1
            j+=1
        else:
            if(j!=0):
                j = lps[j-1]
            else:
                i+=1
        if(j==len_pat):
            print ("Found pattern at
index " + str(i-j) )

            j = lps[j-1]
            return True
    return False
```

Algoritma ini akan mengembalikan nilai true apabila telah ditemukan suatu pattern dalam suatu teks dan akan menampilkan ke layer index yang ditemukan pada suatu string. Algoritma ini akan mengembalikan nilai false apabila tidak ada pattern dalam suatu string.

IV. APLIKASI ALGORITMA KNUTH-MORRIS-PRATT PADA PERMAINAN NUMBER SEARCH

Permainan Number Search pada dasarnya mempunyai ukuran n x n yang berukuran bervariasi. Permainan ini merupakan permainan yang terdiri atas tabel yang berisi angka-angka, dan terdapat beberapa kalimat yang terdapat pada tabel dan dapat dicari secara horizontal dari kiri ke kanan, vertikal dari atas ke bawah, diagonal dari kiri ke nanan, horizontal dari kanan ke kiri, vertikal dari bawah ke atas, maupun diagonal dari kiri ke kanan. Dalam permainan ini. suatu string yang harus dicari dalam tabel pasti tertera dalam tabel dan hanya muncul satu kali dalam tabel.

Berikut merupakan contoh Number Search yang berukuran 9 x 9 dan akan diselesaikan menggunakan algoritma Knuth-Morris-Pratt. Dalam permainan ini terdapat 10 kata yang harus dicari dalam tabel yaitu “2682”, ”3022”, “3762”, “4091”, “4398”, “6157”, “6426”, “7072”, “7826”, dan “8268”.

4	9	7	0	8	6	5	9	0
4	8	4	7	0	6	4	5	2
3	6	2	1	6	5	3	0	2
1	6	0	4	6	1	3	9	7
1	0	3	8	4	2	5	9	4
7	1	0	4	7	0	7	2	1
9	8	2	6	8	2	6	6	9
1	6	2	4	6	1	5	7	0
6	5	3	6	7	8	9	3	4

2682
3022
3762
4091
4398
6157
6426
7072
7826
8268

Gambar 2 : Contoh Permainan Number Search yang Akan diselesaikan
Sumber : http://www.childrenspuzzles.net/ns127/3.pdf

Pencarian pertama adalah pattern pertama yaitu “2682”.

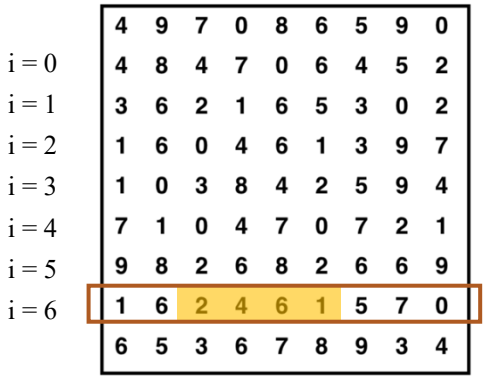
j	0	1	2	3
P[j]	2	6	8	2
k	-	0	1	2
b(k)	-	0	0	1

Tabel 4 : Tabel fungsi pembatas pattern “2628”
Sumber : Penulis

Pertama, melakukan pengecekan untuk setiap baris horizontal dari kiri ke kanan dan dimulai dari baris pertama. Pada baris pertama (i=0) dengan string “497086590” tidak ditemukan pattern “2682”. Pada baris kedua (i=1) dengan string “484706452” tidak ditemukan pattern “2682”. Pada baris ketiga (i=2) dengan string “362165302” tidak ditemukan pattern “2682”. Pada baris keempat (i=3) dengan string “160461397” tidak ditemukan pattern “2682”. Pada baris kelima (i=4) dengan string “103842593” tidak ditemukan pattern “2682”. Pada baris keenam (i=5) dengan string “710470721” tidak ditemukan pattern “2682”. Pada barie ketujuh (i=6) terlihat bahwa terdapat pattern “2682” pada baris ke 6 pada string “98**2682**669”. Proses pencarian untuk pattern “2682” diberhentikan karena sudah ditemukan pattern pada proses pertama.

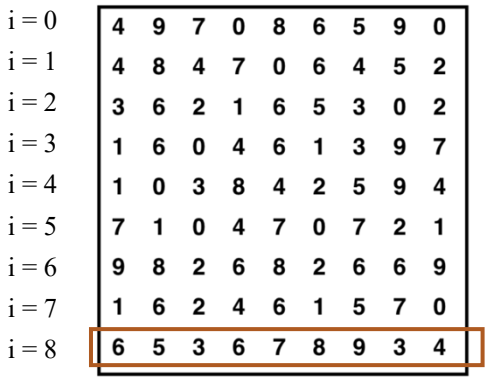
i	0	1	2	3	4	5	6	7	8
T[i]	9	8	**2**	**6**	**8**	**2**	6	6	9
	2	6	8	2					
		2	6	8	2				
			2	**6**	**8**	**2**			

Tabel 5 : Tabel teks “74203022”
Sumber : Penulis



Gambar 3 : Pencarian Horizontal dari Kiri ke Kanan dengan Pattern “2682”
Sumber : http://www.childrenspuzzles.net/ns127/3.pdf

Pencarian pattern kedua adalah pattern kedua yaitu “3022”. Pencarian dilakukan secara horizontal dari kiri ke kanan dimulai dari baris pertama.



Gambar 4 : Pencarian Horizontal dari Kiri ke Kanan dengan Pattern “3022”
Sumber : http://www.childrenspuzzles.net/ns127/3.pdf

j	0	1	2	3
P[j]	3	0	2	2

k	-	0	1	2
b(k)	-	0	0	0

Tabel 6 : Tabel fungsi pembatas pattern “3022”
Sumber : Penulis

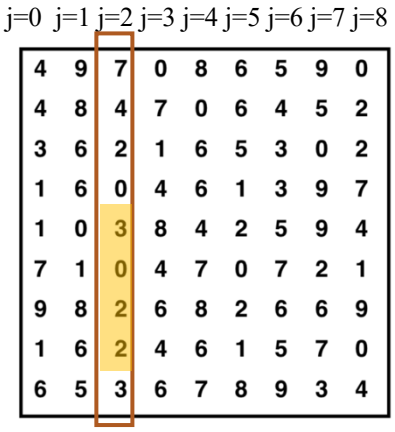
Pada baris pertama (i=0) dengan string “497086590” tidak ditemukan pattern “3022”. Pada baris kedua (i=1) dengan string “484706452” tidak ditemukan pattern “3022”. Pada baris ketiga (i=2) dengan string “362165302” tidak ditemukan pattern “3022”. Pada baris keempat (i=3) dengan string “160461397” tidak ditemukan pattern “3022”. Pada baris kelima (i=4) dengan string “103842593” tidak ditemukan pattern “3022”. Pada baris keenam (i=5) dengan string “710470721” tidak ditemukan pattern “3022”. Pada baris ketujuh (i=6) dengan string “982682669” tidak ditemukan pattern “3022”. Pada baris kedelapan (i=7) dengan string “162461570” tidak ditemukan pattern “3022”. Pada baris kesembilan (i=8) dengan string “653678934” tidak ditemukan pattern “3022”.

Namun, pada pencarian secara horizontal dari kiri ke kanan tidak ditemukan, sehingga perlu melakukan proses pencarian selanjutnya yaitu pencarian secara vertical dari atas ke bawah.

Pada kolom pertama (j=0) dengan string “443117916” tidak ditemukan pattern “3022”. Pada kolom kedua (j=1) dengan string “986601865” tidak ditemukan pattern “3022”. Pada kolom ketiga (j=2) terlihat ditemukan pattern “3022” dengan string “7420**3022**3”.

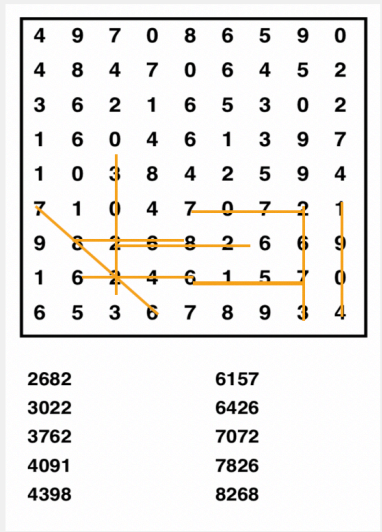
i	0	1	2	3	4	5	6	7	8
T[i]	7	4	2	0	**3**	**0**	**2**	**2**	3
	3	0	2	2					
		3	0	2	2				
			3	0	2	2			
				3	0	2	2		
					3	**0**	**2**	**2**	

Tabel 7 : Tabel teks “74203022”
Sumber : Penulis



Gambar 5 : Pencarian Vertikal dari Atas ke Bawah dengan Pattern “3022”
Sumber : http://www.childrenspuzzles.net/ns127/3.pdf

Pencarian pattern selanjutnya yaitu “3762”, “4091”, “4398”, “6157”, “6426”, “7072”, “7826”, dan “8268” juga akan melakukan proses pencocokan string yang sama dengan pattern “2682” dan “3022” yaitu melakukan proses yang dimulai secara horizontal dari kiri ke kanan dan melakukan pecocokan dengan algoritma Knuth-Morris-Pratt disetiap iterasinya.



Gambar 5 : Hasil Akhir yang Diharapkan
Sumber : http://www.childrenspuzzles.net/ns127/3.pdf

V. LINK VIDEO YOUTUBE

Sebagai pendukung dari pembuatan makalah mengenai penyelesaian permainan Number Search menggunakan algoritma Knuth-Morris-Pratt, penulis juga telah menyiapkan video yang memuat penjelasan mengenai algorima Knuth-Morris-Pratt dan penyelesaian permainan dengan menggunakan program bahasa Python. Video ini diupload di Youtube dengan link https://youtu.be/HwYueaVXQV0 .

V. KESIMPULAN

Algoritma adalah deretan instruksi yang disusun untuk memecahkan suatu masalah, yaitu untuk memperoleh keluaran yang diinginkan. Permainan adalah bentuk aktivitas yang menyenangkan yang dilakukan semata-mata untuk aktivitas itu sendiri, bukan karena ingin memperoleh sesuatu yang dihasilkan dari aktivitas tersebut. Salah satunya adalah permainan Number Search Penerapan dari algoritma Knuth-Morris-Pratt. Algoritma untuk menyelesaikan permainan Number Search adalah algoritma pencocokan string. Algoritma pencocokan string terbagi menjadi 4, algoritma brute force, Knuth-Morris-Pratt, Booyer-Moore, dan regular expression atau yang biasa dikenal dengan regex. Dengan algoritma KMP, dapat ditemukan solusi dari permainan Number Search dengan pengecekan string pada pengecekan horizontal,vertikal, maupun diagonal.

VII. UCAPAN TERIMA KASIH

Pertama-tama, penulis mengucapkan terimakasih kepada puji syukur kepada Tuhan Yang Maha Esa atas berkat dan rahmatnya, penulis bisa menyelesaikan tugas makalah ini. Penulis juga mengucapkan terimakasih kepada kedua orangtua penulis serta kakak yang selalu mendoakan akan kesuksesan penulis dan turut mendukung penulis. Penulis juga mengucapkan terimakasih kepada Ibu Ulfa selaku dosen mata kuliah Strategi Algoritma, yang selama ini memberikan ilmu Strategi Algoritma yang sangat membantu pengerjaan makalah ini dan Bapak Rinaldi Munir, yang selama satu semester ini selalu membantu dengan adanya *website* yang berisi materi-materi kuliah, latihan-latihan soal untuk kuis dan ujian, dan semua dokumen pembelajaran, soal dan lainnya yang sangat berguna dalam proses pembelajaran Strategi Algoritma.

REFERENSI

[1] http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/Pencocokan-String-(2018).pdf. Diakses pada 2 Mei 2020.
[2] http://www.childrenspuzzles.net/number-search.php. Diakses pada 2 Mei 2020
[3] https://idcloudhost.com/mengenal-apa-itu-algoritma-definisi-ciri-ciri-dan-contohnya/. Diakses pada 2 Mei 2020
[4] http://dyanrch.weebly.com/design-course/definisi-permainan-menurut-para-ahli. Diakses pada 2 Mei 2020

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.

Jakarta, 2 Mei 2020

Inka Anindya Riyadi / 13518038