IMPLEMENTASI ALGORITMA *BRUTE-FORCE* PADA *TIKTOK SUGGESTED ACCOUNTS*

## HALAMAN JUDUL

A picture containing shape

Description automatically generated

Disusun Oleh:

|  |  |
| --- | --- |
| N a m a  NIM | : Bilhaq Avi Dewantara  : 120140141 |

**PROGRAM STUDI TEKNIK INFORMATIKA – PROGRAM SARJANA**

**JURUSAN TEKNOLOGI PRODUKSI DAN INDUSTRI**

**INSTITUT TEKNOLOGI SUMATERA**

**2023**

## HALAMAN PERNYATAAN KEASLIAN UTS

Yang bertanda tangan di bawah ini:

Nama : Bilhaq Avi Dewantara

NIM : 120140141

Tugas akhir dengan judul:

IMPLEMENTASI ALGORITMA *BRUTE-FORCE* PADA *TIKTOK SUGGESTED ACCOUNTS*

Menyatakan bahwa seluruh komponen dan isi dalam Ujian Tengah Semester ini adalah hasil karya saya sendiri. Apabila dikemudian hari terbukti ada beberapa bagian dari karya ini adalah bukan hasil karya sendiri, Hasil Ujian Tengah Semester yang diajukan sebagai hasil karya sendiri ini siap ditarik kembali dan siap menanggung resiko dan konsekuensi apapun.

Demikian surat pernyataan ini dibuat, semoga dapat dipergunakan sebagaimana mestinya.

|  |
| --- |
| Lampung Selatan, 31 Maret 2023 |
|  |
| Bilhaq Avi Dewantara |
| NIM.120140141 |

## GLOSARIUM

|  |  |
| --- | --- |
| *Auto Captions* | Memberikan *subtitle* pada video secara otomatis. |
| *Suggested Accounts* | Akunyang disarankan, biasanya terdapat pada fitur sebuah aplikasi media sosial. |
| *Followed* | Pengguna media sosial yang memilih untuk mengikuti akun atau profil orang lain. |
| *Mutual* | Situasi pada media sosial dimana dua pengguna mengikuti satu sama lain. |
| *Heuristic* | Pendekatan atau teknik pemecahan masalah menggunakan metode berbasis pengalaman. |

## DAFTAR ISI

[**HALAMAN JUDUL i**](#_Toc131318052)

[**HALAMAN PERNYATAAN KEASLIAN UTS ii**](#_Toc131318053)

[**GLOSARIUM iii**](#_Toc131318054)

[**DAFTAR ISI iv**](#_Toc131318055)

[**DAFTAR TABEL iv**](#_Toc131318056)

[**DAFTAR GAMBAR iv**](#_Toc131318057)

[**BAB I PENDAHULUAN 1**](#_Toc131318058)

[1.1 Latar Belakang 1](#_Toc131318059)

[1.2 Rumusan Masalah 1](#_Toc131318060)

[1.3 Tujuan Penelitian 1](#_Toc131318061)

[**BAB II METODOLOGI DAN PEMBAHASAN 2**](#_Toc131318062)

[2.1 Metodologi 2](#_Toc131318063)

[2.1.1 Studi Pustaka 2](#_Toc131318064)

[**BAB III HASIL DAN KESIMPULAN 6**](#_Toc131318065)

[3.1 Hasil 6](#_Toc131318066)

[3.1.1 Struktur Data dan Implementasi 6](#_Toc131318067)

[3.1.2 Implementasi Algoritma 6](#_Toc131318068)

[3.1.3 Pengujian 12](#_Toc131318069)

[3.2 Kesimpulan 15](#_Toc131318070)

[**DAFTAR PUSTAKA 16**](#_Toc131318071)

[**LAMPIRAN 17**](#_Toc131318072)

## DAFTAR TABEL

[Tabel 1 Contoh Algoritma *Brute-Force* 3](#_Toc131318174)

[Tabel 2 Fungsi *fileHandle()* 6](#_Toc131318175)

[Tabel 3 Fungsi *notInList()* 8](#_Toc131318176)

[Tabel 4 Fungsi *mutualNum()* 8](#_Toc131318177)

[Tabel 5 Fungsi *sortingPair()* 9](#_Toc131318178)

[Tabel 6 Implementasi Algoritma Pemrograman Python 9](#_Toc131318179)

[Tabel 7 File Pengujian 13](#_Toc131318180)

## DAFTAR GAMBAR

[Gambar 1 Metode Penelitian 2](#_Toc131318184)

[Gambar 2 Tiktok 4](#_Toc131318185)

[Gambar 3 *Suggested Accounts* Tiktok 5](#_Toc131318186)

[Gambar 4 Pengujian 1 14](#_Toc131318187)

[Gambar 5 Pengujian 2 15](#_Toc131318188)

## BAB I PENDAHULUAN

### Latar Belakang

Pada zaman sekarang ini kita bisa melihat banyaknya aplikasi dunia maya yang mulai populer dan menjadi wadah dalam beberapa aktivitas bermanfaat, seperti menyambungkan silaturahmi dengan orang lain, mendapatkan teman baru, kegiatan jual-beli, dan lain-lain. Hal tersebut membuat banyak pengguna yang menekuni dunia maya.

Tiktok adalah salah satu sosial media yang sedang populer di semua umur saat ini. Aplikasi ini memberikan pelayanan ke penggunanya dalam mengirim pesan berupa gambar ataupun video. Dengan banyaknya pengguna, aplikasi ini hari demi hari menambahkan banyak fitur juga.

Salah satu fitur yang terdapat pada Tiktok adalah *suggested accounts*. Fitur tersebut menyediakan rekomendasi akun Tiktok untuk di-*follow* berdasarkan kontak pada nomor *handphone* dan rekomendasi dariakun-akun yang *followed* oleh *mutual* pengguna. Dengan adanya fitur ini, dapat membantu pengguna dalam mencari teman-temannya di dunia nyata yang mempunyai akun media sosial juga.

Pada fitur *suggested accounts* di aplikasi Tiktok ini tentunya banyak algoritma yang diimplementasikan, salah satunya ialah algoritma *Brute-Force*. Oleh sebab itu, penulis ingin melakukan penelitian dalam implementasi algoritma *Brute-Force* untuk fitur aplikasi media sosial Tiktok.

### Rumusan Masalah

* Bagaimana cara algoritma di implementasikan pada kehidupan sehari-hari?
* Bagaimana cara penerapan dari algoritma *Brute-Force* pada fitur *suggested accounts* aplikasi Tiktok?

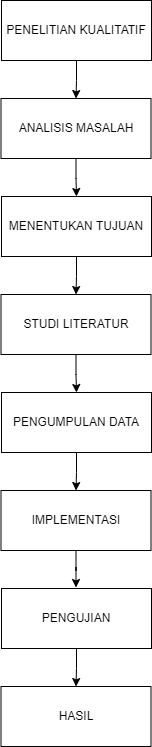
### Tujuan Penelitian

Tujuan dari penelitian ini agar mengtahui bagaimana cara algoritma diterapkan pada kehidupan sehari-hari. Khususnya pada fitur *suggested accounts* di aplikasi Tiktok ini dalam meningkatkan akurasi untuk memberikan rekomendasi akun-akun relevan bagi penggunanya. Dengan adanya penelitian ini dapat memberikan kontribusi dan solusi dalam pengembangan fitur *suggested account* pada Tiktok dan juga memberikan pemahaman mendalam terkait penggunaan algoritma *Brute-Force* di aplikasi media sosial.

## BAB II METODOLOGI DAN PEMBAHASAN

### Metodologi

Dalam melakukan penelitian tentunya berdasarkan metode penelitian, sehingga dapat mempermudah jalannya penelitian. Dengan begitu, berikut merupakan metode penelitian yang akan dilakukan pada penelitian ini.



Gambar 1 Metode Penelitian

#### Studi Pustaka

1. Algoritma *Brute-Force*

Algoritma *Brute-Force*, juga dikenal sebagai *exhaustive search* ialah metode yang sederhana namun efektif untuk menyelesaikan suatu permasalahan. Algoritma ini dapat digunakan dengan cara yang terstruktur, mudah dipahami, dan cepat. Meskipun demikian, algoritma ini dianggap kurang cerdas dan akan bekerja lebih lambat ketika dihadapkan pada data input yang besar.

Proses penyelesaian menggunakan algoritma *Brute-Force* membutuhkan waktu yang cukup lama dan volume komputasi yang besar, sehingga sering disebut sebagai algoritma yang naif.

Oleh sebab itu, dalam mempercepat pencarian dapat menerapkan fungsi *heuristic* agar kemungkinan solusi yang muncul tidak diproses secara penuh karna *heuristic* merupakan teknik yang mengacu tentang penyelesaian masalah berbasis pengalaman secara optimal.

1. Kekuatan algoritma *Brute-Force*
   * Memiliki kemampuan untuk digunakan dalam hampir semua jenis masalah (*wide applicability*).
   * Sangat sederhana dan mudah dipahami.
   * Dapat menghasilkan solusi yang layak untuk beberapa masalah krusial seperti *searching, sorting, matching,* dan perkalian matriks.
   * Algoritma Brute-Force juga dapat menghasilkan algoritma standar untuk tugas-tugas komputasi yang umum seperti penjumlahan atau perkalian n buah bilangan.
   * Dapat menentukan nilai minimum atau maksimum dalam sebuah larik.
2. Kelemahan algoritma *Brute-Force*
   * Memiliki keterbatasan dalam hal keefektifan.
   * Algoritma yang dihasilkan mungkin kurang optimal dan cenderung lebih lambat untuk masukan data yang besar.
   * Tidak sekonstruktif atau se-kreatif strategi pemecahan masalah lainnya, yang dapat menghasilkan solusi yang lebih efisien dan efektif.
3. Contoh Penggunaan Algoritma *Brute*-*Force*

Algoritma *Brute-Force* digunakan pada pencarian elemen terbesar dari sebuah *array,* berikut merupakan implementasi dari algoritma ini.

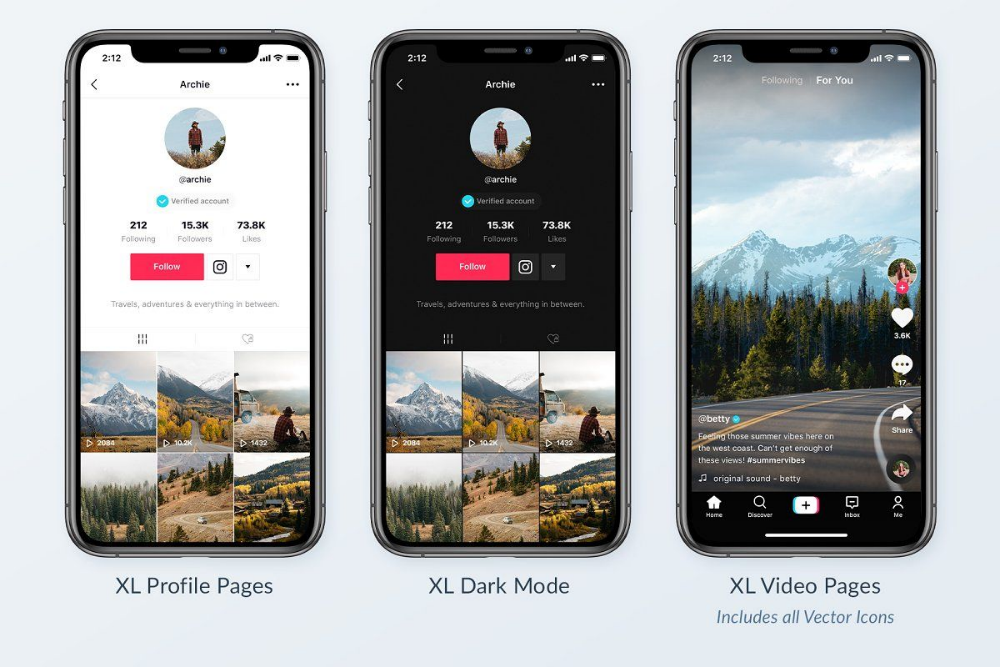
Tabel 1 Contoh Algoritma *Brute-Force*

|  |
| --- |
| **procedure** *maxElement(input a1,a2,.....an* : **int, output***max* : **int**){  *Searching max element based a1,a2,.....an*  *Store max element on variabel max.*  *Input : a1,a2,.....an*  *Output : max*  }  **Declare**  *k* : **int**  **Algorithm :**  *max* *← a1*  ***for*** *k* *←2* ***to*** *n* ***do***  ***if*** *ak > max* ***then***  *max* *← ak*  ***endif***  ***endfor*** |

Beberapa contoh dari penerapan algoritma Brute-Force pada kehidupan sehari-hari biasanya dipakai pada pencocokan string, penyelesaian game sudoku, crossword puzzle, dan lain-lain.

1. *Tiktok Suggested Accounts*

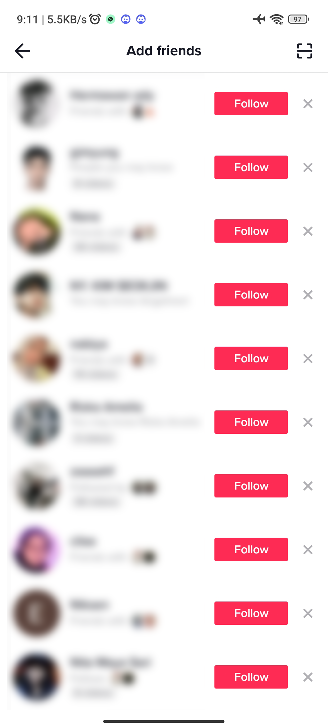
Tiktok adalah sebuah aplikasi media sosial yang membagikan momen penggunanya berupa video maupun foto kepada publik. Tiktok mulai populer di Indonesia apda tahun 2018, tetapi diblokir oleh Kementerian Komunikasi dan Informatika karena dianggak aplikasi tersebut tidak memberi manfaat yang baik. Namun, pada tahun 2020, Tiktok mulai populer kembali di berbagai kalangan masyarakat Indonesia termasuk artis, pejabat, pebisnis, dan lain-lain (Berita Hari Ini, 2020).



Gambar 2 Tiktok

Terdapat banyak fitur pada aplikasi Tiktok yang memberi dukungan pada penggunanya dalam membuat konten-konten kreatif, seperti penambahan musik pada video yang akan dibuat, pemberian efek dan filter pada video, *Auto captions*, dan *Live streaming*.

Kemudian terdapat juga fitur *suggested accounts* pada Tiktok yang memberikan rekomendasi akun kepada *user* berdasarkan keadaan akun *user* tersebut. Keadaannya bisa berupa akun yang diikuti oleh teman *user* atau nomor yang tersimpan pada akun Tiktok *user*.



Gambar 3 *Suggested Accounts* Tiktok

Dengan adanya fitur tersebut dapat mempermudah *user* dalam mencari teman baru, fitur ini juga dapat memfilter pengguna dalam mencari teman baru pada media sosial tanpa takut adanya bahaya orang asing yang mungkin terjadi dengan menambahkan teman baru.

## BAB III HASIL DAN KESIMPULAN

### Hasil

#### Struktur Data dan Implementasi

Dalam penelitian ini penulis menggunakan bahasa pemrograman Python 3.x dalam menyelesaikan permasalahan dan tidak menggunakan *library* apapun.

Pada implementasi program terdapat 2 struktur data, yaitu :

* List

List adalah jenis struktur data yang memungkinkan pengguna untuk menyimpan sejumlah nilai atau elemen dalam urutan tertentu. Setiap elemen dalam list disebut sebagai value, dan keseluruhan daftar nilai tersebut disimpan antara tanda kurung siku [].

List dapat digunakan untuk berbagai aplikasi, seperti *stored* data berasal dari file tertentu, meyimpan nama-nama akun yang telah diverifikasi, menyimpan pasangan data, serta menyimpan nama-nama akun yang direkomendasikan.

* Pair

Adalah struktur data yang mempunyai 2 nilai berkorespondensi. Hal ini juga berguna dalam menyimpan banyaknya data akun yang mengikuti satu sama lain dan jumlah *mutual*.

#### Implementasi Algoritma

Pada pengimplementasian algoritma Brute-Force dalam mencari suggested accounts mempunyai tahapan tertantu, antara lain :

1. Mendapatkan/ menerima *list* yang berisi daftar akun saling *follow* antara satu sama lain. Pada kondisi ini, *list* berasal *file* dengan cara berikut :

Tabel 2 Fungsi *fileHandle()*

|  |
| --- |
| **function***fileHandle(fileName : string) → listOfPair*  { *I.S.* File terdapat isi/ terdefinisi dan tidak kosong }  { *F.S. Return list* berupa *Pair* yang mempunyai 2 elemen nama akun yang saling *follow* antara satu sama lain.}  **DECLARE**  *List : listOfPair*  *Line : string*  *Temp : listOfString*  **ALGORITHM**  ***if exist****(fileName)* ***then***  ***open input****(fileName)*  *List ←* []  ***repeat***  *Line ←* ***readline****(fileName)*  *Temp ←* ***line.split***  *append* ***Temp*** *to* ***List***  ***until end of for***  **→ *List*** |

1. Mendapatkan nama *user* yang dibuat sebagai acuan dalam mencari *suggested account*-nya.
2. Meng-*create* *empty* *list* sebagai penyimpan akun-akun yang diikuti teman-teman dari akun *user* dan juga menyimpan nama-nama akun yang di-*follow* oleh akun *user*.
3. Mulai melakukan iterasi untuk seluruh elemen yang berupa *pair* sesuai pada *list* hasil *open file*.
4. Mencari elemen yang mempunyai nilai nama akun *user*.
5. Menyimpan nilai pasangan elemen sebelumnya pada *empty* *list* untuk akun yang di-*follow* oleh akun *user*.
6. Selanjutnya iterasi pada *list* kembali dalam mencari elemen salah satu nilainya merupakan pasangan dari akun *user* yang telah ditemukan pada langkah sebelumnya. Semua elemen yang memenuhi kriteria tersebut akan disimpan pada *empty list* sebagai tempat menyimpan nama *suggested accounts*.
7. Menghapuskan akun *mutual* dari *list* yang menyimpan nama-nama *suggested accounts*.
8. Mulai mencari jumlah *mutual* yang dimiliki *list* dengan akun *user*. Berikut merupakan implementasi algoritmanya.

Tabel 3 Fungsi *notInList()*

|  |
| --- |
| **function***notInList(name: string, List : listOfPair) → boolean*  { *I.S.* File terdapat isi/ terdefinisi dan tidak kosong, nama terdefinisi }  { *F.S. Return* nilai *true* bila *name* tidak pada *list* dan *false* bila nama terdapat pada *list* }  **DECLARE**  **ALGORITHM**  ***repeat***  ***if*** *(name = List element)* ***then***  **→** *false*  **u*ntil end of list***  ***→*** *true* |

Tabel 4 Fungsi *mutualNum()*

|  |
| --- |
| **function***mutualNum(List : listOfString) → listOfPair*  { *I.S.* File terdapat isi/ terdefinisi dan tidak kosong }  { *F.S. Return list* yaitu *Pair* yang mempunyai 2 elemen berupa elemen ke-1 nama akun (*unique*) dan elemen ke-2 berupa jumlah munculnya nama akun pada *list*}  **DECLARE**  *Pair : listOfPair*  *Count : int*  **ALGORITHM**  *Pair ← []*  ***for*** *i* ***in*** *list*  *Count ← 0*  ***If*** *(notInList (i, Pair)* ***then***  ***for*** *j* ***in*** *list*  ***if*** *(j=i)* ***then***  *Count ← Count + 1*  **a*ppend*** *([i, Count])* ***to*** *Pair*  **→** *Pair* |

1. Terakhir, mengurutkan *list* dari jumlahnya *mutual* terbanyak hingga terkecil. Untuk implementasi algortimanya adalah sebagai berikut.

Tabel 5 Fungsi *sortingPair()*

|  |
| --- |
| **procedure***sortingPair(****input*** *pair: listOfPair)*  { *I.S.* Pair terdefinisi dan tidak kosong }  { *F.S. Return Pair* yang pada elemen kedua telah terurut mengecil }  **DECLARE**  *Temp : pairOfString*  **ALGORITHM**  ***for*** *i* ***in*** *range (****len****(pair))*  ***for*** *b* ***in*** *range (****len****(pair)-1)*  ***if*** *(pair[j][1] < pair[j+1][1])* ***then***  *Temp ← pair[j]*  *pair[j] ← pair[j+1]*  *pair[j+1] ← Temp* |

Setelah itu dibuatlah penerapan algoritmanya menggunakan bahasa Python sebagai berikut.

Tabel 6 Implementasi Algoritma Pemrograman Python

|  |
| --- |
| 1. **from** typing **import** List, Tuple 2. **import** os 3. **import** time  6. **def** fileHandle(FileName: str) -> List[Tuple[str, str]]: 7. *# { I.S. File terdapat isi/ terdefinisi dan tidak kosong }* 8. *# { F.S. Return list berupa Pair yang mempunyai 2 elemen nama akun yang saling follow antara satu sama lain.}* 10. ***# DECLARE*** 11. List = [] 12. temp = [] 14. *# ALGORITHM* 15. **if os.path.exists(FileName):** 16. **with** open(FileName, 'r') **as** file: 17. **for** line **in** file: 18. temp = line.strip().split() 19. List.append((temp[0], temp[1])) 20. **else:** 21. **print**("Error, File not found!") 22. exit(0) 24. **return** List  27. **def** notInList(name: str, List: List[Tuple[str, str]]) -> bool: 28. *# { I.S. File terdapat isi/ terdefinisi dan tidak kosong, nama terdefinisi }* 29. *# { F.S. Return nilai true bila name tidak pada list dan false bila nama terdapat pada list }* 31. *# ALGORITHM* 32. **for** element **in** List: 33. **if** name == element[0] **or** name == element[1]: 34. **return** False 36. **return** True  39. **def** mutualNum(List: List[str]) -> List[Tuple[str, int]]: 40. ***# { I.S. File terdapat isi/ terdefinisi dan tidak kosong }*** 41. *# { F.S. Return list yaitu Pair yang mempunyai 2 elemen berupa elemen ke-1 nama akun (unique) dan elemen ke-2 berupa jumlah munculnya nama akun pada list}* 43. *# DECLARE* 44. Pair = [] 45. **Count = 0** 47. *# ALGORITHM* 48. **for** i **in** List: 49. Count = 0 50. **if notInList(i, Pair):** 51. **for** j **in** List: 52. **if** j == i: 53. Count += 1 54. Pair.append((i, Count)) 56. **return** Pair  59. **def** sortingPair(pair: List[Tuple[str, int]]) -> List[Tuple[str, int]]: 60. ***# { I.S. Pair terdefinisi dan tidak kosong }*** 61. *# { F.S. Return Pair yang pada elemen kedua telah terurut mengecil }* 63. *# DECLARE* 64. temp = () 66. *# ALGORITHM* 67. **for** i **in** range(len(pair)): 68. **for** j **in** range(len(pair) - 1): 69. **if** pair[j][1] < pair[j+1][1]: 70. **temp = pair[j]** 71. pair[j] = pair[j+1] 72. pair[j+1] = temp 74. **return** pair  77. *# Input* 78. fileName = input("Enter filename : ") 79. List = fileHandle(fileName) 80. **user = input("Enter name of user : ")** 82. start = time.time() 84. *# Searching* 85. **Recom = []** 86. Mutual = [] 88. **for** pair **in** List: 89. *# Checking user in pair* 90. **if (pair[0] == user or pair[1] == user):** 91. *# Condition 1, user inside pair[0]* 92. **if** (pair[0] == user): 93. **if** (notInList(pair[1], Mutual)): 94. Mutual.append(pair[1]) 95. **for *follow*ing in List:** 96. **if** (*follow*ing[0] != user **and** *follow*ing[1] != user): 97. **if** (*follow*ing[0] == pair[1]): 98. Recom.append(*follow*ing[1]) 99. **elif** (*follow*ing[1] == pair[1]): 100. **Recom.append(*follow*ing[0])** 102. *# Condition 2, user inside pair[1]* 103. **elif** (pair[1] == user): 104. **if** (notInList(pair[0], Mutual)): 105. **Mutual.append(pair[0])** 106. **for** *follow*ing **in** List: 107. **if** (*follow*ing[0] != user **and** *follow*ing[1] != user): 108. **if** (*follow*ing[0] == pair[0]): 109. Recom.append(*follow*ing[1]) 110. **elif (*follow*ing[1] == pair[0]):** 111. Recom.append(*follow*ing[0])  114. *# Delete mutual account that already followed by user* 115. **newRecom = [r for r in Recom if r not in Mutual]** 117. *# Output* 118. **print**("**\n**" + user + " Tiktok") 120. **print("\nSuggested accounts : ")** 121. i = 1 122. newRecom = mutualNum(newRecom) 123. sortingPair(newRecom) 124. **for** suggested **in** newRecom: 125. **print(str(i) + ". " + suggested[0] + ", has " +** 126. str(suggested[1]) + " mutual friends") 127. i += 1 129. end = time.time() 130. **print("\n" + str(end - start) + " seconds")** |

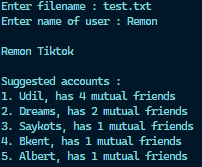
#### Pengujian

Pengujian penelitian ini melalui kode program pada sub-bab terlampir sebelumnya. Dalam menguji kode di atas membutuhkan suatu file format .txt berisi list nama akun-akun yang saling *follow* satu sama lain. Dari pengujian ini dibuatlah file sebagai berikut.

Tabel 7 File Pengujian

|  |
| --- |
| **Udil Zxuan**  **Handy Udil**  **Handy Zxuan**  **Zxuan Remon**  **Handy Remon**  **Remon Branz**  **Zxuan Branz**  **Branz Dreams**  **Saykots Albert**  **Beknt Bagong**  **Bagong Albert**  **Bagong Udil**  **Udil Branz**  **Remon Bagong**  **Handy Handy**  **Zxuan Dreams**  **Dreams Bkent** |

Selanjutnya akan dipilih salah satu akun dari isi file pengujian sebagai user dalam mencari suggested accounts. Pada pengujian pertama ini diambil akun dengan nama Remon sebagai user.



Gambar 4 Pengujian 1

Dari gambar pengujian 1 di atas dapat disimpulkan bahwa terdapat 7 suggested accounts yang diberikan algoritma untuk user secara berurutan mengecil berdasarkan mutual akun user. Berdasarkan file pengujian, dapat terlihat bahwa Remon (akun acuan) *follow* Zxuan, Handy, Bagong, dan Branz. Untuk akun yang mengikutinya :

* Zxuan : Udil, Handy, Brans, dan Dreams
* Handy : Udil, Zxuan, dan Saykots
* Bagong : Bkent, Albert, dan Udil
* Branz : Dreams, Zxuan, dan Udil

Dengan adanya list tersebut, membuat akun-akun yang telah *follow*ed oleh akun user juga berlaku pada akun-akun tertulis lebih dari satu. Sehingga menghasilkan *suggested accounts* :

* Udil : 4 *suggested accounts* (Zxuan, Handy, Bagong, dan Branz)
* Dreams : 2 *suggested accounts* (Zxuan dan Branz)
* Saykots : 1 *suggested accounts* (Handy)
* Bkent : 1 *suggested accounts* (Bagong)
* Albert : 1 *suggested accounts* (Bagong)

Pada pengujian pertama ini membutuhkan waktu selama 0.00403 *seconds* untuk algoritma pencariannya. Durasi tersebut bisa dibilang cukup lama untuk data yang tergolong masih kecil.

Selanjutnya adalah pengujian kedua dengan menggunakan akun user dengan nama Albert, pengujian kedua menghasilkan hasil sebagai berikut.



Gambar 5 Pengujian 2

Dari gambar pengujian kedua di atas, dapat disimpulkan bahwa terdapat 4 *suggested accounts* direkomendasikan algoritma. Bila melihat file masukkan, diketahui akun user dengan nama Albert *follow* Saykots dan Bagong saja. Sehingga untuk akun yang mengikutinya, yaitu :

* Saykots : Handy
* Bagong : Bkent, Udil, dan Remon

Dengan begitu didapatkan *list suggested account sebagai berikut*.

* Handy : 1 *suggested accounts* (Saykots)
* Bkent : 1 *suggested accounts* (Bagong)
* Udil : 1 *suggested accounts* (Bagong)
* Remon : 1 *suggested accounts* (Bagong)

Kemudian untuk pengujian kedua ini dibutuhkan waktu selama 0.005249 *seconds.*

### Kesimpulan

Algoritma *Brute-Force* dapat digunakan pada penerapan bermacam-macam kegiatan sehari-hari, contoh nyatanya adalah fitur *Suggested Accounts* di aplikasi Tiktok ini. Untuk pasangan akun yang saling *follow* berkumpul daftar, yang kemudian akan digunakan algoritma *Brute-Force* dalam menemukan *suggested accounts* yang menghitung banyaknya mengikuti teman yang menggunakan akun *user*. Namun, jika Anda memperhatikan, algoritma ini membutuhkan waktu yang lambat dalam menghasilkan rekomendasi. Maka dari itu, perlunya adaptasi untuk kedepannya agar program berjalan lebih cepat dan lebih efisien.

## DAFTAR PUSTAKA

Fadhilah, R. D. (2022). Penerapan Algoritma Brute-Force pada Instagram Friend Suggestions. 3-5.

Lubis, F. (2021). PENERAPAN MODEL ALGORITMA PEMROGRAMAN DALAM KEHIDUPAN SEHARI-HARI. 3-4.

Munir, Rinaldi. Diambil kembali dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag1.pdf

Munir, Rinaldi. Diambil kembali dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2021-2022/Algoritma-Brute-Force-(2022)-Bag2.pdf

Rico, B. G. (2017). PENERAPAN ALGORITMA APRIORI DALAM POLA PEMINJAMAN BUKU (STUDI KASUS : PERPUSTAKAAN STIKES PRIMA JAMBI). *SCIENTIA JOURNAL (STIKES PRIMA JAMBI)*, 56.

Santo, D. (2021). MODEL AISAS UNTUK MEMETAKAN KEPUTUSAN PEMBELIAN KONSUMEN BERDASARKAN KUALITAS KONTEN TIKTOK #RACUNSHOPEE. *E-Journal Universitas Atma Jaya Yogyakarta*, 46-49.

## LAMPIRAN

Link Github :

<https://github.com/BilhaqAD07/SUGGESTED-ACCOUNTS-BRUTE-FORCE-ALGORITHM.git>

Link Video :

[UTS KAPITA.mkv](https://iteraacid-my.sharepoint.com/:v:/g/personal/bilhaq_120140141_student_itera_ac_id/EQZ11AQIVrZFuYSLbisoorMBByUeJrK4kGnPfBOFMSFE8g?e=fBDAw5)