|  |
| --- |
| **TUGAS (01) PBA** |

Tito Nugroho 155150201111223 TIF-C

Ezra  Reihardian Sihotang 175150207111015 TIF-C

Kevin Aryo Wicaksono 175150201111039 TIF-C

Pretty Natalia Hutapea 175150200111016 TIF-C

**TUGAS**

**PEMROSESAN TEKS (REGULAR EXPRESSION)**

1. **SOAL**
2. Dengan menggunakan Regex, ekstraklah ke dictionary/dict() semua nama (key=authors), tahun (key=year), dan judul (key=title) pada berkas doc-1.txt. Bila tidak ditemukan tahun maka tidak perlu dituliskan. Simpan hasilnya pada berkas a\_judul.json dengan format JSON.
3. Cari 30 kata-kata unik (kecuali stopword) dan case insensitive (case folding) beserta frekuensinya (terurut menurun/descending) yang muncul di dokumen doc-2.txt. Simpan dalam file b\_kataunik.txt. Format penyimpanan :

Kata\tFrekuensi 🡪 \t merupakan karakter tabulasi

Misalnya :

yang 100

di 75

…

1. File doc\_3.srt berisi subtitle suatu film. Lakukan proses cleaning pada file tsb. dengan cara menghilangkan semua isi yang tidak penting dengan Regex, tidak sekedar replace teks biasa (str.replace()), antara lain:

* Nomor baris
* Penanda waktu, dengan format mis. 00:00:00,000 🡪 00:00:00,000
* Tag <i>, <font>, dll. dalam teks
* Newline kosong

Simpan dalam c\_subtitle.txt. Contoh hasil cleaning :

Giliranku.

Aku dapat.

Bisa kau buat…

…sedikit lebih menantang?

Baik. Dengar.

1. **SOURCE CODE & PENJELASAN**
2. Tabel sc dan penjelasan

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80 | import re  import json  fileOpen = open("doc\_1.txt", "r")  text = fileOpen.read()  class Citation:  def \_\_init\_\_(self, title ,author, year):  self.title = title  self.author = author  self.year = year  citations = re.split('\n\n',text)  citationList = []  for citation in citations:  # get year  year = re.findall('\((?:19|20)[0-9][0-9]\)',citation)  authorCitation = re.split('\((?:19|20)[0-9][0-9]\)',citation)  if len(year) == 0:  authorCitation = re.split('[12]\d{3}',citation)  else:  year = year[0][1:-1]  # get author  author = re.findall('[A-Z][A-Z|a-z]+\,?\s[A-Z][a-z]\*\s[A-Z][a-z]\*\.?|[A-Z][a-z]+\,?\s[A-Z][a-z]\*\.?|[A-Z][a-z]{1,}',authorCitation[0])  if len(author) > 1 :  poped\_item = 0  for index in range(len(author)) :  if ',' not in author[index-poped\_item] and '.' not in author[index-poped\_item]:  author.pop(index-poped\_item)  poped\_item +=1  # get title  title = re.findall('\".\*\"', citation)  if len(title) == 0 :  if len(author) ==0 :  title = re.findall('[A-Za-z,]{1,}\s[A-Za-z, ]{0,}',authorCitation[0])[0]  else:  title = re.split(author[0], citation)[1]  if ').' not in title :  title = re.findall('[A-Za-z,]{1,}\s[A-Za-z, ]{0,}[\.]',title)  title = title[0]  else :  title = title.split(').')[1]  title = re.findall('[A-Za-z,]{1,}\s[A-Za-z,: ]{0,}[\.(]',title)  title = title[0]  else:  title = title[0][1:-1]    # Creat Citation  citationList.append(Citation(title, author, year))  #dictionary  dict = []  for citation in citationList:  if len(citation.year)>1:  dict.append(  {  "authors": '; '.join(citation.author),  "title": citation.title,  "year": citation.year  }  )  elif len(citation.year)==0 and len(citation.author)==0:  dict.append(  {  "title": citation.title  }  )  elif len(citation.year)==0:  dict.append(  {  "authors": '; '.join(citation.author),  "title": citation.title  }  )    with open('a\_judul.json', 'w') as f:  json.dump(dict, f) |

Penjelasan :

Untuk mengambil informasi berupa pengarang, tahun dan judul pada format daftar pustaka yang diberikan, menggunakan algoritma sebagai berikut: pertama(baris 18-23), dengan menggunakan regex kita mencari tahun yang terletak diantara tanda kurung “()” kemudian setelah tahunnya kita simpan dalam variable year, teks tersebut di pisah dengan regex tahun tersebut untuk memudahkan ekstraksi selanjutnya. Kedua(baris 26-32), kita mengambil informasi nama penulis dengan 3 macam regex yang dipisahkan oleh tanda “|” yang berarti atau, kemudian dikarenakan dalam satu baris daftar pustaka memungkinkan ada lebih dari satu nama penulis maka dilakukan pengecekan(baris 27-32) selanjutnya dan setelah selesai baru akan dimasukkan dalam variable author. Ketiga(baris 34-49) dilakukan pengambilan informasi berupa judul dengan menggunakan beberapa regex dikarenakan judul dalam daftar pustaka tersebut ditulis dengan format yang berbeda. Dari ketiga langkah tersebut telah didapatkan informasi-informasi yang dibutuhkan. Dalam soal diberikan perintah untuk memasukkan informasi tersebut ke dalam file JSON, maka pertama dibuatlah kelas Citation dengan konstruktor didalamnya(baris 7-11) kemudian informasi yang didapat sebelumnya dimasukkan ke dalam list citationList dengan konstruktor tersebut. Setelah itu membuat dictionary yang kemudian diisi informasi dan key yang sesuai(baris 55-77), dan kemudian dimasukkan ke dalam file JSON(baris 79-80).

1. Tabel sc dan penjelasan

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  29  30  31  32  33  34  35  36  37  38  39  40  41  42 | from collections import Counter  import csv  import re  documentdata = open("doc\_2.txt","r")  words = documentdata.read()  def casefolding(source):  file = source.lower()  return file  def punctuation(text):  for ch in '!"#$%&()\*+,-./:;<=>?@[\\]^\_\'{|}~':  text = text.replace(ch,"")  return text  def removenumber(word):  remove = re.sub(r'\d+','',word)  return remove.split()  hasil\_casefolding = casefolding(words)  hasil\_punctuation = punctuation(hasil\_casefolding)  tokenization = removenumber(hasil\_punctuation)  def filtering(x):  clean = []  stopwords = open("id.stopwords.02.01.2016.txt", "r")  stopwords\_data = stopwords.read()  for word in x:  if word not in stopwords\_data:  clean.append(word)  return clean  filtering\_data = filtering(tokenization)  top\_30\_words = Counter(filtering\_data).most\_common(30)  with open('b\_kataunik.txt', 'w') as f:  csv.writer(f,delimiter=' ').writerows(top\_30\_words) |

Penjelasan :

Algoritma yang digunakan untuk mencari 30 kata-kata unik pada file yang diberikan adalah sebagai berikut: (baris 5-15) berisi method yang bertujuan untuk menormalisasi teks dengan cara menyamakan kapitalisasi huruf (casefolding) serta menghilangkan tanda baca pada teks, (baris 17-19) berisi method untuk normalisasi teks dengan menghilangkang angka yang terdapat pada teks dengan menggunakan regex. Selanjutnya, (baris 27-36) berisi method untuk menghilangkan stopwords yang ada pada teks dengan menggunakan list pada file stopwords. Pada langkah terakhir (baris 40-42), list teks yang telah dinormalisasi dihitung frekuensinya secara menurun dengan menggunakan Counter dan diberikan batas hingga 30 kata dengan frekuensi terbanyak, kemudian hasilnya dimasukkan ke dalam file dengan nama b\_kataunik.txt.

1. Subtitle.py

|  |  |
| --- | --- |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10.  11.  12.  13.  14.  15.  16.  17.  18.  19.  20.  21.  22.  23.  24.  25.  26.  27.  28.  29.  30.  31.  32.  33.  34.  35.  36.  37.  38.  39.  40.  41.  42.  43.  44.  45.  46.  47.  48.  49.  50.  51.  52. | import re  def time(subs):  time\_cleaning = r'(\d{2}.\d{2}.\d{2}.\d{3})\s.\*\s(\d{2}.\d{2}.\d{2}.\d{3})$'  if re.search(time\_cleaning, subs):  return True  return False  def tag(subs):  tag\_cleaning = r'<[^>]\*>'  if re.search(tag\_cleaning, subs):  return True  return False  def pagenumber(subs):  number\_cleaning = r'^\d{1,4}$'  if re.search(number\_cleaning, subs):  return True  return False  def not\_subs(subs):  if time(subs):  return True  if tag(subs):  return True  if pagenumber(subs):  return True  return False  def cleaning\_data(subs):  after\_cleaning = []  subs = subs.split('\n')  for line in subs:  if not\_subs(line) == False:  after\_cleaning.append(line)  return after\_cleaning  def main():  subtitles = open('doc\_3.txt','r').read()  after\_cleaning = cleaning\_data(subtitles)  # remove blank space and blank lines  remove\_blank = [i for i in after\_cleaning if i[:-1]]  file = open('c\_subtitle.txt','w+')  for data in remove\_blank:  file.write(data)  file.write('\n')  file.close()  main() |

Penjelasan :

Algoritma yang digunakan dimulai dengan mencari lokasi string yang match dengan regex. String yang dicari adalah timestamp, tag, page number yang implementasinya terdapat pada baris 3-20. Method not\_subs() bertujuan untuk menyatukan bagian subtitle yang bukan merupakan teks dari subtitle sehingga timestamp, tag, page number bernilai True. Pada baris 31-37 terdapat method cleaning\_data() dimulai dengan membuat list kosong, lalu dilakukan split string dengan enter. Dilanjutkan dengan perulangan berisi kondisi method not\_subs() bernilai False sehingga selain dari yang sesuai dengan kondisi tersebut akan ditambahkan ke dalam list kosong tadi. Terakhir untuk method main() berisi open file doc\_3.txt yang berisi subtitle lalu dilakukan pemanggilan method cleaning\_data() untuk mendapatkan hasil hanya teks dari subtitle saja. Selanjutnya penghapusan blank space dan blank lines dan terakhir memasukkan hasilnya ke file c\_subtitle.txt.

1. **PENJELASAN REGEX**
2. Daftar Pustaka
   1. Year

\((?:19|20)[0-9][0-9]\)

Penjelasan :

Tahun pada daftar pustaka yang diberikan hanya merujuk pada format tahun yang berada di dalam kurung, (?:19|20) digunakan untuk menentukan rangkaian angka yang dianggap tahun merupakan rangkaian angka yang dimulai angka 19 atau 20 dan [0-9][0-9] kemudian diikuti dengan 2 digit angka decimal dengan rentang 0-9

* 1. Author

[A-Z][A-Z|a-z]+\,?\s[A-Z][a-z]\*\s[A-Z][a-z]\*\.?|[A-Z][a-z]+\,?\s[A-Z][a-z]\*\.?|[A-Z][a-z]{1,}

Penjelasan :

Nama pengarang dalam daftar pustaka tersebut memiliki 3 format antara lain [A-Z][A-Z|a-z]+\,?\s[A-Z][a-z]\*\s[A-Z][a-z]\*\.? untuk nama yang terdiri dari 3 kata dan dipisahkan dengan “,” lalu [A-Z][a-z]+\,?\s[A-Z][a-z]\*\.? untuk nama yang terdiri dari 2 kata dan dipisahkan dengan “,” dan akhirnya [A-Z][a-z]{1,} untuk nama yang hanya terdiri dari 1 kata. Masing-masing regex tersebut disambungkan dengan operator or “|” karena tiap regex tersebut dibandingkan bersamaan.

* 1. Title

Penjelasan :

Pada format daftar pustaka yang diberikan terdapat beberapa format penulisan judul yang digunakan, untuk itu digunakan juga regex yang berbeda untuk masing-masing format penulisan judul. \".\*\" untuk format judul paling sederhana yaitu yang berada diantara tanda petik dua (“…”), kemudian dilakukan seleksi lagi untuk format penulisan judul yang beda seperti [A-Za-z,]{1,}\s[A-Za-z,: ]{0,}[\.(] untuk judul yang sebelum tanda kurung ”(“ atau [A-Za-z,]{1,}\s[A-Za-z, ]{0,}[\.] untuk judul yang sebelumnya terdapat tanda kurung “)”

1. Normalisasi
2. Removenumber

(r'\d+','')

Penjelasan:

Normalisasi pada teks doc\_2 menggunakan regex \d yang bertujuan untuk menghilangkang semua karakter angka pada teks yang diberikan. Kemudian digunakan karakter + untuk menghapus angka dengan perulangan digit sebanyak 1 atau lebih.

1. Subtitle
2. Timestamp

(\d{2}.\d{2}.\d{2}.\d{3})\s.\*\s(\d{2}.\d{2}.\d{2}.\d{3})$

Penjelasan :

Timestamp memiliki format 00:00:00,000 🡪 00:00:00,000 karena timestamp terdiri dari angka yang dipisahkan karakter sehingga dapat digunakan \d untuk digit dan ditambahkan {2} atau {3} untuk perulangan digit sebanyak 2 atau 3 kali. Selanjutnya adalah blank space, panah, blank space dengan regex \s.\*\s untuk mendapatkan karakter tersebut. Terakhir ditutup dengan $ untuk penanda akhir kalimat.

1. Tag

<[^>]\*>

Penjelasan :

Tag dimulai dengan karakter < dan di akhiri dengan karakter > yang merupakan case sensitive dan dilanjutkan dengan ^> yang artinya akan match dengan selain karakter >. Karakter \* untuk mengambil semua string yang terdapat tag.

1. Page Number

^\d{1,4}$

Penjelasan :

\d{1,4}untuk mendapatkan string yang memiliki digit dengan perulangan sebanyak 1 hingga 4 kali. Karakter ^ menandakan awal kalimat dan karakter $ menandakan akhir kalimat.

1. **PENANGGUNG JAWAB**

Nomor 1 : Kevin Aryo Wicaksono & Tito

Nomor 2 : Ezra

Nomor 3 : Pretty Natalia Hutapea