PP/018/VI/R5

LABORATORIUM
INFORMATIKAFAKULTAS
TEKNOLOGIINDUSTRI

PETUNJUK PRAKTIKUM

EDISI KURIKULUM OBE

PEMROSESAN BAHASA ALAMI

Penyusun:
Dewi Soyusiawaty

HAK CIPTA

PETUNJUK PRAKTIKUM PEMROSESAN BAHASA ALAMI

Diterbitkan oleh:
Program Studi Informatika Fakultas
Teknologi Industri Universitas Ahmad
Dahlan
Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166

Penulis : Dewi Soyusiawaty
Editor : Laboratorium Informatika, Universitas Ahmad Dahlan **Desain sampul** : Laboratorium Informatika, Universitas Ahmad Dahlan
Tata letak : Laboratorium Informatika, Universitas Ahmad Dahlan

Ukuran/Halaman : 21 x 29,7 cm / 70 halaman

Didistribusikan oleh:

Laboratorium Informatika
Universitas Ahmad Dahlan
Jalan Ring Road Selatan, Tamanan, Banguntapan, Bantul Yogyakarta 55166
Indonesia

KATA PENGANTAR

Alhamdulillah segala puji dan syukur kehadirat Allah SWT, hanya atas rahmat dan hidayah-Nyalah akhirnya petunjuk praktikum ini dapat terselesaikan.

Petunjuk praktikum ini digunakan sebagai bahan untuk kegiatan praktikum mata kuliah Pemrosesan Bahasa Alami sebagai mata kuliah pilihan di Program Studi Informatika.

Petunjuk praktikum ini masih jauh dari sempurna, namun penulis berharap dapat bermanfaatbagi mahasiswa. Saran dan kritik sangat diharapkan untuk perkembangan selanjutnya.

Yogyakarta, 1 Maret 2022
Penyusun

DAFTAR PENYUSUN

Nama Penyusun 1

	Dewi Soyusiawaty

HALAMAN REVISI

Yang bertanda tangan di bawah ini:
Nama : Dewi Soyusiawaty
NIY 60040497
Jabatan : Dosen Pengampu Mata Kuliah **Pemrosesan Bahasa Alami**
Dengan ini menyatakan pelaksanaan Revisi Petunjuk Praktikum **Pemrosesan Bahasa Alami** untuk Program Studi Informatika telah dilaksanakan dengan penjelasan sebagai berikut:

No	Keterangan Revisi	Tanggal Revisi	Nomor Modul
1	Perubahan software yang digunakan ke Python	2017	PP/018/VI/R1
2	Penambahan materi pengantar Python	2018	PP/018/VI/R2
3	Penambahan materi Flask	2019	PP/018/VI/R2
4	Penambahan materi ekspresi regular,terjemahan	2021	PP/018/VI/R2
5	Perbaikan semua materi dari semua pertemuan	2022	PP/018/VI/R2

Yogyakarta, 1 Maret 2022
Penyusun

Dewi Soyusiawaty
NIY. 60040497

HALAMAN PERNYATAAN

Yang bertanda tangan di bawah ini:
Nama : Murein Miksa Mardhia, S.T., M.T.
NIK/NIY : 60160960
Jabatan : Kepala Laboratorium Informatika
Menerangkan dengan sesungguhnya bahwa Petunjuk Praktikum ini telah direview dan akan digunakan untuk pelaksanaan praktikum di Semester Gasal Tahun Akademik 2020/2021 di Laboratorium Praktikum Informatika, Program Studi Informatika, Fakultas Teknologi Industri, Universitas Ahmad Dahlan.

Yogyakarta, 2023

Mengetahui, Ketua Kelompok Keilmuan	Kepala Laboratorium Informatika
Nama Ketua KK NIP/NIY.	**Murein Miksa Mardhia, S.T., M.T.** NIY. 60160960

VISI DAN MISI PRODI INFORMATIKA

VISI

Menjadi Program Studi Informatika yang diakui secara internasional dan unggul dalam bidangInformatika serta berbasis nilai-nilai Islam.

MISI

1. Menjalankan pendidikan sesuai dengan kompetensi bidang Informatika yang diakui nasional dan internasional
2. Meningkatkan penelitian dosen dan mahasiswa dalam bidang Informatika yang kreatif, inovatif dan tepat guna.
3. Meningkatkan kuantitas dan kualitas publikasi ilmiah tingkat nasional dan internasional
4. Melaksanakan dan meningkatkan kegiatan pengabdian masyarakat oleh dosen dan mahasiswa dalam bidang Informatika.
5. Menyelenggarakan aktivitas yang mendukung pengembangan program studi dengan melibatkan dosen dan mahasiswa.
6. Menyelenggarakan kerja sama dengan lembaga tingkat nasional dan internasional.
7. Menciptakan kehidupan Islami di lingkungan program studi.

TATA TERTIB LABORATORIUM INFORMATIKA

DOSEN/KOORDINATOR PRAKTIKUM

1. Dosen harus hadir saat praktikum minimal 15 menit di awal kegiatan praktikum untuk mengisi materi dan menandatangani presensi kehadiran praktikum.
2. Dosen membuat modul praktikum, soal seleksi asisten, pre-test, post-test, dan responsi dengan berkoordinasi dengan asisten dan pengampu mata praktikum.
3. Dosen berkoordinasi dengan koordinator asisten praktikum untuk evaluasi praktikum setiap minggu.
4. Dosen menandatangani surat kontrak asisten praktikum dan koordinator asisten praktikum.
5. Dosen yang tidak hadir pada slot praktikum tertentu tanpa pemberitahuan selama 2 minggu berturut-turut mendapat teguran dari Kepala Laboratorium, apabila masih berlanjut 2 minggu berikutnya maka Kepala Laboratorium berhak mengganti koordinator praktikum pada slot tersebut.

PRAKTIKAN

1. Praktikan harus hadir 15 menit sebelum kegiatan praktikum dimulai, dan dispensasi terlambat 15 menit dengan alasan yang jelas (kecuali asisten menentukan lain dan patokan jam adalah jam yang ada di Laboratorium, terlambat lebih dari 15 menit tidak boleh masuk praktikum & dianggap Inhal).
2. Praktikan yang tidak mengikuti praktikum dengan alasan apapun, wajib mengikuti INHAL, maksimal 4 kali praktikum dan jika lebih dari 4 kali maka praktikum dianggap GAGAL.
3. Praktikan harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
   b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
   d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Praktikan tidak boleh makan dan minum selama kegiatan praktikum berlangsung, harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di dalam laboratorium (tidak boleh membuang sampah sembarangan baik kertas, potongan kertas, bungkus permen baik di lantai karpet maupun di dalam ruang CPU).
5. Praktikan dilarang meninggalkan kegiatan praktikum tanpa seizin Asisten atau Laboran.
6. Praktikan harus meletakkan sepatu dan tas pada rak/loker yang telah disediakan.
7. Selama praktikum dilarang NGENET/NGE-GAME, kecuali mata praktikum yang membutuhkan atau menggunakan fasilitas Internet.
8. Praktikan dilarang melepas kabel jaringan atau kabel power praktikum tanpa sepengetahuan laboran
9. Praktikan harus memiliki FILE Petunjuk praktikum dan digunakan pada saat praktikum dan harus siap sebelum praktikum berlangsung.
10. Praktikan dilarang melakukan kecurangan seperti mencontek atau menyalin pekerjaan praktikan yang lain saat praktikum berlangsung atau post-test yang menjadi tugas praktikum.
11. Praktikan dilarang mengubah setting software/hardware komputer baik menambah atau mengurangi tanpa permintaan asisten atau laboran dan melakukan sesuatu yang dapat merugikan laboratorium atau praktikum lain.
12. Asisten, Koordinator Praktikum, Kepala laboratorium dan Laboran mempunyai hak untuk menegur, memperingatkan bahkan meminta praktikan keluar ruang praktikum apabila dirasa anda

mengganggu praktikan lain atau tidak melaksanakan kegiatan praktikum sebagaimana mestinya dan atau tidak mematuhi aturan lab yang berlaku.

13. Pelanggaran terhadap salah satu atau lebih dari aturan diatas maka Nilai praktikum pada pertemuan tersebut dianggap 0 (NOL) dengan status INHAL.

ASISTEN PRAKTIKUM

1. Asisten harus hadir 15 Menit sebelum praktikum dimulai (konfirmasi ke koordinator bila mengalami keterlambatan atau berhalangan hadir).
2. Asisten yang tidak bisa hadir WAJIB mencari pengganti, dan melaporkan kepada Koordinator Asisten.
3. Asisten harus berpakaian rapi sesuai dengan ketentuan Universitas, sebagai berikut:
   a. Tidak boleh memakai Kaos Oblong, termasuk bila ditutupi Jaket/Jas Almamater (Laki-laki / Perempuan) dan Topi harus Dilepas.
   b. Tidak Boleh memakai Baju ketat, Jilbab Minim dan rambut harus tertutup jilbab secara sempurna, tidak boleh kelihatan di jidat maupun di punggung (khusus Perempuan).
   c. Tidak boleh memakai baju minim, saat duduk pun pinggang harus tertutup rapat (Laki-laki / Perempuan).
   d. Laki-laki tidak boleh memakai gelang, anting-anting ataupun aksesoris Perempuan.
4. Asisten harus menjaga kebersihan, keamanan dan ketertiban selama mengikuti kegiatan praktikum atau selama berada di laboratorium, menegur atau mengingatkan jika ada praktikan yang tidak dapat menjaga kebersihan, ketertiban atau kesopanan.
5. Asisten harus dapat merapikan dan mengamankan presensi praktikum, Kartu Nilai serta tertib dalam memasukan/Input nilai secara Online/Offline.
6. Asisten harus dapat bertindak secara profesional sebagai seorang asisten praktikum dan dapat menjadi teladan bagi praktikan.
7. Asisten harus dapat memberikan penjelasan/pemahaman yang dibutuhkan oleh praktikan berkenaan dengan materi praktikum yang diasisteni sehingga praktikan dapat melaksanakan dan mengerjakan tugas praktikum dengan baik dan jelas.
8. Asisten tidak diperkenankan mengobrol sendiri apalagi sampai membuat gaduh.
9. Asisten dimohon mengkoordinasikan untuk meminta praktikan agar mematikan komputer untuk jadwal terakhir dan sudah dilakukan penilaian terhadap hasil kerja praktikan.
10. Asisten wajib untuk mematikan LCD Projector dan komputer asisten/praktikan apabila tidak digunakan.
11. Asisten tidak diperkenankan menggunakan akses internet selain untuk kegiatan praktikum, seperti Youtube/Game/Medsos/Streaming Film di komputer praktikan.

LAIN-LAIN

1. Pada Saat Responsi Harus menggunakan Baju Kemeja untuk Laki-laki dan Perempuan untuk Praktikan dan Asisten.
2. Ketidakhadiran praktikum dengan alasan apapun dianggap INHAL.
3. Izin praktikum mengikuti aturan izin SIMERU/KULIAH.
4. Yang tidak berkepentingan dengan praktikum dilarang mengganggu praktikan atau membuat keributan/kegaduhan.
5. Penggunaan lab diluar jam praktikum maksimal sampai pukul 21.00 dengan menunjukkan surat ijin dari Kepala Laboratorium Prodi Teknik Informatika.

Yogyakarta, 1 Agustus 2021

Kepala Laboratorium

Murein Miksa Mardhia, S.T., M.T.
NIY. 60160960

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

SKENARIO PRAKTIKUM SECARA DARING

Nama Mata Praktikum : Pemrosesan Bahasa Alami
Jumlah Pertemuan 10

TABEL SKENARIO PRAKTIKUM DARING

Pertemuan ke	Judul Materi	Waktu (Lama praktikum sampai pengumpulan posttest)	Skenario Praktikum dari pemberian pre-test, post-test dan pengumpulannya sertamencantumkan metode yang digunakan misal video, whatsapp group, Google meet atau lainnya
1	Pengenalan Python dan Pengolahan Dasar Teks	90 menit	Pretest dilaksanakan dengan memberikansoal 1 jam sebelum praktikum dimulai dan praktikan mengupload hasil pekerjaan pretest paling lambat sebelum praktikum dimulai. Asisten membuat rekaman kegiatan praktikum dan praktikan menyimakrekaman tersebut. Soal Posttest diberikan 45 menit sebelumpraktikum berakhir dan praktikan mengerjakan sampai batas waktu akhir jam praktikum.
2	Preprocessing	90 menit	
3	Reguler Expression	90 menit	
4	TF-IDF	90 menit	
5	N Grams	90 menit	
6	Parsing	90 menit	
7	Named Entity Recognition	90 menit	
8	Text Summarization	90 menit	
9	Translator	90 menit	
10	Flask	90 menit	
11	Responsi	90 menit	Responsi diberikan dalam bentuk tugasproyek yang dipresentasikan.

PRAKTIKUM 1: PENGENALAN PYTHON DAN PENGOLAHAN DASAR TEKS

Pertemuan ke 1

Total Alokasi Waktu: 90 menit
- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%
- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasardan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas.
CPMK-01	Mampu menjelaskan konsep umum pemrosesan bahasa alami dan ruang lingkuppenerapan aplikasinya.

1.1. DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:
1. Menjelaskan pemrosesan awal terhadap teks
2. Menerapkan pemrosesan awal terhadap teks

1.2. INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-01	Kemampuan mahasiswa dalam menjelaskan konsep umum pemrosesan bahasa alami
CPL-03	CPMK-01	Kemampuan mahasiswa dalam menerapkan konsep umum pemrosesan bahasa alami

1.3. TEORI PENDUKUNG

A. Sejarah Python

Python dikembangkan oleh Guido van Rossum pada tahun 1990 di CWI, Amsterdam sebagi kelanjutan dari bahasa pemrograman ABC. Versi terakhir yang dikeluarkan CWI adalah 1.2. tahun 1995, Guido pindah ke CNRI sambil terus melanjutkan pengembangan Python. Versi terakhir yang dikeluarkan adalah 1.6. Tahun 2000, Guido dan para pengembang inti Python pindah ke

BeOpen.com yang merupakan sebuah perusahaan komersial dan membentuk BeOpen PythonLabs. Python 2.0 dikeluarkan BeOpen. Setelah mengeluarkan Python 2.0, Guido dan beberapa anggota tim PythonLabs pindah ke DigitalCreations.

Saat ini pengembangan Python terus dilakukan oleh sekumpulan pemrogram yang dikoordinir Guido dan Python Software Foundation. Python Software Foundation adalah sebuah organisasi non-profit yang dibentuk sebagai pemegang hak cipta intelektual Python sejak versi 2.1 dan dengan demikian mencegah Python dimiliki oleh perusahaan komersial. Saat ini distribusi Python sudah mencapai versi 2.6.1 dan versi 3.0.

Nama Python dipilih oleh Guido sebagai nama bahasa ciptaannya karena kecintaan guido pada acara televisi Monty Python's Flying Circus. Oleh karena itu seringkali ungkapan-ungkapan khas dari acara tersebut seringkali muncul dalam korespondensi antar pengguna Python.

B. Pengenalan Python

Python merupakan bahasa pemrograman dinamis yang mendukung pemrograman berbasis objek. Python dapat digunakan untuk berbagai keperluan pengembangan perangkat lunak dan dapat berjalan di berbagai platform sistem operasi. Saat ini skrip python dapat dijalankan pada sistem berbasis : Windows, Linux / Unix, Mac OS X, OS/2, Amiga. Python didistribusikan dengan beberapa lisensi yang berbeda dari beberapa versi. Lisensi Python tidak bertentangan baik menurut definisi Open Source maupun General Public License (GPL). Interpreter Python dapat diperoleh diwebsite resminya di http://www.python.org. Python adalah bahasa pemrograman yang mudah dipelajari dan powefull. Python memiliki struktur data tingkat tinggi yang efisien dan merupakan pendekatan yang sederhana tetapi efektif pada pemrograman yang berorientasi objek (Object-Oriented Programming). Syntax elegan dan dynamic typing yang dimiliki oleh python, bersama interpreted nature dari Python, menjadikannya bahas pemrograman yang ideal untuk melakukan 'scripting' dan pengembangan aplikasi yang pesat dalam banyak area pada kebanyakan platform. Dengan kode yang simple dan mudah diimplementasikan, seorang programmer dapat lebih mengutamakan pengembangan aplikasi yang dibuat, bukan malah mencai syntax error. Hanya dengan menuliskan kode print seperti diatas sudah bisa mencetak apapun yang diinginkan didalam tanda kurung (). Dibagian akhir kodepun tidak harus mengakhirnya dengan semicolon.

C. Mengapa Python?

Sisi utama yang membedakan Python dengan bahasa pemrograman lainnya adalah dalam hal aturan penulisan kode program. Bagi para programmer di luar python siap-siap dibingungkan dengan aturan indentasi, tipe data, tuple, dan dictionary. Python memiliki kelebihan tersendiri dibandingkan dengan bahasa lain terutama dalam hal penanganan modul, ini yang membuat beberapa programmer menyukai python. Selain itu python merupakan salah satu produk yang opensource, free, dan multiplatform.

Beberapa fitur yang dimiliki Python adalah:

1) Memiliki kepustakaan yang luas; dalam distribusi Python telah disediakan modul-modul siap pakai untuk berbagai keperluan.

2) Memiliki tata bahasa yang jernih dan mudah dipelajari.

3) Memiliki aturan layout kode sumber yang memudahkan pengecekan, pembacaan kembali dan penulisan ulang kode sumber. berorientasi obyek.

4) Memiliki sistem pengelolaan memori otomatis (garbage collection, seperti java) modular, mudah dikembangkan dengan menciptakan modul-modul baru; modul-modul tersebut dapat dibangun dengan bahasa Python maupun C/C++.

5) Memiliki fasilitas pengumpulan sampah otomatis, seperti halnya pada bahasa pemrograman Java, python memiliki fasilitas pengaturan penggunaan ingatan komputer sehingga para pemrogram tidak perlu melakukan pengaturan ingatan komputer secara langsung. Pada praktikum ini menggunakan software paket Anaconda(Python) yang didalamnya sudah ada paket Spyder (Scientific Python Development Environment), Jupyter lengkap dengan NumPy, Matplotlib, dan SciPy.Pada praktikum kita akan menggunakan Jupyter notebook.

D. Hello Word Python

Syntax bahasa python hampir sama dengan bahasa pemrograman pada umumnyaseperti Java atau php. Di python untuk mencetak cukup gunakan fungsi print (), dimana sesuatu yang akan dicetak harus diletakkan diantara kurung buka dan kurung tutup, bahkan di Python versi 2.x tidak harus menggunakan tanda kurung kurawal, cukup pisahkan denganspasi. Jika ingin mencetak data string langsung, harus memasukan data ke dalam tanda kutip terlebih dahulu.

```
>>> print ("hello word")
hello word
>>> print "hello word python"
hello word python
```

Python Case Sensitivity

Python bersifat case sensitive, artinya huruf besar dan huruf kecil memiliki perbedaan. Sebagai contoh jika menggunakan fungsi print dengan huruf kecil print() akan berhasil.

Lain hal jika menggunakan hurus besar/capital Print() atau PRINT(), akan muncuk pesan error. Aturan ini berlaku untuk nama variabel ataupun fungsi-fungsi lainnya.

KomentarPython

Komentar adalah kode didalam script Python yang tidak dieksekusi atau tidak dijalankan mesin. Untuk menggunakan komentar cukup menulis tanda pagar #, diikuti dengan isi/tulisan komentar

E. Tipe Data

Python mempunyai tipe data yang cukup unik bila dibandingkan dengan bahasa pemrograman yang lain. Berikut tipe data bahasa pemrograman Python :

Tipe Data	Contoh	Penjelasan
Boolean	True atau False	Menyatakan benar (True) yang bernilai 1, atau salah(False) yang bernilai 0.
String	“Hello Word Python”	Menyatakan karakter/kalimat bisa berupa huruf,angka, dll (diapit/didalam tanda “ atau ‘)
Integer	25 atau 1209	Menyatakan bilangan bulat
Float	3.14 atau 0.99	Menyatakan bilangan yang mempunyai koma
Hexadecimal	9a atau 1d3	Menyatakan bilangan dalam format heksa (bilangan berbasis 16)
Complex	1 + 5j	Menyatakan pasangan angka real dan imajiner
List	[‘xyz’, 768, 3.14]	Data untaian yang menyimpan berbagai tipe data tapiisinya bisa diubah-ubah
Tuple	(‘xyz’, 768, 3.14)	Data untaian yang menyimpan berbagai tipe data tapiisinya tidak bisa diubah
Dictionary	{‘nama’ : ‘ani’ ,‘umur’ : 17}	Data untaian yang menyimpan berbagai tipe data berupa pasangan, penunjuk dan nilai

Variable Python

Pada pemrograman Python, variabel mempunyai sifat yang dinamis, artinya variabel Python tidak perlu dideklarasikan tipe data tertentu dan variabel Python dapat diubah saat programdijalankan. Penulisan variabel Python memiliki aturan tertentu, yaitu :

Karakter pertama harus berupa huruf atau garis bawah/underscore _
Karakter selanjutnya dapat berupa huruf, garis bawah/underscore _ atau angka

Karakter pada nama variabel bersifat sensitive (case-sensitif). Artinya huruf kecil dan huruf besar dibedakan. Contoh : Variabel namaDepan dan namadepan adalah variabel yang berbeda.

Untuk memulai membuat variabel di Python cukup dengan menuliskan variable lalu mengisinya dengan suatu nilai dengan cara menambahkan tanda sama dengan = diikuti dengan nilai yang ingin dimasukan.

Kondisi IF di Python

Pengambilan keputusan (kondisi IF) digunakan untuk mengantisipasi kondisi yang terjadi saat jalannya program dan menentukan tindakan apa yang akan diambil sesuai dengan kondisi. Pada Python ada beberapa statement/kondisi diantaranya : IF, ELSE dan ELIF. Kondisi IF digunakan untuk mengeksekusi kode jika kondisi bernilai benar. Jika kondisi bernilai salah maka statement/kondisi IF tidak akan di-eksekusi.

Kondisi IF-ELSE di Python

Pengambilan keputusan (kondisi IF-ELSE) tidak hanya digunakan untuk menentukan tindakan apa yang akan diambil sesuai dengan kondisi, tetapi juga digunakan untuk menentukan tindakan apa yang akan diambil/dijalankan jika kondisi tidak sesuai. Kondisi IF-ELSE adalah kondisi dimana jika pernyataan benar (true) maka kode dalam IF akan dieksekusi, tetapi jika bernilai salah(false) maka akan mengeksekusi kode didalam ELSE.

Kondisi ELIF di Python

Pengambilan keputusan (kondisi IF-ELIF) merupakan lanjutan/percabanagan logika dari “kondisi IF”. Dengan ELIF dapat membuat kode program yang akan menyeleksi beberapa kemungkinan yang bisa terjadi. Hampir sama dengan kondisi “ELSE”, bedanga kondisi “ELIF” bisa banyak dan tidak hanya satu.

WHILE

Pengulangan while mirip dengan sebuah if : mengeksekusi code didalamnya jika kondisi yang diberikan bernilai True. Perbedaannya adalah di while code didalamnya akan terus berulang

terus selama kondisinya benar, agar codenya tidak berjalan terus menerus (inifinite loop), maka harus ada code didalam loop yang membuat suatu_kondisi:mejadi False.

Karakteristik adalah kualitas tertentu atau ciri yang khas dari sesuatu. Berikut karakteristik dokumen text menurut Loretta Auvil dan Duane Searsmith :

5. *Database text* berukuran besar
6. Memliki dimensi yang tinggi, artinya satu kata merupakan satu dimensi
7. Mngandung kumpulan kata yang saling terkait (*frase*) dan antara kumpulan kata satu dengan lain dapat memiliki arti yang berbeda
8. Banyak mengandung kata ataupun arti yang bias (*ambiguity*)
9. Dokumen email merupakan dokumen yang tidak memiliki struktir bahasa yang baku, karena didalamnya terkadang muncul istilah slank. Conoth istilah slank yaitu “r u there?”, “hellllooo bosss, whatzzzzzzz up?”, dan masih banyak lagi.

1.4. HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

5. Komputer
6. Bahasa Pemrograman Python
7. Jupyter Notebook

1.5. PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Jelaskan perbedaan dan kelebihan Python dari Bahasa Pemrograman lain	25
2.	CPL-03	CPMK-01	Sebutkan dan Jelaskan tipe data pada Python	25
3.	CPL-03	CPMK-01	Sebutkan dan Jelaskan Library untuk meload dokumen .csv dan tuliskan script load dokumentnya	25
4.	CPL-03	CPMK-01	Buatlah script Python untuk menampilkan nama anda	25

1.6. LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-01	Selesaikan langkah praktikum bagian Pengenalan Bahasa Python	Hasil praktikum	50

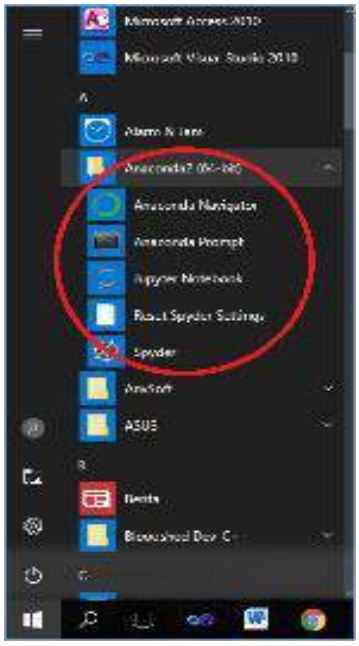
2.	CPL-03	CPMK-01	Selesaikan langkah praktikum bagian Pengolahan Dasar Dokumen	Hasil praktikum	50

Langkah-Langkah Praktikum:

I. Pengenalan Anaconda dan Jupyter

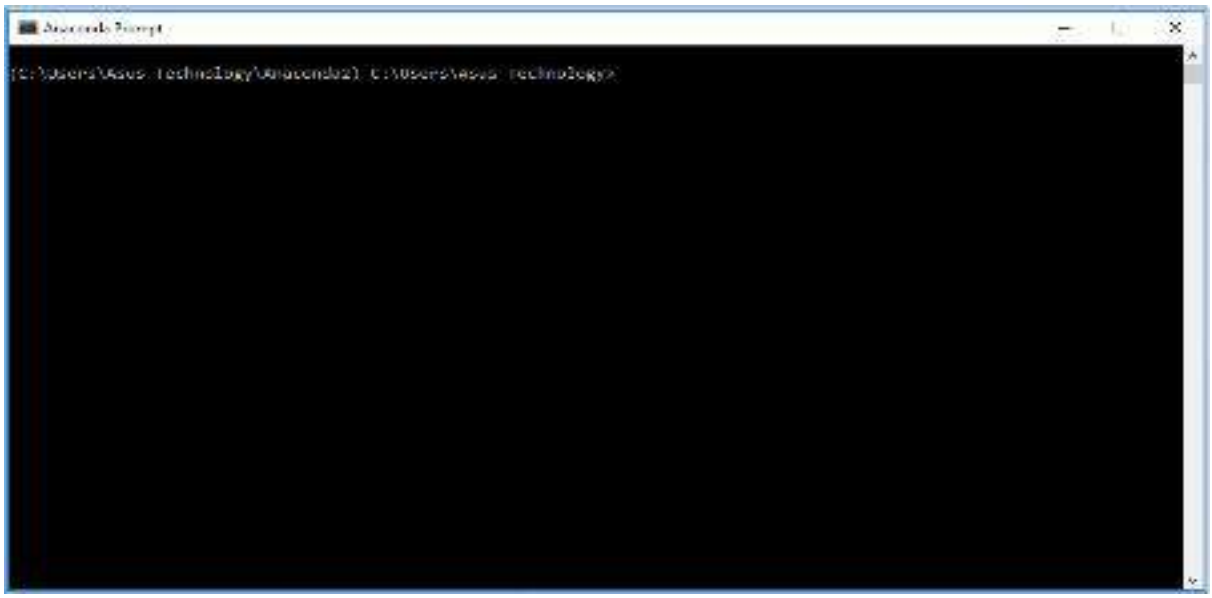
1. Untuk memulai praktikum, dari menu all programs di Windows, carilah program “Anaconda”.

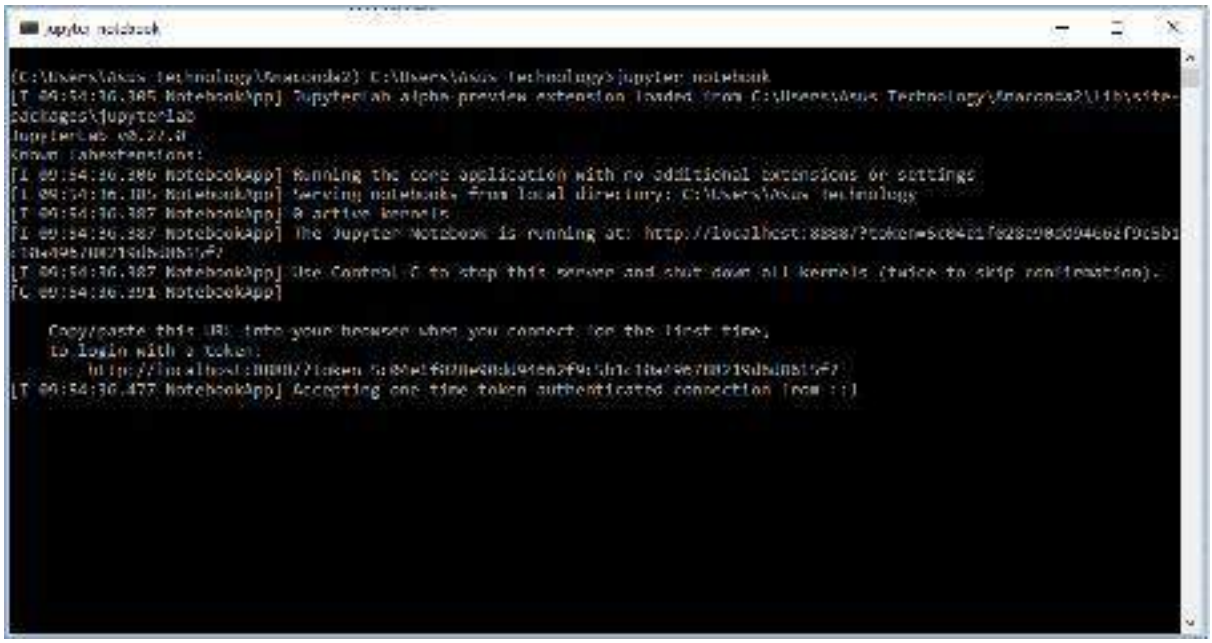
Anda akan mendapati tampilan sebagai berikut:



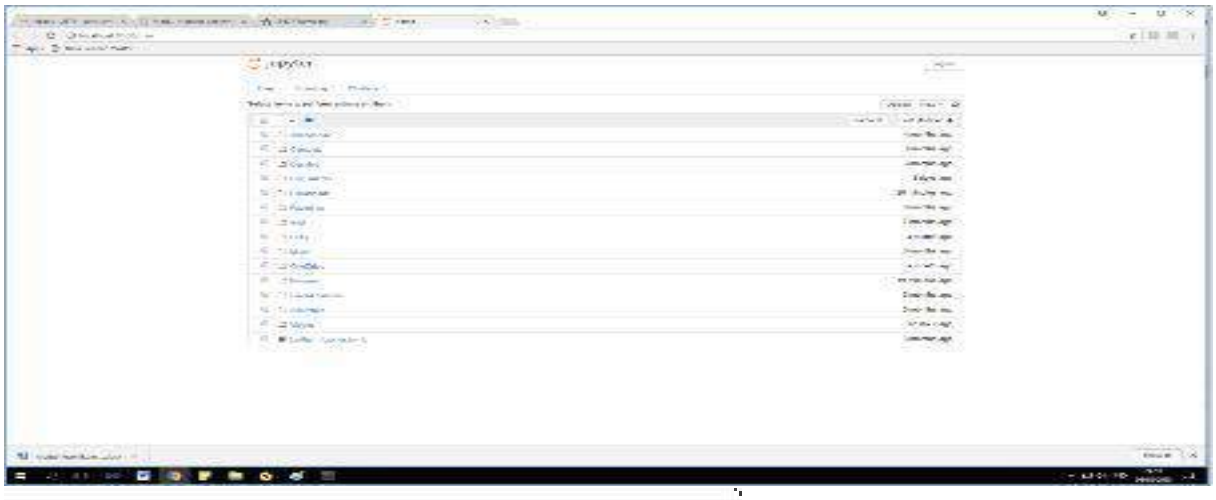
2. Pilihlah menu Anaconda prompt. Anaconda prompt adalah shell/console/command prompt python pada paket Anaconda. Setelah Anda pilih, akan muncul tampilan sebagai berikut:

3. Setelah itu, ketikkan “jupyter notebook” (tanpa tanda petik)untuk menjalankan aplikasi jupyternotebook.



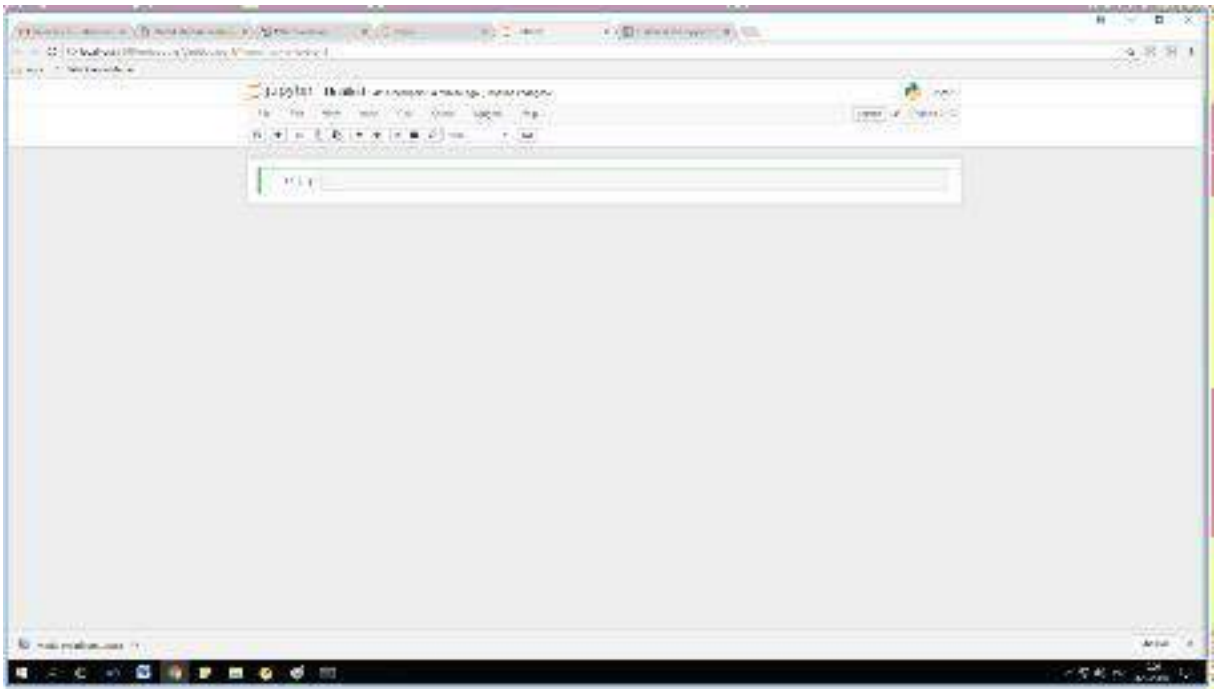


4. Perhatikan, akan muncul pada browser internet, tampilan sebagai berikut:



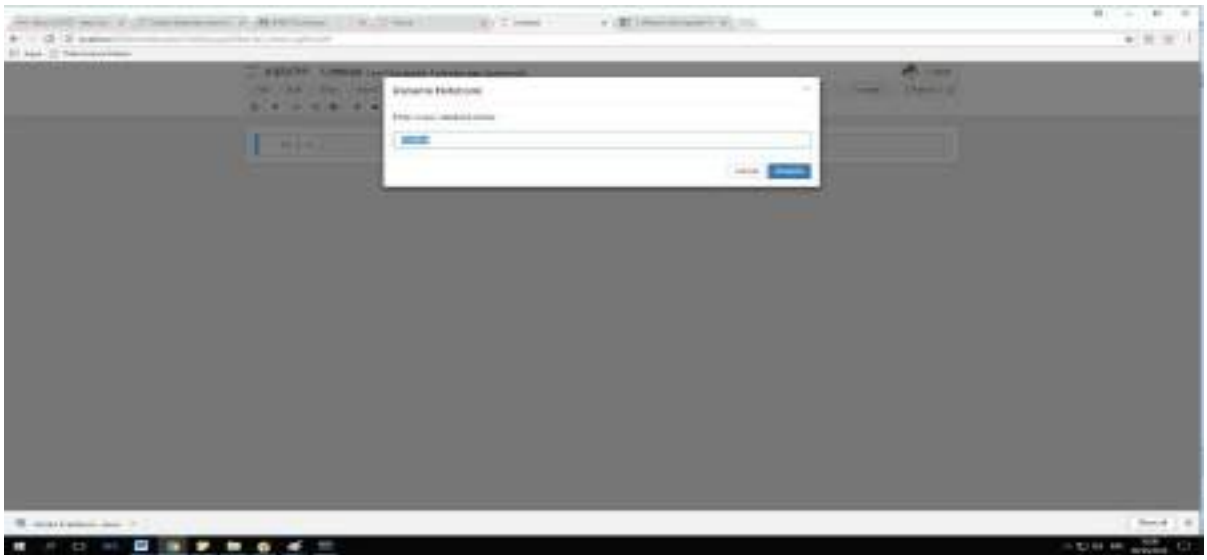
Aplikasi Jupyter Notebook adalah aplikasi client server yang memungkinkan kita mengedit dan menjalankan notebook melalui browser web. Di sini yang dimaksud notebook itu apa? Notebook adalah dokumen yang kita tulis menggunakan Aplikasi Jupyter Notebook, dapat berupakode program, paragraf, gambar, link, dsb.

5. Untuk memulai membuat notebook, pilihlah New notebook (Python 2).



Tampilan di atas adalah tampilan shell/console dimana kita akan mengetikkan kode program python

6. Untuk menyimpan kode program yang sudah kita tuliskan pada shell/console Jupyter dengan nama file tertentu, pilihlah File Rename kemudian akan muncul isian seperti di bawah ini:



Isikan “Praktikum 1” (tanpa tanda petik) pada isian Rename Notebook.

7. Sekarang, kembali kepada shell/console dan kita akan mulai mengetikkan kode program python.

II. Pengenalan Bahasa Python

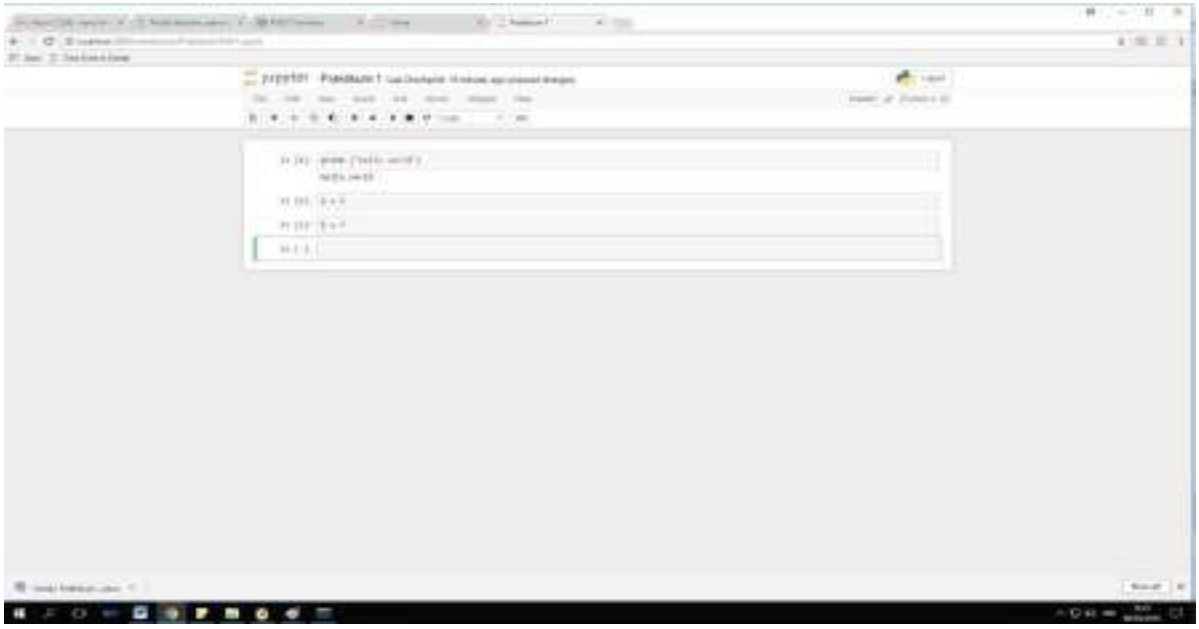
1. Ketikkan print (“hello world”) di console jupyter. Kemudian tekan Shift + Enter . Akan muncul tulisan hello world di di bawah baris program.

2. Selanjutnya buat dua variable seperti di bawah ini:

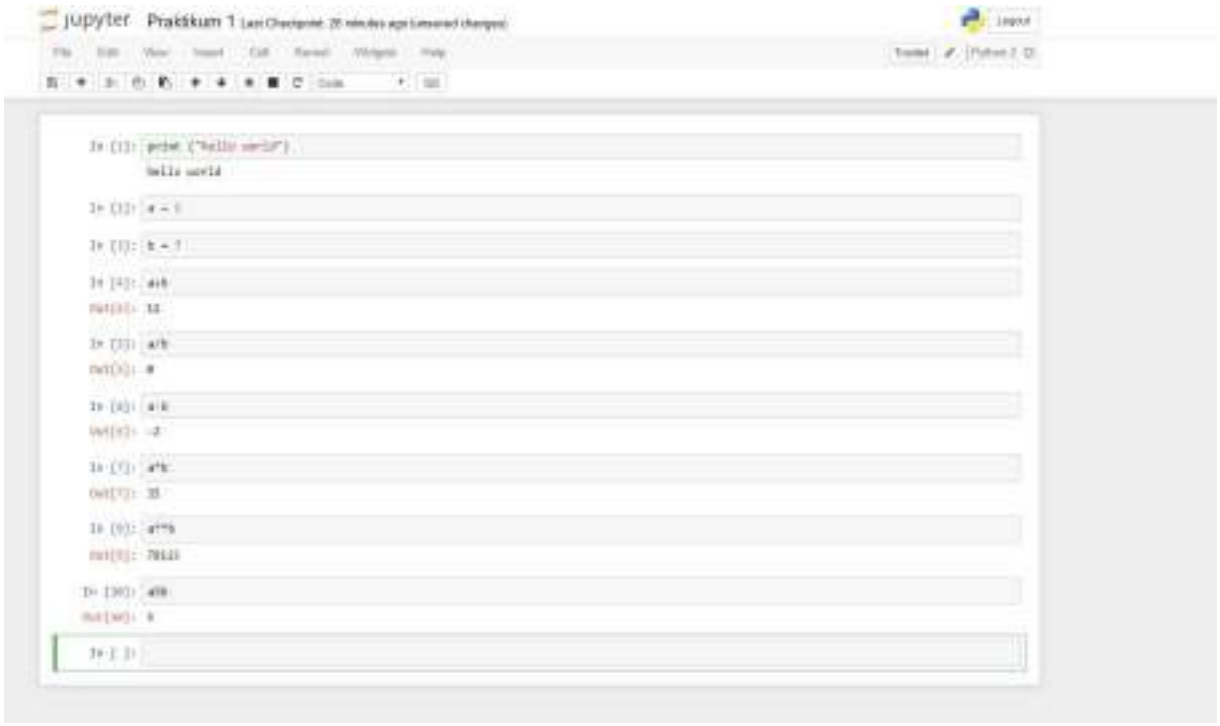
   a =5

   b =7

Perhatikan, pada Python, kita tidak perlu menuliskan tipe dari variabel yang dibuat, cukupmenuliskan nama variabel kemudian isi dari variabel tersebut.

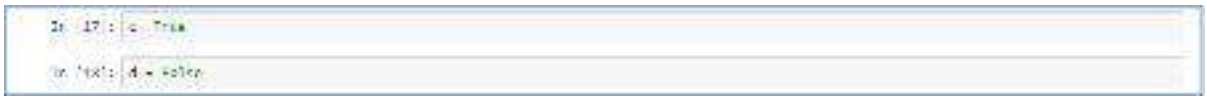


3. Lakukan melakukan operasi-operasi aritmatika pada variable a dan b, mencakup penjumlahan,pengurangan, pembagian, perkalian, pangkat serta modulo.



4. Lakukan operasi-operasi relasional/perbandingan pada variabel a dan b, meliputi <, >, <=, >=, !=, ==, seperti di bawah ini.

5. Buat dua variabel bertipe boolean, c dan d, dimana c = True dan d = False



6. Lakukan operasi logika yang mencakup and, or, dan not, seperti pada tampilan di bawah ini

7. Selanjutnya kita akan membuat kode/script untuk kondisi percabangan if else. Pada langkah sebelumnya kita sudah memiliki variabel a dan b. Kita akan membuat kode untuk kondisi jika a lebih besar dari b maka kita tuliskan “a lebih besar dari b” dan jika kondisi sebaliknya maka akan kita tuliskan “b lebih besar dari a”. Jika a dan b sama besar maka kita tuliskan “a dan b sama”.

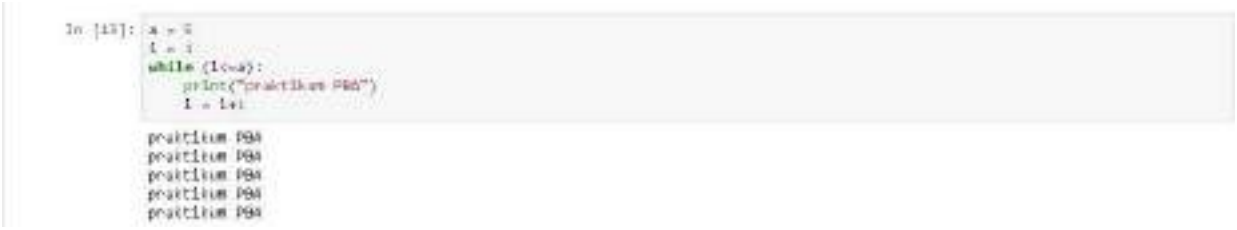
```
  nilai1=2nilai2=3
if(nilai1<nilai2):
    print("nilai 1 lebih kecil dari nilai 2")elif(nilali1>nilai2):
    print("nilai 1 lebih besar dari nilai 2")else:
```

print("nilai 1 sama dengan nilai 2")

Perhatikan, Python sangat sensitif dengan indentasi (tulisan menjorok). Tulisan elif harus lurusdengan if, begitu juga else harus lurus dengan if.

8. Selanjutnya kita akan membuat kode/script untuk perulangan pada Python dengan menggunakan while. Misalnya, kita akan menuliskan “Kuliah Machine Learning” sebanyak 5 kali, dengan memanfaatkan variabel a yang sudah kita buat sebelumnya.

```
a = 5

i = 1

while (i<=a): print("praktikum PBA

  ")i = i+1
```



9.. Berikutnya, cara membuat list pada Python. List ibaratnya seperti array pada bahasa pemrograman yang lain. Akan tetapi, list dapat berisi berbagai data dengan tipe yang berbedasedangkan array hanya bisa berisi data yang tipenya sama. List ditandai dengan kurung [].

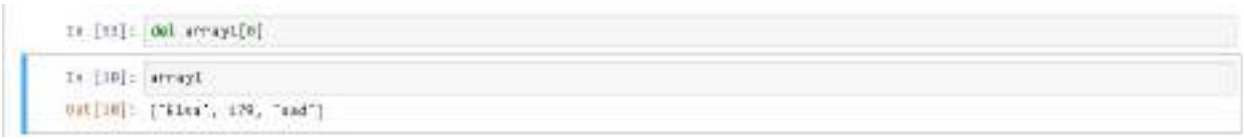
ListL terdiri dari data-data yang bertipe int, float, string. List juga dapat beanggotakan list.

10. Cara pengaksesan elemen List menggunakan index. Index dimulai dari 0. Berikut ini adalah

contoh pengaksesan elemen list dari kiri (dimulai dari 0).

Selain pengaksesan dari kiri, List juga dapat diakses dari sebelah kanan. Dimulai dengan indeks -1, dst.

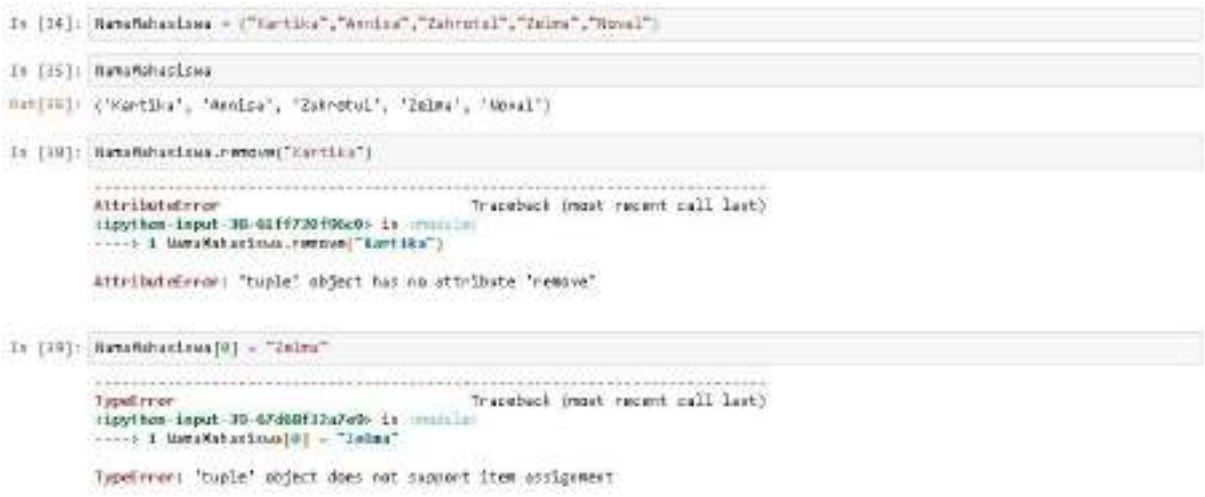
0. Elemen list dapat kita hapus dengan menggunakan perintah del, perhatikan contoh di bawah ini



Untuk menggunakan del, kita harus tahu indeks data yang akan dihapus. Sedangkan jika kita tidaktahu indeks data yang akan dihapus, maka kita gunakan perintah remove

10. List dapat beranggotakan list yang lain. Contohnya kita akan punya listD yang salah satu isinya adalah listL.

12. Selain list, pada Python juga terdapat tipe data lain yaitu Tuple. Berbeda dengan list, isi tuple bersifat immutable yang artinya tidak dapat diubah/digantinilai/dihapus. Contoh kita akan buat Tuple NamaSiswa yang berisi “Kartika”, “Annisa”, “Zahrotul”, “Zelma”, dan “Noval”. Tuple ditandai dengan kurung ()



Perhatikan, ketika perintah remove dipanggil pada Tuple akan ada pesan eror “Tuple object has no attribute ‘remove’ ”. Begitu juga ketika isi Tuple akan diupdate ada pesan eror “Tuple ” obejct does not support item assignment. Hal ini karena Tuple sifatnya immutable.

13. Selain list dan tuple, pada bahasa Python juga memiliki tipe data Dictionary. Dictionary menggunakan beberapa kata kunci untuk mengindeks data. Sehingga pada Dictionary, data tidak diindeks menggunakan angka 0,1,dst tetapi menggunakan kata-kata kunci tertentu. Contohnya kita akan membuat dictionary yang berisi data {'nama': 'azzam', 'alamat': 'malang', 'jurusan': 'fisika', 'nim': 1234}. Dictionary ditandai dengan {}.

14. Dictionary tidak bersifat immutable sehingga kita dapat mengupdate/mengubah/menghapusdata pada dictionary. Penghapusan data pada dictionary hanya menggunakan del, tidak dapat menggunakan remove.

Ingat :

B. Python bersifat case sensitive, artinya huruf kapital dan huruf kecil dianggap berbeda

C. Setiap kali selesai menuliskan kode/script pada console jupyter, tekan Enter + Shift

untuk menjalankan kode/script

III. Pengolahan Dasar Dokumen

Import data dari file Excel

Umumnya data *text* didalam bahasa pemrograman python berbentuk list, dictionary atau tuple. Data *text* tersebut dapat diambil dari DBMS *(Data Base Management System)* atau *file* dengan ekstensi json, xlsx, csv ataupun *file* lain yang sejenis. Berikut dijelaskan cara untuk melakukan *import* data komentar hotel ekstensi xlsx (Microsoft Excel), data tersebut lengkap dengan (Nama Hotel, Alamat Hotel, Komentar, Waktu Komentar, Nama Orang yang Memberikan Komentar) dapat dilihat pada Gambar di bawah ini.

	A	B	C	D	E
1	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Menu breakfast tidak merancerankan hotel bintang 4	19 Desember 2016	Indrawati
2	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Telepon di kamar tidak bisa digunakan, kamar mandi yg terlalu kecil.	12 September 2016	Rahman
3	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Fasilitas yg lumayan lengkap, lokasi yg strategis, staff yg ramah	12 September 2016	Aris
4	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Breakfast banyak pilihan menu, kolam renang ok, cake slice di lobby	13 Agustus 2016	Indri
5	favehotel Premier	Jl. Cihampelas 129 Bandung	Tidak ada tissue dikamar tidur jg tidak disediakan teh celup, kopi dll	12 Februari 2017	Dwi
6	favehotel Premier	Jl. Cihampelas 129 Bandung	Area meraja kolam renang kurang bersih	14 Januari 2017	Joko
7	favehotel Premier	Jl. Cihampelas 129 Bandung	Fasilitas dikamar mandi rusak/copot	08 Januari 2017	Bagus
8	favehotel Premier	Jl. Cihampelas 129 Bandung	Tidak tersedia extra bed	03 Januari 2017	Erin
9	favehotel Premier	Jl. Cihampelas 129 Bandung	Lokasi dekat dengan pusat belanja dan tempat makan	27 Desember 2016	Yessi

Langkah *import* data xlsx kedalam Python List dengan menggunakan jupyter notebook :

1. *Import library* yang akan digunakan.

```
from openpyxl import load_workbook
import pandas as pd
```

- `openpyxl` adalah *library* yang digunakan untuk *read* dan *write* file Excel (xlsx/xlsm/xltx/xltm)
- `pandas` adalah *library data analysis*, digunakan untuk mengolah data secara terstruktur.

2. Inisialisasi *file* excel yang akan di *import*

```
wb = load_workbook(filename = 'komentar hotel.xlsx')
sheet_ranges = wb['Sheet1']

df = pd.DataFrame(sheet_ranges.values)
df.columns = ['Hotel', 'Alamat', 'Komentar', 'Tanggal','User']

df
```

- merupakan nama *function* dari *library* openpyxl yang

`load_workbook`

digunakan untuk melakukan *import* data dari excel (kemudian disimpan dalamvariabel wb)

- `Sheet_ranges` merupakan *variable* yang menampung data dari sheet mana yang akan diambil dalam file excel (pada contoh ini adalah Sheet1).
- `DataFrame` adalah *function* dari *library* pandas yang digunakan untuk melakukan parsing data terstruktur kedalam bentuk kolom dan baris, dengan demikian data yang telah diparsing akan menjadi sebuah *table* yang nampak seperti susunan pada *relational database,* dimana sebuah baris tunggal mewakili sebuah contoh tunggal dan kolom mewakili atribut tertentu.
  (Kemudian dimasukkan ke dalam variabel `df`).
- `Columns` *function* dari *library* pandas yang digunakan untuk memberikan header pada setiap kolom data.
- Hasil dari dataframe `df` dapat dilihat pada Tabel di bawah ini:

	Hotel	Alamat	Komentar	Tanggal	User
0	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Menu breakfast tidak mencerminkan hotel bintang 4	2016-12-19	Indrawati
1	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Telepon di kamar tidak bisa digunakan, kamar m...	2016-09-12	Rahman
2	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Fasilitas yg lumayan lengkap, lokasi yg strate...	2016-09-12	Aris
3	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Breakfast banyak pilihan menu, kolam renang ok...	2016-06-13	Indri
4	favehotel Premier	Jl. Cihampelas 129 Bandung	Tidak ada tissue dikamar tidur jg tidak disedi...	2017-02-12	Dwi
5	favehotel Premier	Jl. Cihampelas 129 Bandung	Area menuju kolam renang kurang bersih	2017-01-14	Joko
6	favehotel Premier	Jl. Cihampelas 129 Bandung	Fasilitas dikamar mandi rusak/copot	2017-01-05	Bagas
7	favehotel Premier	Jl. Cihampelas 129 Bandung	Tidak tersedia extra bed	2017-01-03	Estri
8	favehotel Premier	Jl. Cihampelas 129 Bandung	Lokasi dekat dengan pusat belanja dan tempat ri...	2016-12-27	Yessi

3. Mengambil data pada DataFrame

   Data dapat diambil berdasarkan indeks kolom tertentu dengan mendefinisikan urutanindeks kolom pada *variable* DataFrame (`df`).

   a. Menampilkan data pada kolom komentar.

```
df['Komentar']
```

```
0    Menu breakfast tidak mencerminkan hotel bintang 4
1    Telepon di kamar tidak bisa digunakan, kamar m...
2    Fasilitas yg lumayan lengkap, lokasi yg strate...
3    Breakfast banyak pilihan menu, kolam renang ok...
4    Tidak ada tissue dikamar tidur jg tidak disedi...
5                Area menuju kolam renang kurang bersih
6                Fasilitas dikamar mandi rusak/copot
7                            Tidak tersedia extra bed
8    Lokasi dekat dengan pusat belanja dan tempat m...
Name: Komentar, dtype: object
```

b. Menampilkan data pada kolom komentar dengan bentuk tabel.

```
df[['Komentar']]
```

	Komentar
0	Menu breakfast tidak mencerminkan hotel bintang 4
1	Telepon di kamar tidak bisa digunakan, kamar m...
2	Fasilitas yg lumayan lengkap, lokasi yg strate...
3	Breakfast banyak pilihan menu, kolam renang ok...
4	Tidak ada tissue dikamar tidur jg tidak disedi...
5	Area menuju kolam renang kurang bersih
6	Fasilitas dikamar mandi rusak/copot
7	Tidak tersedia extra bed
8	Lokasi dekat dengan pusat belanja dan tempat m...

c. Menampilkan data dengan jumlah tertentu

```
df[:3]
```

	Hotel	Alamat	Komentar	Tanggal	User
0	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Menu breakfast tidak mencerminkan hotel bintang 4	2016-12-19	Indrawati
1	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Telepon di kamar tidak bisa digunakan, kamar m...	2016-09-12	Rahman
2	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Fasilitas yg lumayan lengkap, lokasi yg strate...	2016-09-12	Ans

d. Menampilkan data secara Ascending atau Descending

```
df[::-1]
```

	Hotel	Alamat	Komentar	Tanggal	User
9	favehotel Premier	Jl. Cihampelas 129 Bandung	Lokasi dekat dengan pusat belanja dan tempat m...	2016-12-27	Yessi
7	favehotel Premier	Jl. Cihampelas 129 Bandung	Tidak tersedia extra bed	2017-01-03	Estri
6	favehotel Premier	Jl. Cihampelas 129 Bandung	Fasilitas dikamar mandi rusak/copot	2017-01-06	Bagas
5	favehotel Premier	Jl. Cihampelas 129 Bandung	Area menuju kolam renang kurang bersih	2017-01-14	Joko
4	favehotel Premier	Jl. Cihampelas 129 Bandung	Tidak ada tissue dikamar tidur jg tidak disedi...	2017-02-12	Dwi
3	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Breakfast banyak pilihan menu, kolam renang ok...	2016-06-13	Indri
2	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Fasilitas yg lumayan lengkap, lokasi yg strate...	2016-09-12	Ans
1	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Telepon di kamar tidak bisa digunakan, kamar m...	2016-09-12	Rahman
0	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Menu breakfast tidak mencerminkan hotel bintang 4	2016-12-19	Indrawati

e. Menampilkan data secara Ascending atau Descending berdasarkan kolom *komentar*.

```
df.sort_values(['Komentar'], ascending=[1])
```

	Hotel	Alamat	Komentar	Tanggal	User
7	favehotel Premier	Jl. Cihampelas 129 Bandung	Tidak tersedia extra bed	2017-01-03	Estri
4	favehotel Premier	Jl. Cihampelas 129 Bandung	Tidak ada tissue dikamar tidur jg tidak disedi...	2017-02-12	Dwi
1	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Telepon di kamar tidak bisa digunakan, kamar m...	2016-09-12	Rahman
0	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Menu breakfast tidak mencerminkan hotel bintang 4	2016-12-19	Indrawati
9	favehotel Premier	Jl. Cihampelas 129 Bandung	Lokasi dekat dengan pusat belanja dan tempat m...	2016-12-27	Yessi
2	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Fasilitas yg lumayan lengkap, lokasi yg strate...	2016-09-12	Ans
6	favehotel Premier	Jl. Cihampelas 129 Bandung	Fasilitas dikamar mandi rusak/copot	2017-01-06	Bagas
3	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Breakfast banyak pilihan menu, kolam renang ok...	2016-06-13	Indri
5	favehotel Premier	Jl. Cihampelas 129 Bandung	Area menuju kolam renang kurang bersih	2017-01-14	Joko

f. Menampilkan data beberapa kolom tertentu

```
df[['Hotel','Komentar']]
```

	Hotel	Komentar
0	Discovery Hotel	Menu breakfast tidak mencerminkan hotel bintang 4
1	Discovery Hotel	Telepon di kamar tidak bisa digunakan, kamar m...
2	Discovery Hotel	Fasilitas yg lumayan lengkap, lokasi yg strate...
3	Discovery Hotel	Breakfast banyak pilihan menu, kolam renang ok...
4	favehotel Premier	Tidak ada tissue dikamar tidur jg tidak disedi...
5	favehotel Premier	Area menuju kolam renang kurang bersih
6	favehotel Premier	Fasilitas dikamar mandi rusak/copot
7	favehotel Premier	Tidak tersedia extra bed
8	favehotel Premier	Lokasi dekat dengan pusat belanja dan tempat m...

g. Memasukkan data beberapa kolom tertentu ke dalam list python

```
df[['Hotel','Komentar']].values.tolist()
```

```
[[u'Discovery Hotel', u'Menu breakfast tidak mencerminkan hotel bintang 4'],
 [u'Discovery Hotel',
  u'Telepon di kamar tidak bisa digunakan, kamar mandi yg terlalu kecil, kurangnya petanda arah ke fasilitas hotel..'],
 [u'Discovery Hotel',
  u'Fasilitas yg lumayan lengkap, lokasi yg strategis, staff yg ramah'],
 [u'Discovery Hotel',
  u'Breakfast banyak pilihan menu, kolam renang ok, cake slice di lobby enak & kalau malam disc 30% \u263a'],
 [u'favehotel Premier',
  u'Tidak ada tissue dikamar tidur jg tidak disediakan teh celup, kopi dan gula selain air mineral. Untuk sekelas dan sebar
ga itu sangat tidak sepadan sama sekali.'],
 [u'favehotel Premier', u'Area menuju kolam renang kurang bersih'],
 [u'favehotel Premier', u'Fasilitas dikamar mandi rusak/copot'],
 [u'favehotel Premier', u'Tidak tersedia extra bed'],
 [u'favehotel Premier', u'Lokasi dekat dengan pusat belanja dan tempat makan']]
```

h. Menampilkan data pada baris/*row* tertentu

```
df.iloc[1]
```

```
Hotel                                    Discovery Hotel
Alamat                          Jl. Lodan Timur No. 7 Jakarta
Komentar    Telepon di kamar tidak bisa digunakan, kamar m...
Tanggal                                 2016-09-12 00:00:00
User                                                 Rahman
Name: 1, dtype: object
```

```
df.iloc[[1]]
```

	Hotel	Alamat	Komentar	Tanggal	User
1	Discovery Hotel	Jl. Lodan Timur No. 7 Jakarta	Telepon di kamar tidak bisa digunakan, kamar m...	2016-09-12	Rahman

i. Menampilkan spesifik field (data pada kolom dan baris tertentu)

```
df[['3']].iloc[[1]]
```

	Tanggal
1	2016-09-12

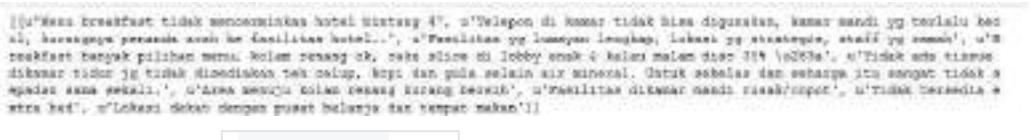
```
df.loc[1,'Tanggal']
```

```
Timestamp('2016-09-12 00:00:00')
```

j. Memasukkan data pada kolom *komentar* ke dalam list python

```
list_komentar = []
list_komentar.append(df['Komentar'].values.tolist())

print list_komentar
```

- Deklarasi list array dengan nama `list_komentar`
- Menambahkan isi `dataframe` indeks ke-2 ke dalam list `list_komentar` .
- Hasil print dari `list_komentar`



- Menampilkan `list_komentar` satu persatu.

```
for komentar in list_komentar:
    for komen in komentar:
        print komen
```

```
Menu breakfast tidak mencerminkan hotel bintang 4
Telepon di kamar tidak bisa digunakan, kamar mandi yg terlalu kecil, kurangnya pengarahan arah ke fasilitas hotel..
Fasilitas yg lumayan lengkap, lokasi yg strategis, staff yg ramah
Breakfast banyak pilihan menu, kolam renang ok, cake slice di lobby enak & kalau malam disc 30% @
tidak ada tissue dikamar tidur yg tidak disediakan teh celup, kopi dan gula selain air mineral. untuk sekelas dan seharga i
tu sangat tidak sepadan sama sekali.
Area menuju kolam renang kurang bersih
Fasilitas dikamar mandi rusak/copot
Tidak tersedia extra bed
lokasi dekat dengan pusat belanja dan tempat makan
```

k. Memasukkan data pada kolom *komentar* ke dalam list_komentar 1 untuk di hitung jumlah kata yang terdapat dalam komentar

```
list_komentar1=[]
list_komentar1 = df['komentar'].values.tolist() #mengubah komentar dalam bentuk list
print(list_komentar1)
```

['Sudah sampai gan. Mantap barangnya...', 'kualitas bgs. dpt bonus lagi dr penjual", 'Barang rapih packingnya n bagus. Cara pengirimannya yang agak lama', 'Makasih barang sudah sampai dengan selamat sampai tujuan.', 'pihak penjual memperhatikan d an mengikuti perkembangan proses pengiriman dengan seksama,pelayanan yang baik dan aman menurut pemantauan saya,terimakasi h', 'respon cepat, pengiriman super cepat, packing aman dan rapi.', 'puas belanja disini. mudahan lancar terus ya gan.', 'p acking rapih, barang sesuai dgn yg diiklankan. nice and re**list_komentar**ni nya mantapp yaa aman banget. seller juga mau bertanggung jawab kalo ni barang mengalami kerusakan. keren lah barang juga ori thank you yaaa', 'barangnya mulu s, pengirimannya juga cepat. makasih ya sis", 'Produk diterima sesuai deskripsi. berfungsi dengan baik']

l. Mengubah list_komentar1 yang berisikan komentar menjadi lowercase

```
for i in range(len(list_komentar1)):
    list_komentar1[i] = list_komentar1[i].lower()
print(list_komentar1)
```

['sudah sampai gan. mantap barangnya...', 'kualitas bgs. dpt bonus lagi dr penjual", 'barang rapih packingnya n bagus. cara pengirimannya yang agak lama', 'makasih barang sudah sampai dengan selamat sampai tujuan.', 'pihak penjual memperhatikan d an mengikuti perkembangan proses pengiriman dengan seksama,pelayanan yang baik dan aman menurut pemantauan saya,terimakasi h', 'respon cepat, pengiriman super cepat, packing aman dan rapi.', 'puas belanja disini. mudahan lancar terus ya gan.', 'p acking rapih, barang sesuai dgn yg diiklankan. nice and recommended seller", 'packing nya mantapp yaa aman banget. seller juga mau bertanggung jawab kalo ni barang mengalami kerusakan. keren lah barang juga ori thank you yaaa', 'barangnya mulu s, pengirimannya juga cepat. makasih ya sis", 'produk diterima sesuai deskripsi. berfungsi dengan baik']

m. Memasuukan library python untuk menghitung term pada kalimat dengan CountVectorizer

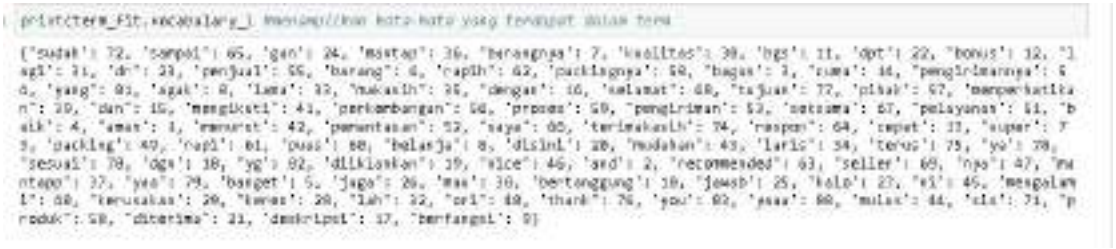
n. Menghitung jumlah kata

```
menghitung jumlah kata tanpa duplikat
CV = CountVectorizer()
term_fit = CV.fit(list_komentar)

print (len(term_fit.vocabulary_))
```

```
84
```

o. Menampilkan kata-kata yang terdapat dalam term

```
print(term_fit.vocabulary_) menampilkan kata-kata yang terdapat dalam term
```



2.1. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Dari dataset toko.xlsx: a. Tampilkan data secara ascending berdasarkan nama produk dan secara descending berdasakan komentar. b. Tampilkan data pada baris ke 3 c. Tampilkan data dengan jumlah 5 d. Tampilkan 2 kolom tertentu e. Tampilkan banyak kata tanpa duplikat kata	75
2.	CPL-03	CPMK-01	Beri komentar tentang penjelasan setiap kode yang dijalankan	25

2.2. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
					Total Nilai	

PRAKTIKUM 2: PREPROCESSING

Pertemuan ke 2

Total Alokasi Waktu: 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasardan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas.
CPMK-01	Mampu menjelaskan konsep umum pemrosesan bahasa alami dan ruang lingkuppenerapan aplikasinya.

2.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu menjelaskan dan menerapkanpemrosesan awal terhadap teks

2.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-01	CPMK-01	Kemampuan mahasiswa dalam menerapkan library NLTK untuk menjalankan pemrosesan awal terhadap teks dengan tokenisasi, casefolding, stop words dan stemming baik pada teks bahasa Inggris atau bahasa Indonesia.

2.3 TEORI PENDUKUNG

Tokenization adalah proses memecah teks yang diberikan dalam pemrosesan bahasa alami menjadi unit terkecil dalam kalimat yang disebut token. Tanda baca, kata, dan angka dapat dianggap sebagai token. Umumnya tokenisasi berguna saat kita ingin mengekstrak makna dari sebuah teks. Contohnya dalam sebuah kalimat kita ingin mendeteksi kata benda dan kata kerja yang ada dalam kalimat tersebut, atau kita ingin mencari nama orang yang dimention dalam sebuah kalimat. Maka dengan melakukan pemecahan 1 per 1, sistem dapat melakukan pengecekan 1 per 1 juga terhadap tiap-tiap teks yang ada pada kalimat. Contoh tokenisasi dengan kalimat “saya suka belajar pba” di pecah menjadi “saya, suka, belajar, pba”.

Menggunakan Distribusi Frekuensi kita dapat menemukan berapa kali sebuah kata digunakan dalam sebuah kalimat, atau kita dapat menghitung kata dalam teks kita menggunakan Distribusi Frekuensi NLP. Contoh “saya belajar tokenisasi dan saya belajar distibusi frekuensi”. Distribusi frekuensi dari saya adalah 2.

Case folding adalah salah satu bentuk text preprocessing yang paling sederhana dan efektif meskipun sering diabaikan. Tujuan dari case folding untuk mengubah semua huruf

dalam dokumen menjadi huruf kecil. Hanya huruf 'a' sampai 'z' yang diterima. Karakter selain huruf dihilangkan dan dianggap delimiter. Ada bebrapa tahapan untuk case folding yaitu salah satunya lowercase dan dan penghapusan karakter tidak penting.

Stopword adalah kata umum yang biasanya muncul dalam jumlah besar dan dianggap tidak memiliki makna. Contoh stopword dalam bahasa Indonesia adalah "yang", "dan", "di", "dari", dll. Makna di balik penggunaan stopword yaitu dengan menghapus kata-kata yang memiliki informasi rendah dari sebuah teks, kita dapat fokus pada kata-kata penting sebagai gantinya.

Teks Normalisasi adalah suatu cara yang digunakan untuk mengurangi bentuk-bentuk infleksional dan kadang-kadang bentuk-bentuk kata yang terkait secara derivatif menjadi bentuk dasar yang sama. Ada 2 teknik untuk melakukan teks normalisasi yaitu lemmatizer dan stemmer.

Stemmer adah proses mengubah suatu kata menjadi kata dasar tanpa mengetahui konteks dari kata tersebut seperti memotong ujung kata-kata. Sedangkan Lemmatizer adalah proses mengubah suatu kata menjadi kata dasar dengan mengetahui konteks dari kata tersebut

2.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer
2. Bahasa Pemrograman Python
3. Jupyter Notebook

2.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Jelaskan apa yang dimaksud dengan tokenisasi	20
2.	CPL-03	CPMK-01	Jelaskan apa yang dimaksud dengan distribusifrekuensi	10
3.	CPL-03	CPMK-01	Jelaskan macam macam cara untuk melakukan tokenisasi dan distribusi frekuensi dalam python	30
4.	CPL-03	CPMK-01	Jelaskan yang dimaksud dengan case Folding dan stopword? Dan berikan masing2 contohnya	20
5.	CPL-03	CPMK-01	Jelaskan perbedaan lemmatizer dan stemmer danberikan contohnya	20

2.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-01	Selesaikan langkah praktikum tokenisasi	Hasil praktikum	
2.	CPL-03	CPMK-01	Selesaikan langkah praktikum case folding dan stopwords	Hasil praktikum	

3.	CPL-03	CPMK-01	Selesaikan langkah praktikum case folding dan stopwords	Hasil praktikum	
4.	CPL-03	CPMK-01	Selesaikan langkah praktikum case folding dan stopwords	Hasil praktikum	

Langkah-Langkah Praktikum:

I. TOKENISASI

1. Menginstal library NLTK dengan *!pip intall nltk* di dalam jupyter notebook

```
In [ ]:  1 !pip install nltk
```

2. Import nltk yang sudah diinstall dan kemudian download package yang dibutuhkan.

```
In [1]:  1 import nltk
         2 nltk.download()
```

```
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
```

3. Load suatu text dari Gutenberg dan menggunakan nltk.corpus

```
In [2]:  1 from nltk.corpus import gutenberg
```

```
In [3]:  1 text = gutenberg.raw('blake-poems.txt')
```

```
In [4]:  1 print(text)
```

```
[Poems by William Blake 1789]

SONGS OF INNOCENCE AND OF EXPERIENCE
and THE BOOK of Thel

 SONGS OF INNOCENCE

 INTRODUCTION

Piping down the valleys wild,
  Piping songs of pleasant glee,
On a cloud I saw a child,
  And he laughing said to me:

"Pipe a song about a Lamb!"
  So I piped with merry cheer.
```

4. Lakukan word tokenization dan sentence tokenization dengan terlebih dahulu mengimport function yang dibutuhkan.

```
In [5]:  1 from nltk import word_tokenize, sent_tokenize
```

5. Lakukan tokenisasi pada kata dan kalimat.

```
In [6]:  1 word_token = word_tokenize(text)
```

```
In [7]:  1 sent_token = sent_tokenize(text)
```

6. Lihat Panjang dari kata dan kalimat menggunakan function len.

```
In [10]:  1 len(word_token)
Out[10]: 8239

In [11]:  1 len(sent_token)
Out[11]: 355
```

7. Kemudian menghitung Frekuensi Distibusi atau *Frequency Distribution* dengan terlebih dahulu import function yang dibutuhkan yaitu FreqDist.

```
In [12]:  1 from nltk import FreqDist
```

8. Gunakan function yang sudah diimport tadi untuk melakukan frekuensi distribusi dan lihat 15 kata yang sering muncul dalam text menggunakan fuction most_common

```
In [13]:  1 fqdist = FreqDist(text)
          2 fqdist.most_common(20)
Out[13]: [(' ', 7444),
          ('e', 3638),
          ('t', 2832),
          ('a', 1881),
          ('h', 1869),
          ('n', 1850),
          ('o', 1711),
          ('r', 1622),
          ('i', 1587),
          ('s', 1577),
          ('\n', 1441),
          ('l', 1193),
          ('d', 1171),
          (',', 685),
          ('m', 664),
          ('y', 660),
          ('u', 648),
          ('g', 589),
          ('w', 568),
          ('f', 511)]
```

9. Melakukan filtering pada text dengan mengambil kata yang lebih dari 6 huruf.

```
In [14]: long_words = [w for w in word_token if len(w)>6]
         long_words
         'SHEPHERD',
         'Shepherd',
         'evening',
         'innocent',
         'watching',
         'Shepherd',
         'ECHOING',
         'welcome',
         'skylark',
         'cheerful',
         'echoing',
         'Sitting',
         'youth-time',
         'echoing',
         'descend',
         'mothers',
         'sisters',
         'brothers',
         'darkening',
```

10. Dan lakukan filtering lagi dengan menambah kondisi yang frequency distributionya lebih dari 6.

```
In [21]: fqdist = FreqDist(long_words)
         info_words = [w for w in long_words if fqdist[w] > 6]

         set(info_words)
Out[21]: {'another', 'children', 'delight', 'flowers', 'morning', 'through', 'weeping'}
```

II. CASE FOLDING DAN STOP WORDS

1. Import library yang dibutuhkan

```
In [1]: from nltk.corpus import gutenberg, stopwords
        from nltk import word_tokenize, FreqDist
```

2. Load data text yang digunakan pada pertemuan sebelumnya

```
In [2]: text = gutenberg.raw('blake-poems.txt')
```

3. Seperti biasa sebelumnya lakukan tokenisasi dan frequensi distribusi dan tampilkan 20 kata yang sering muncul.

```
In [3]: words = word_tokenize(text)
        fqdist = FreqDist(text)
        fqdist.most_common(20)
Out[3]: [(' ', 7444),
         ('e', 5656),
         ('t', 3852),
         ('a', 3801),
         ('h', 3665),
         ('n', 3650),
         ('o', 3744),
         ('r', 3622),
         ('i', 3507),
         ('s', 3477),
         ('\n', 3441),
         ('l', 2205),
         ('d', 2171),
         (',', 489),
         ('m', 884),
         ('y', 660),
         ('w', 608),
         ('g', 597),
         ('u', 562),
         ('f', 511)]
```

4. Untuk lowercasing gunakan function lower yang sudah ada dan kemudian tampilkan.

```
In [4]:
# lowercasing

clean_word = [w for w in word_tokenize(text.lower())]

word_lower = FreqDist(clean_word).most_common(20)
word_lower
```

```
Out[4]: [(',', 695),
 ('the', 458),
 ('and', 348),
 ('.', 331),
 ('of', 146),
 ('in', 141),
 ('i', 130),
 ('a', 127),
 ('to', 111),
 (':', 99),
 ('my', 85),
 ('!', 76),
 ('I', 68),
 ('with', 66),
 ('?', 65),
 ('his', 57),
 ('he', 56),
 ('is', 51),
 ('...', 50),
 ("'s", 46)]
```

5. Untuk menghapus karakter yang tidak penting gunakan alphanumeric atau function isalpha.

```
In [5]:
# lowercasing and alphabet

clean_word = [w for w in word_tokenize(text.lower()) if w.isalpha()]
FreqDist(clean_word).most_common(20)
```

```
Out[5]: [('the', 458),
 ('and', 348),
 ('of', 146),
 ('in', 141),
 ('i', 130),
 ('a', 127),
 ('to', 111),
 ('my', 85),
 ('with', 66),
 ('his', 57),
 ('he', 56),
 ('is', 51),
 ('little', 45),
 ('on', 44),
 ('they', 44),
 ('not', 43),
 ('thee', 42),
 ('that', 39),
 ('all', 38),
 ('but', 38)]
```

6. Dan untuk stopword gunakan function stopword juga, dan dikarenakan textnya berbahasa inggris maka kita menggunakan stopword Bahasa inggris dan tampilkan hasilnya.

```
In [6]:
# lowercasing and alphabet stopwords

clean_word = [w for w in word_tokenize(text.lower()) if w.isalpha() and w not in stopwords.words('english')]
FreqDist(clean_word).most_common(20)
```

```
Out[6]: [('little', 45),
 ('thee', 42),
 ('like', 35),
 ('thou', 33),
 ('thy', 31),
 ('love', 29),
 ('sweet', 26),
 ('night', 26),
 ('joy', 25),
 ('many', 24),
 ('away', 24),
 ('father', 22),
 ('sleep', 21),
 ('happy', 19),
 ('shall', 19),
 ('day', 18),
 ('mother', 18),
 ('child', 18),
 ('every', 17),
 ('never', 17)]
```

7. Kemudian untuk text Bahasa Indonesia disini sudah disediakan text Bahasa Indonesia silahkan di download di classroom.

8. Load data text yang sudah di download.

```
In [7]:
1 with open('sirkuit lombok.txt') as f:
2     teks_indo = f.read()
3     print(teks_indo)
```

9. Lakukan tokenisasi.

```
In [8]: 1 FreqDist(word_tokenize(teks_indo)).most_common(20)

Out[8]: [('di', 24),
         (',', 23),
         ('.', 23),
         ('sirkuit', 12),
         ('yang', 12),
         ('Sirkuit', 10),
         ('Mandalika', 8),
         ('MotoGP', 7),
         ('penandangan', 7),
         ('juga', 7),
         ('I', 6),
         ('ini', 6),
         ('untuk', 5),
         ('?', 4),
         ('akan', 4),
         ('``', 4),
         ('Aida', 4),
         ('area', 4),
         ('berada', 4),
         ('Terindah', 3)]
```

10. Untuk stopword Bahasa Indonesia kita menggunakan library PySastrawi dan silahkan di install librarynya.

```
In [9]: 1 !pip install Pysastrawi

Requirement already satisfied: PySastrawi in c:\users\micro star\anaconda3\lib\site-packages (1.1.0)
```

11. Import library yang dibutuhkan untuk stopword.

```
In [10]: 1 #LowerCasing, Alphanum, dan Stopword
         2
         3 from Sastrawi.StopWordRemover.StopWordRemoverFactory import StopWordRemoverFactory
```

12. Lihat list kata apa saja yang termasuk kata yang tidak penting didalam Bahasa Indonesia.

```
In [11]: 1 stopwordObj = StopWordRemoverFactory()
         2 stopwords_sastrawi_indo = stopwordObj.get_stop_words()
         3 stopwords_sastrawi_indo

         'apaan',
         'apabila',
         'apakah',
         'apalagi',
         'apatah',
         'arti',
         'artinya',
         'asal',
         'asalkan',
         'atas',
         'atau',
         'ataukah',
         'ataupun',
         'awal',
         'awalnya',
         'b',
         'bagai',
         'bagaikan',
         'bagaimana',
         'bagaimanakah',
```

13. Kemudian lakukan stopword dengan kode seperti di bawah ini

```
In [12]: clean_word = [w for w in word_tokenize(teks_inds.lower()) if w.isalpha() and w not in stopwords_sastrawi_indo]
         FreqDist(clean_word).most_common(20)

Out[12]: [('sirkuit', 29),
          ('pemerdasgan', 9),
          ('mandalika', 8),
          ('motogp', 7),
          ('aksa', 4),
          ('area', 4),
          ('terindah', 3),
          ('dunia', 3),
          ('laut', 3),
          ('alam', 3),
          ('hijau', 3),
          ('lintasan', 3),
          ('penulis', 2),
          ('masya', 2),
          ('faely', 2),
          ('ruhulessin', 2),
          ('editor', 2),
          ('hilda', 2),
          ('alexander', 2),
          ('international', 2)]
```

III. Lemmatizer dan stemmer

1. Import library yang dibutuhkan seperti dibawah ini

```
In [3]: #import Stemmer Class

        from nltk.stem.porter import PorterStemmer
        from nltk.stem.lancaster import LancasterStemmer
        from nltk.stem.snowball import SnowballStemmer

        #import Lemmatizer class

        from nltk.stem import WordNetLemmatizer
```

2. Buatlah kata input bebas berbahasa inggris. Contoh ['writing','wrote','written','friends','friendship','friendly','connect','connected','connection','playing','played']

```
In [5]: input_words = ['writing','wrote','written','friends','friendship','friendly','connect','connected','connection','playing','p
```

3. Inisiasi suatu objek stemmer dan lemmatizer

```
In [6]: #inisiasi suatu objek stemmer  dan lemmatizer

        portstemmer_obj = PorterStemmer()
        lancasterstemmer_obj = LancasterStemmer()
        snowballstemmer_obj = SnowballStemmer('english')

        wordnet_lemmatizer = WordNetLemmatizer()
```

4. Import library pandas

```
In [7]: import pandas as pd
```

5. Lakukan normalisai dengan menggunakan Teknik stemmer dan lemmatizer

```
In [8]: port_result = [portstemmer_obj.stem(w) for w in input_words]
        lancaster_result = [lancasterstemmer_obj.stem(w) for w in input_words]
        snowball_result = [snowballstemmer_obj.stem(w) for w in input_words]

        lemmatize_result_V = [wordnet_lemmatizer.lemmatize(w, pos='v') for w in input_words]
        lemmatize_result_N = [wordnet_lemmatizer.lemmatize(w, pos='n') for w in input_words]
```

6. Kemudan dengan pandas tampilkan hasilnya mengguanakan dataFrame.

```
In [9]:
pd.DataFrame({
    'input_word' : input_words,
    'port' : port_result,
    'lancaster' : lancaster_result,
    'snowball' :snowball_result,
    'lemmatize_Verb' : lemmatize_result_V,
    'lemmatize_Noun' : lemmatize_result_N,
})
```

Out[9]:

	input_word	port	lancaster	snowball	lemmatize_Verb	lemmatize_Noun
0	writing	write	writ	write	write	writing
1	wrote	wrote	writ	wrote	write	wrote
2	written	written	writ	written	write	written
3	friends	friend	friend	friend	friends	friend
4	friendship	friendship	friend	friendship	friendship	friendship
5	friendly	friendli	friend	friend	friendly	friendly
6	connect	connect	connect	connect	connect	connect
7	connected	connect	connect	connect	connect	connected
8	connection	connect	connect	connect	connection	connection
9	playing	play	play	play	play	playing
10	played	play	play	play	play	played

2.7 POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Lakukan tokenisasi dan frekuensi distribusi pada dataset yang telah diberikan oleh asisten praktikum.	20
2.	CPL-03	CPMK-01	Tampilkan 25 kata yang sering muncul yang memiliki kata lebih dari 5 huruf pada dataset tersebut.	10
3.	CPL-03	CPMK-01	Lakukan case folding dan stopword.	20
4.	CPL-03	CPMK-01	Tampilakn 20 kata yang serig mucul	10
5.	CPL-03	CPMK-01	Lakukan stemming	20
6.	CPL-03	CPMK-01	Jelaskan hasil akhir yang didapat	20

2.8 HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		

3.	Post-Test	CPL-03	CPMK-01	50%		
UNIVERSITAS AHMAD DAHLAN					**Total Nilai**	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 3: REGULER EXPRESSION

Pertemuan ke 3

Total Alokasi Waktu: 90 menit
- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%
- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematikadasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-02	Mampu menjelaskan metode yang digunakan untuk penyelesaian task: languagemodeling, POS Tagging, parsing, representasi semantik, dan aplikasi PBA.

3.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:
1. Menjelaskan ekspresi reguler pada pemrosesan teks
2. Mengimplementasikan ekspresi reguler pada pemrosesan teks

3.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-01	CPMK-01	Kemampuan mahasiswa dalam menerapkan library re dan spacyuntuk menjalankan ekspresi regular.

3.3 TEORI PENDUKUNG

3.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:
1. Komputer.
2. Bahasa pemrograman Python
3. JupyterNotebook

3.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

3.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

Langkah-Langkah Praktikum:

A. Ekstraksi email

1. Install *library* re

```
import re
```

2. Inisiasi suatu kalimat berisi email

```
kalimat = "Hello from shubhamg199630@gmail.com to priya@yahoo.com about the meeting @2PM"
```

3. Pola regex untuk ekstraksi email

```
ekstraksi = re.findall(r'[\w\.-]+@[\w\.-]+\.\w+', kalimat)
ekstraksi
```

```
['shubhamg199630@gmail.com', 'priya@yahoo.com']
```

B. Ekstraksi Kata Kunci

1. *Install library* spacy

```
!pip install -U spacy
```

2. Buka *anaconda prompt* dan masuk ke direktori tempat menyimpan *project* kalian. Download model bahasa inggris dengan beberapa perintah berikut

```
(base) E:\python\pba>python -m spacy download en_core_web_md
Collecting en-core-web-md==3.0.0 from https://github.com/explosion/sp
n_core_web_md-3.0.0-py3-none-any.whl#egg=en_core_web_md==3.0.0
  Downloading https://github.com/explosion/spacy-models/releases/down
none-any.whl (47.1MB)
```

```
(base) E:\python\pba>python -m spacy download en_core_web_lg
Collecting en-core-web-lg==3.0.0 from https://github.com/explosion/
n_core_web_lg-3.0.0-py3-none-any.whl#egg=en_core_web_lg==3.0.0
  Downloading https://github.com/explosion/spacy-models/releases/do
none-any.whl (778.8MB)
    100% |████████████████████████████████| 778.8MB 26kB/s
Requirement already satisfied: spacy<3.1.0,>=3.0.0 in c:\users\muja
=3.0.0) (3.0.5)
```

3. *Import* beberapa *library* yang diperlukan

```
import spacy
```

```
from collections import Counter
from string import punctuation
```

4. *Load* model bahasa inggris yang sudah diunggah

```
nlp = spacy.load("en_core_web_lg")
```

```
import en_core_web_lg
nlp = en_core_web_lg.load()
```

5. Buat fungsi untuk mencari kata kunci

```
def get_hotwords(text):
    result = []
    pos_tag = ['PROPN', 'ADJ', 'NOUN']
    doc = nlp(text.lower())
    for token in doc:
        if(token.text in nlp.Defaults.stop_words or token.text in punctuation):
            continue
        if(token.pos_ in pos_tag):
            result.append(token.text)

    return result
```

6. Tampilkan 5 kata kunci dari suatu teks dengan *hashtags (#)*

```
output = get_hotwords('''Welcome to Medium! Medium is a publishing platform where people can read important, insightful storie
hashtags = [('#' + x[0]) for x in Counter(output).most_common(5)]
print(' '.join(hashtags))
```

```
#medium #welcome #publishing #platform #people
```

3.7 POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

3.8 HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 4: PEMBOBOTAN

Pertemuan ke 4

Total Alokasi Waktu: 90 menit
- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%
- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas
CPMK-02	Mampu menjelaskan metode yang digunakan untuk penyelesaian task: language modeling, POS Tagging, parsing, representasi semantik, dan aplikasi PBA.

4.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:
1. Menjelaskan Menjelaskan dan mengimplementasikan pembobotan dokumen
2. Mengimplementasikan pembobotan dokumen

4.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-02	Kemampuan mahasiswa dalam mengimplementasikan pembobotan dokumen dengan tepat.

4.3 TEORI PENDUKUNG

A. TF (*Term Frequency*)

TF (*Term Frequency*) adalah nilai frekuensi kemunculan token atau kata dalam sebuah dokumen. Semakin besar jumlah kemunculan suatu term (TF tinggi) dalam dokumen, semakin besar pula bobotnya atau akan memberikan nilai kesesuaian yang semakin besar. Pada Term Frequency (TF), terdapat beberapa jenis formula yang dapat digunakan:

1. **TF biner (binary TF)**, hanya memperhatikan apakah suatu kata atau term ada atau tidak dalam dokumen, jika ada diberi nilai satu (1), jika tidak diberi nilai nol(0).
2. **TF murni (raw TF)**, nilai TF diberikan berdasarkan jumlah kemunculan suatu term di dokumen. Contohnya, jika muncul lima (5) kali maka kata tersebut akan bernilai lima (5).
3. **TF logaritmik**, hal ini untuk menghindari dominansi dokumen yang mengandung sedikit term dalam query, namun mempunyai frekuensi yang tinggi.
4. **TF normalisasi**, menggunakan perbandingan antara frekuensi sebuah term dengan nilai maksimum dari keseluruhan atau kumpulan frekuensi term yang ada pada suatu dokumen.

Misalnya terdapat 3 dokumen/kalimat seperti berikut.

1) "Hotel sangat bersih, pelayan hotel juga ramah"
2) "Fasilitas hotel yg lumayan lengkap"
3) "Fasilitas di kamar mandi banyak yg rusak"

Nilai *term frequency* dapat dilihat pada tabel dibawah ini:

NO.	Term	Term Frequency		
		D1	D2	D3
1	banyak	0	0	1
2	bersih	1	0	0
3	di	0	0	1
4	fasilitas	0	1	1
5	hotel	2	1	0
6	juga	1	0	0
7	kamar	0	0	1
8	lengkap	0	1	0
9	lumayan	0	1	0
10	mandi	0	0	1
11	pelayan	1	0	0
12	ramah	1	0	0
13	rusak	0	0	1

14	sangat	1	0	0
15	yg	0	1	1

B. IDF (*Inverse Document Frequency*)

IDF (Inverse Document Frequency) merupakan sebuah perhitungan dari bagaimana term didistribusikan secara luas pada koleksi dokumen yang bersangkutan. IDF menunjukkan hubungan ketersediaan sebuah term dalam seluruh dokumen. Semakin sedikit jumlah dokumen yang mengandung term yang dimaksud, maka nilai IDF semakin besar.

Inverse Document Frequency (IDF) dihitung dengan menggunakan formula sebagai berikut:

$$IDF_j = \log(D/df_j)$$ (Rumus IDF)

Dimana D adalah jumlah semua dokumen dalam koleksi sedangkan dfj adalah jumlah dokumen yang mengandung term (tj).

4.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Bahasa pemrograman Python
3. JupyterNotebook

4.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-02	Jelaskan apa yang dimaksud dengan TF dan IDF?	20
2.	CPL-03	CPMK-02	Apa Fungsi TF-IDF pada Pemrosesan Bahasa Alami?	30
3.	CPL-03	CPMK-02	Hitung TF-IDF secara manual. dari dokumen di bawah ini: (Tiap dokumen dipisahkan oleh tanda baca titik) Dokumen = “Kebun binatang itu terlihat sangat luas. Berbagai macam binatang terdapat di dalamnya. Binatang-binatang di dalam kebun tersebut terlihat gemuk.”	50

4.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-02	Selesaikan langkah praktikum	Hasil praktikum	

Langkah-Langkah Praktikum:

Menggunakan *library* Sklearn.

Langkah untuk menghitung nilai term frequency :

1. *Import library* yang akan digunakan.

```
from sklearn.feature_extraction.text import CountVectorizer, TfidfTransformer
```

- Modul `sklearn.feature_extraction` digunakan untuk ekstraksi fitur yang

  umumnya digunakan pada machine learning.

- [illegible] *convert* dokumen kedalam token matrik.
- `TfidfTransformer` mengubah sejumlah token matriks kedalam bentuknormalisasi tf atau tf-idf.

2. Dokumen komentar hotel dalam bentuk list `list_dokumen`

```
list_dokumen= ['hotel sangat bersih, pelayan hotel juga ramah',
               'fasilitas hotel yg lumayan lengkap',
               'fasilitas di kamar mandi banyak yg rusak']
```

3. Menghitung jumlah *unique term/vocabulary* dalam

```
CV = CountVectorizer()
term_fit=CV.fit(list_dokumen)

print len(term_fit.vocabulary_)
```

```
15
```

- *function* **fit()** digunakan untuk mempelajari kamus kosa kata semua token

  dalam dokumen.
- `vocabulary_` digunakan untuk pemetaan fitur / *term*

4. Menampilkan uniq *term/vocabulary* sesuai dengan *index*

```
print term_fit.vocabulary_
```

```
{u'lumayan': 8, u'bersih': 1, u'kamar': 6, u'ramah': 11, u'di': 2, u'sang
at': 13, u'banyak': 0, u'hotel': 4, u'yg': 14, u'fasilitas': 3, u'lengkap
': 7, u'rusak': 12, u'juga': 5, u'mandi': 9, u'pelayan': 10}
```

5. Menampilkan uniq *term/vocabulary*

```
print term_fit.get_feature_names()
```

```
[u'banyak', u'bersih', u'di', u'fasilitas', u'hotel', u'juga', u'kamar',
u'lengkap', u'lumayan', u'mandi', u'pelayan', u'ramah', u'rusak', u'sanga
t', u'yg']
```

- `get_feature_names()` digunakan untuk pemetaan Array dari indeks fitur

  integer untuk menampilkan nama.

6. Menampilkan *term frequency* secara keseluruhan

```
term_frequency_all = term_fit.transform(list_dokumen)
print term_frequency_all
```

```
  (0, 1)        1
  (0, 4)        2
  (0, 5)        1
  (0, 10)       1
  (0, 11)       1
  (0, 13)       1
  (1, 3)        1
  (1, 4)        1
  (1, 7)        1
  (1, 8)        1
  (1, 14)       1
  (2, 0)        1
  (2, 2)        1
  (2, 3)        1
  (2, 6)        1
  (2, 9)        1
  (2, 12)       1
  (2, 14)       1
```

- `Transform()` digunakan untuk *Transform vocabulary* documents kedalam

  matrik *term frequency*.

7. Menampilkan *term frequency* untuk dokumen/kalimat tertentu.
   - Mengambil dokumen dengan indeks ke-0, kemudian dimasukkan ke dalam variabel komentar_tf.

```
komentar_tf = list_dokumen[0]
print komentar_tf
```

```
hotel sangat bersih, pelayan hotel juga ramah
```

   - Menampilkan *term frequency* pada dokumen yang disimpan dalam variabel komentar_tf.

```
term_frequency = term_fit.transform([komentar_tf])
print term_frequency
```

```
  (0, 1)        1
  (0, 4)        2
  (0, 5)        1
  (0, 10)       1
  (0, 11)       1
  (0, 13)       1
```

- Menampilkan *term* atau *feature* tertentu berdasarkan indeks *termvocabulary*.

```
print term_fit.get_feature_names()[4]
```

```
hotel
```

*TF-IDF (*Term Frequency dan Inverse Document Frequency*)*

TF-IDF adalah perkalian dari formula perhitungan TF dengan formula IDF.

1. Menghitung nilai TF-IDF untuk dokumen/kalimat tertentu.

```
komentar_term = term_fit.transform(list_dokumen)
tfidf_transformer = TfidfTransformer().fit(komentar_term)
tfidf = tfidf_transformer.transform(term_frequency)
print tfidf
```

```
  (0, 13)       0.369772375024
  (0, 11)       0.369772375024
  (0, 10)       0.369772375024
  (0, 5)        0.369772375024
  (0, 4)        0.562442844513
  (0, 1)        0.369772375024
```

- `TfidfTransformer()` digunakan untuk *Transform vocabulary* documents kedalam matrik *term frequency* atau TFIDF.

2. Menghitung nilai IDF untuk dokumen/kalimat tertentu.

```
print tfidf_transformer.idf_
```

```
[ 1.69314718  1.69314718  1.69314718  1.28768207  1.28768207  1.69314718
  1.69314718  1.69314718  1.69314718  1.69314718  1.69314718  1.69314718
  1.69314718  1.69314718  1.28768207]
```

- `.idf_` digunakan untuk menampilkan atribut IDF

3. Menghitung nilai IDF untuk *term* tertentu.

```
print tfidf_transformer.idf_[term_fit.vocabulary_['fasilitas']]
print tfidf_transformer.idf_[term_fit.vocabulary_['kamar']]
```

```
1.28768207245
1.69314718056
```

4. Menghitung nilai TF-IDF untuk seluruh dokumen.

```
komentar_tfidf = tfidf_transformer.transform(komentar_term)
print komentar_tfidf
```

```
  (0, 13)       0.369772375024
  (0, 11)       0.369772375024
  (0, 10)       0.369772375024
  (0, 5)        0.369772375024
  (0, 4)        0.562442844513
  (0, 1)        0.369772375024
  (1, 14)       0.393511204094
  (1, 8)        0.517419943932
  (1, 7)        0.517419943932
  (1, 4)        0.393511204094
  (1, 3)        0.393511204094
  (2, 14)       0.306504216242
  (2, 12)       0.403016210804
  (2, 9)        0.403016210804
  (2, 6)        0.403016210804
  (2, 3)        0.306504216242
  (2, 2)        0.403016210804
  (2, 0)        0.403016210804
```

4.7 POST TEST

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-02	Dengan menggunakan dokumen dari soal tugas pretest, hitung TF-IDF nya mengikuti code dari modul	50
2.	CPL-03	CPMK-02	Berikan penjelasan tiap line code	30
3.	CPL-03	CPMK-02	Jelaskan perbedaan hasil dari jawaban manual dengan code	20

4.8 HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 5: N GRAMS

Pertemuan ke 5

Total Alokasi Waktu: 90 menit
- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%
- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematikadasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas.
CPMK-02	Mampu menjelaskan metode yang digunakan untuk penyelesaian task: languagemodeling, POS Tagging, parsing, representasi semantik, dan aplikasi PBA.

5.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:
1. Menjelaskan NGrams pada kata dan kalimat untuk unigram, bigram dst.
2. Mengimplementasikan NGrams pada kata dan kalimat untuk unigram, bigram dst.

5.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-02	Kemampuan mahasiswa dalam menerapkan library yang ada diPython untuk melakukan N Grams.

5.3 TEORI PENDUKUNG

Language model yang sudah dikenal dan banyak digunakan oleh peneliti adalah N-grams. N-grams adalah kumpulan dari item sejumlah n yang disusun secara berurutan dari text atau speech. Representasi N-grams dapat diketahui sebagai berikut :

1. N = 1 disebut dengan Unigram
2. N = 2 disebut dengan Bigram
3. N = 3 disebut dengan Trigram
4. Dst

Contoh N-grams dalam kalimat

Kalimat = “Andi suka bermain sepakbola di lapangan Senayan”

1. Unigram (N=1)

   andi, suka, bermain, sepakbola, di, lapangan, senayan

2. Bigram (N=2)

   andi suka, suka bermain, bermain sepakbola, sepakbola di, di lapangan, lapangan senayan

3. Trigram (N=3)

   andi suka bermain, suka bermain sepakbola, bermain sepakbola di, sepakbola di lapangan, di lapangan senayan

Contoh N-grams dalam kata

Kata = “pemerintah”

1. Unigram (N=1) : p, e, m, e, r, i, n, t, a, h
2. Bigram (N=2) : pe, em, me, er, ri, in, nt, ta, ah
3. Trigram (N=3) : pem, eme, mer, eri, rin, int, nta, tah

5.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Bahasa pemrograman Python
3. JupyterNotebook

5.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-02	Apa itu N-GRAM?	20
2.	CPL-03	CPMK-02	Tulislah Unigram dan Bigram dari kalimat berikut! “Kemampuan bahasa inggris sangat populer dalam dunia kerja”	40
3.	CPL-03	CPMK-02	Buatlah Unigram dan bigram dari setiap kata/hurufpada nama lengkap kalian sendiri!	40

5.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen	Skor

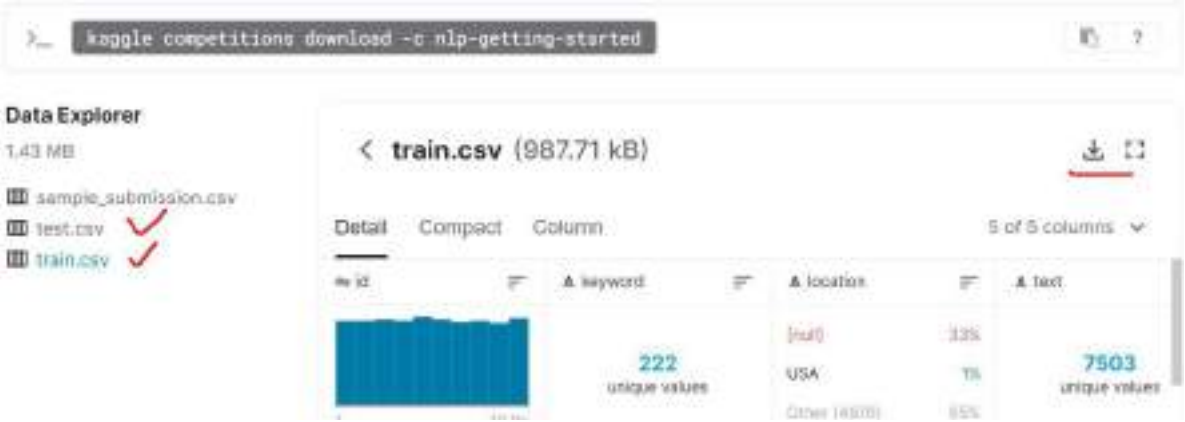
				Pendukung	
1.	CPL-03	CPMK-02	Selesaikan langkah praktikum Dataset Unigram dan Bigram	Hasil langkah praktikum	70
2.	CPL-03	CPMK-02	Selesaikan langkah praktikum kata dan kalimat	Hasil langkah praktikum	30

Langkah-Langkah Praktikum:

A. N-Gram Dataset:

Import dataset train.csv dan test.csv pada jupyterNotebook letakkan pada tempat / folder yangsama dengan notebook anda:

File dataset dapat diunduh pada link berikut:
https://www.kaggle.com/c/nlp-getting-started/data?select=train.csv



Buka Notebook baru dan ketikan perintah berikut pada kotak dialog:

```
pip install nb-black
```

Import dependencies yang akan digunakan pada pemrosesan N-gram

```
from IPython.core.debugger import set_trace

%reload_ext nb_black

import pandas as pd
import numpy as np
import os
import matplotlib.pyplot as plt
import time

plt.style.use(style="seaborn")
%matplotlib inline
```

Jalankan perintah untuk memproses train dan test dataset

```
train = pd.read_csv("train.csv")
test = pd.read_csv("test.csv")
```

Cetak hasil tamplian data awal pada dataset train.csv

```
train.head().T
```

	0	1	2	3	4
id	1	4	5	6	7
keyword	NaN	NaN	NaN	NaN	NaN
location	NaN	NaN	NaN	NaN	NaN
text	Our Deeds are the Reason of this #earthquake M...	Forest fire near La Ronge Sask. Canada	All residents asked to 'shelter in place' are ...	13,000 people receive #wildfires evacuation or...	Just got sent this photo from Ruby #Alaska as ...
target	1	1	1	1	1

Cetak hasil tamplian data awal pada dataset test.csv

```
test.head().T
```

	0	1	2	3	4
id	0	2	3	9	11
keyword	NaN	NaN	NaN	NaN	NaN
location	NaN	NaN	NaN	NaN	NaN
text	Just happened a terrible car crash	Heard about #earthquake is different cities, s...	there is a forest fire at spot pond, geese are...	Apocalypse lighting. #Spokane #wildfires	Typhoon Soudelor kills 28 in China and Taiwan

Untuk memrposes N-gram kita membutuhkan method Stopword yang kita import dari NLTK. List stopword dalam Bahasa inggris:

```
from nltk.corpus import stopwords

stopwords.words("english")
```

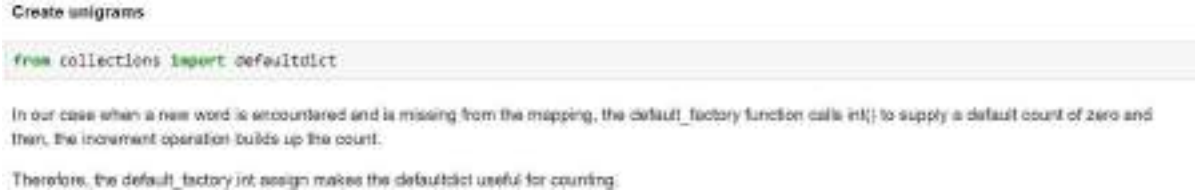
```
['i',
 'me',
 'my',
 'myself',
 'we',
 'our',
 'ours',
 'ourselves',
 'you',
 "you're",
 "you've",
 "you'll",
 "you'd",
 'your',
 'yours',
 'yourself',
 'yourselves',
 'he',
 'him',
```

Berikut fungsi untuk memproses ngrams:

```
from nltk.corpus import stopwords


def generate_ngrams(text, n_gram=1, stop=True):
    """
    Simple n-gram generator.
    """
    stop = set(stopwords.words("english")) if stop else {}

    token = [
        token for token in text.lower().split(" ") if token != "" if token not in stop
    ]
    z = zip(*[token[i:] for i in range(n_gram)])
    ngrams = [" ".join(ngram) for ngram in z]

    return ngrams
```

Unigram

Untuk melakukan unigram kita membuthkan defaultdict dari collections



Dari dataset yang telah diproses sebelumnya kita akan memproses defaultdict:

```
disaster_unigrams = defaultdict(int)
nondisaster_unigrams = defaultdict(int)
```

Mengambil data dari fitur dataset untuk menampilkan list dictionary:

```
for text in train[train.target == 1].text:
    for word in generate_ngrams(text):
        disaster_unigrams[word] += 1

for text in train[train.target == 0].text:
    for word in generate_ngrams(text):
        nondisaster_unigrams[word] += 1
```

Menampilkan list dictioary unigram dari disaster

```
disaster_unigrams
```

```
defaultdict(int,
            {'deeds': 1,
             'reason': 7,
             '#earthquake': 19,
             'may': 47,
             'allah': 4,
             'forgive': 1,
             'us': 40,
             'forest': 44,
             'fire': 150,
             'near': 47,
             'la': 17,
             'ronge': 1,
             'sask.': 1,
             'canada': 4,
             'residents': 7,
             'asked': 1,
             "'shelter": 1,
             "place'": 1,
             'notified': 1,
             'officers.': 1,
             'evacuation': 38,
```

Menampilkan list disaster unigram:

```
disaster_unigrams.items()
```

Membuat dataframe dari list disaster dict. Dan mengurutkan secara ascending:

```
df_disaster_unigrams = pd.DataFrame(
    sorted(disaster_unigrams.items(), key=lambda x: x[1], reverse=True)
)
df_nondisaster_unigrams = pd.DataFrame(
    sorted(nondisaster_unigrams.items(), key=lambda x: x[1], reverse=True)
)
```

Menampilkan tuple data yang telah di sorted disini kita menampilkan 10 data:

```
sorted(disaster_unigrams.items(), key=lambda x: x[1], reverse=True)[:10]
```

```
[('-', 389),
 ('fire', 150),
 ('via', 117),
 ('&amp;', 105),
 ('...', 105),
 ('suicide', 103),
 ('disaster', 97),
 ('people', 93),
 ('police', 93),
 ('killed', 92)]
```

Menampilkan hasil dataframe dalam tabel:

```
df_disaster_unigrams.head()
```

	0	1
0	-	389
1	fire	150
2	via	117
3	&	105
4	...	105

Menampilkan tampilan sebaran data hasil unigram pada bar:

```
d1 = df_disaster_unigrams[0][:10]
d2 = df_disaster_unigrams[1][:10]

nd1 = df_nondisaster_unigrams[0][:10]
nd2 = df_nondisaster_unigrams[1][:10]

plt.figure(1, figsize=(16, 4))
plt.subplot(2, 1, 1)
_ = plt.bar(d1, d2)

plt.subplot(2, 1, 2)
_ = plt.bar(nd1, nd2)
```

Output:

Based on the plot above we can see that most common shared unigrams are punctuation and stopwords so it's important to clean them out.

Regarding the context, even from the unigrams we can see that we get a lot of context regarding whether the text refers to a disaster or not.

Bigram

Untuk proses bigram sama seperti pada Langkah unigram hanya saja untuk deklarasi n_gram=2 untuk bigram:

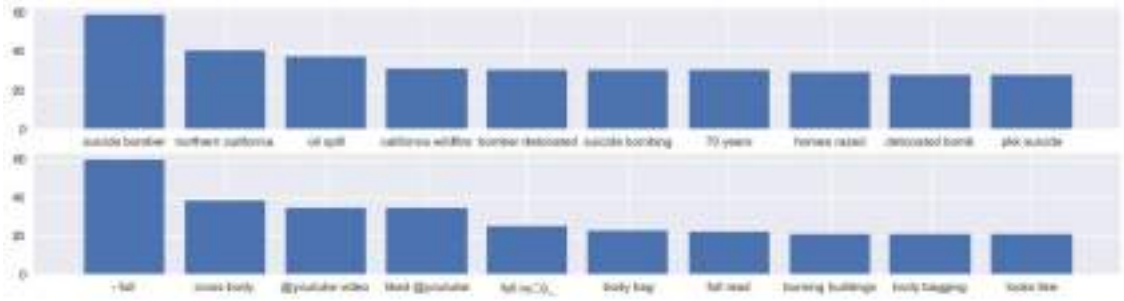
Create bigrams

```
disaster_bigrams = defaultdict(int)
nondisaster_bigrams = defaultdict(int)

for text in train[train.target == 1].text:
    for word in generate_ngrams(text, n_gram=2):
        disaster_bigrams[word] += 1

for text in train[train.target == 0].text:
    for word in generate_ngrams(text, n_gram=2):
        nondisaster_bigrams[word] += 1

df_disaster_bigrams = pd.DataFrame(
    sorted(disaster_bigrams.items(), key=lambda x: x[1])[::-1]
)
df_nondisaster_bigrams = pd.DataFrame(
    sorted(nondisaster_bigrams.items(), key=lambda x: x[1])[::-1]
)
```

Hasil plot bar bigram disaster berdasarkan frekuensinya:

```
d1 = df_disaster_bigrams[0][:10]
d2 = df_disaster_bigrams[1][:10]

nd1 = df_nondisaster_bigrams[0][:10]
nd2 = df_nondisaster_bigrams[1][:10]

plt.figure(1, figsize=(16, 4))
plt.subplot(2, 1, 1)
_ = plt.bar(d1, d2)

plt.subplot(2, 1, 2)
_ = plt.bar(nd1, nd2)
```

Output:



The plot above shows that bigrams give a lot of context to the type of text with regard to whether it's a disaster or not. If we clean the punctuation and stopwords and urls we should get a better understanding.

Untuk mengecek bigrams secara keseluruhan pada corpus atau dataset:

Check the most common bigrams in a whole corpus.

```
from sklearn.feature_extraction.text import CountVectorizer


def get_top_text_ngrams(corpus, ngrams=(1, 1), nr=None):
    """
    Creates a bag of ngrams and counts ngram frequency.

    Returns a sorted list of tuples: (ngram, count)
    """
    vec = CountVectorizer(ngram_range=ngrams).fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_words = bag_of_words.sum(axis=0)
    words_freq = [(word, sum_words[0, idx]) for word, idx in vec.vocabulary_.items()]
    words_freq = sorted(words_freq, key=lambda x: x[1], reverse=True)
    return words_freq[:nr]
```

Menampilkan top 10 biagram dalam corpus:

```
top_text_bigrams = get_top_text_ngrams(train.text, ngrams=(2, 2), nr=10)
```

List top text bigram:

```
top_text_bigrams
```

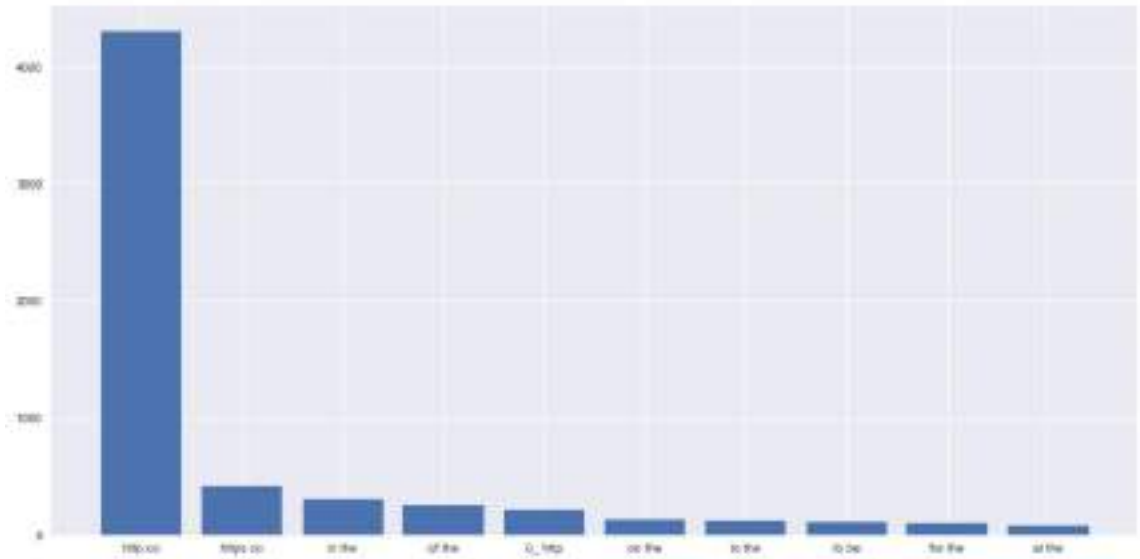
```
[('http co', 4306),
 ('https co', 410),
 ('in the', 308),
 ('of the', 256),
 ('û_ http', 217),
 ('on the', 129),
 ('to the', 126),
 ('to be', 108),
 ('for the', 97),
 ('at the', 85)]
```

Proses sebaran pada plot bar:

```
x, y = zip(*top_text_bigrams)
plt.figure(1, figsize=(16, 8))
plt.subplot(1, 1, 1)
plt.bar(x, y)
```

Output:

```
<BarContainer object of 10 artists>
```



B. N-GRAM KALIMAT dan KATA

Buka notebook baru dan jalankan perintah berikut untuk mengimport library yang diperlukan:

```
#import library yang dibutuhkan
from nltk import ngrams
from nltk.util import ngrams
import nltk
```

Proses fourgram tokenisasi kalimat dengan fungsi split():

```
sentence = 'Buku di perpustakaan sekolah banyak yang rusak'
#print (sentence.split())
n=4 #fourgraw
fourgram = ngrams(sentence.split(),n) #memanggil fungsi ngrams, dari nltk
#ngrams ini punya dua paramaeter, pertama sentence yang dipecah berarti tipe data array
#kedua ngramsnya disini menggunakan fourgram
for grams in fourgram:
    print(grams)
```

```
('Buku', 'di', 'perpustakaan', 'sekolah')
('di', 'perpustakaan', 'sekolah', 'banyak')
('perpustakaan', 'sekolah', 'banyak', 'yang')
('sekolah', 'banyak', 'yang', 'rusak')
```

Proses bigram dengan tokenisasi fungsi word_tokenize():

```
text = "Buku di perpustakaan sekolah banyak yang rusak"
tokenize = nltk.word_tokenize(text)
#print(tokenize)
n = 2
bigrams = ngrams(text.split(),n)
for grams in bigrams:
    print(grams)
```

```
('Buku', 'di')
('di', 'perpustakaan')
('perpustakaan', 'sekolah')
('sekolah', 'banyak')
('banyak', 'yang')
('yang', 'rusak')
```

N-gram karakter/huruf pada Kata:

Jumlah ngram dapat didefinisikan sesuai dengan proses:

```
n = 5
sentence = 'fasilitas di kamar mandi banyak yang rusak'
text_ngrams = ngrams(sentence,n)
for text in text_ngrams:
    print(text)
```

Output:

```
('f', 'a', 's', 'i', 'l')
('a', 's', 'i', 'l', 'i')
('s', 'i', 'l', 'i', 't')
('i', 'l', 'i', 't', 'a')
('l', 'i', 't', 'a', 's')
('i', 't', 'a', 's', ' ')
('t', 'a', 's', ' ', 'd')
('a', 's', ' ', 'd', 'i')
('s', ' ', 'd', 'i', ' ')
(' ', 'd', 'i', ' ', 'k')
('d', 'i', ' ', 'k', 'a')
('i', ' ', 'k', 'a', 'm')
(' ', 'k', 'a', 'm', 'a')
('k', 'a', 'm', 'a', 'r')
('a', 'm', 'a', 'r', ' ')
('m', 'a', 'r', ' ', 'm')
('a', 'r', ' ', 'm', 'a')
('r', ' ', 'm', 'a', 'n')
(' ', 'm', 'a', 'n', 'd')
('m', 'a', 'n', 'd', 'i')
('a', 'n', 'd', 'i', ' ')
('n', 'd', 'i', ' ', 'b')
('d', 'i', ' ', 'b', 'a')
('i', ' ', 'b', 'a', 'n')
(' ', 'b', 'a', 'n', 'y')
('b', 'a', 'n', 'y', 'a')
```

dst.

Tanpa menggunakan library NLTK

```
sentence = 'fasilitas di kamar mandi banyak yang rusak'
n = 5
[sentence[i:i+n] for i in range(len(sentence)-n+1)]
```

```
['fasil',
 'asili',
 'silit',
 'ilita',
 'litas',
 'itas ',
 'tas d',
 'as di',
 's di ',
 ' di k',
 'di ka',
 'i kam',
 ' kama',
 'kamar',
 'amar ',
 'mar m',
 'ar ma',
 'r man',
 ' mand',
 'mandi',
 'andi ',
 'ndi b',
 'di ba',
 'i ban',
 ' bany',
 'banya',
 'anyak',
 'nyak ',
 'yak y',
```

dst.

5.7 POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-02	1. Implementasikan prestest no 2 dan 3 pada prosesngrams! 2. Tampilkan hasil ngrams untuk kalimat berikut ini: “Pemrosesan Bahasa Alami Program Studi Informatika Universitas Ahmad Dahlan tahun 2022” Untuk bigram,trigram, dan fivegram!	100

5.8 HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		

3.	Post-Test	CPL-03	CPMK-01	50%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 6: PARSING

Pertemuan ke 6

Total Alokasi Waktu: 90 menit
- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%
- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas.
CPMK-02	Mampu menjelaskan metode yang digunakan untuk penyelesaian task: language modeling, POS Tagging, parsing, representasi semantik, dan aplikasi PBA.

6.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:
1. Menjelaskan parsing dari grammar
2. Mengimplementasikan parsing dari grammar

6.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-02	Kemampuan mahasiswa dalam menerapkan parsing dari grammar

6.3 TEORI PENDUKUNG

Context Free Grammar (CFG) adalah tata bahasa yang mempunyai tujuan sama seperti halnya tata bahasa regular yaitu merupakan suatu cara untuk menunjukkan bagaimana menghasilkan suatu untai-untai dalam sebuah bahasa.

Context Free Grammar (CFG) menjadi dasar dalam pembentukan suatu parser/proses analisis sintaksis. Bagian sintaks dalam suatu kompilator kebanyakan di definisikan dalam tata bahasa bebas konteks. Pohon penurunan (derivation *tree*/*parse tree*) berguna untuk menggambarkan simbol-simbol variabel menjadi simbol-simbol terminal setiap simbol variabel akan di turunkan menjadi terminal

sampai tidak ada yang belum tergantikan.

Dalam proses *parsing*, Parser melakukan proses kalimat berdasarkan struktur *grammar* yang sudah diciptakan. *Grammar* tersebut dideklarasikan secara *well-formedness* (berdasarkan aturan *grammar* yang telah ditentukan).

Parser tersebut dapat melakukan cek grammar terhadap kalimat yang dimasukan, dimana kesalahan struktur *grammar* banyak dialami pada penulisan kalimat yang tidak baku sesuai ketentuan grammar yang telah disepakati. Didalam pemrosesan bahasa alami *parsing* sering diterapkan pada beberapa kasus, salah satunya pada *question answering*.

6.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Bahasa pemrograman Python
3. JupyterNotebook

6.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-02	Jelaskan *Context Free Grammar* dan juga *Parsing* sesuai dengan pemahaman kalian!	30
2.	CPL-03	CPMK-02	Buat CFG untuk kalimat sederhana dalam Bahasa Indonesia (silakan buat asumsu sendiri)	35
2.	CPL-03	CPMK-02	kalimat : "school of dance and music presents wonderfull town" pada kalimat tersebut lakukan langkah membuat CFG secara top-down parsing (MANUAL/TULIS TANGAN)	35

6.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-02	Selesaikan langkah praktikum	Hasil langkah praktikum	100

Langkah-Langkah Praktikum:

Langkah-langkah membuat *Context Free Grammar* pada NLTK.

1. Import Library NLTK

```
from nltk import CFG
from nltk.parse import RecursiveDescentParser, ShiftReduceParser
```

i. `CFG` modul yang digunakan untuk membuat *context free grammar*.

ii. `RecursiveDescentParser` digunakan untuk memanggil fungsi *rekursive*

7. Menampilkan aturan yang tersimpan dalam *grammar*.

```
for p in grammar.productions():
    print(p)
```

```
S -> NP VP
VP -> V NP
VP -> V NP PP
PP -> P NP
V -> 'saw'
V -> 'ate'
V -> 'walked'
NP -> 'John'
NP -> 'Mary'
NP -> 'Bob'
NP -> Det N
NP -> Det N PP
Det -> 'a'
Det -> 'an'
Det -> 'the'
Det -> 'my'
N -> 'man'
N -> 'dog'
N -> 'cat'
N -> 'telescope'
N -> 'park'
P -> 'in'
P -> 'on'
P -> 'by'
P -> 'with'
```

`productions()` digunakan untuk menampilkan aturan dalam CFG.

8. Memanggil *context free grammar*

```
sentence = "Mary saw Bob".split()
rd_parser = RecursiveDescentParser(grammar)
for tree in rd_parser.parse(sentence):
    print(tree)
```

```
(S (NP Mary) (VP (V saw) (NP Bob)))
```

Berikut merupakan simbol yang digunakan dalam *context free grammar*.

Simbol	Keterangan	Contoh
S	*sentence*	*the man walked*
NP	*noun phrase*	*a dog*
VP	*verb phrase*	*saw a park*

PP	*prepositional phrase*	*with a telescope*
Det	*determiner*	*the*
N	*noun*	*dog*
V	*verb*	*walked*
P	*preposition*	*in*

Context free grammar berisi aturan *grammar* yang akan digunakan dalam proses *parsing*, diantaranya adalah:

- S → NP VP

  Sentence merupakan susunan kata yang terdiri dari *Noun Phrase* dan *Verb Phrase*.VP → V NP | V NP PP

  – *Verb Phrase* merupakan susunan kata yang terdiri dari *Verb* dan *Noun Phrase*.

  – *Verb Phrase* merupakan susunan kata yang terdiri dari *Verb* dan *Noun Phrase* dan *Prepositional Phrase*.

- **PP → P NP**

  Prepositional Phrase merupakan susunan kata yang terdiri dari *Preposition* dan *Noun Phrase*.
  NP → Det N | Det N PP

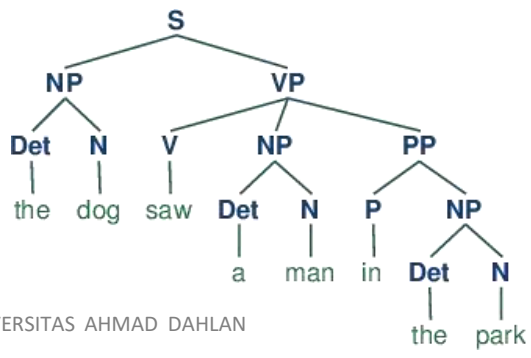
  – *Noun Phrase* merupakan susunan kata yang terdiri dari *Determiner* dan *Noun*.

  – *Noun Phrase* merupakan susunan kata yang terdiri dari *Determiner* dan *Noun* dan *Prepositional Phrase*.

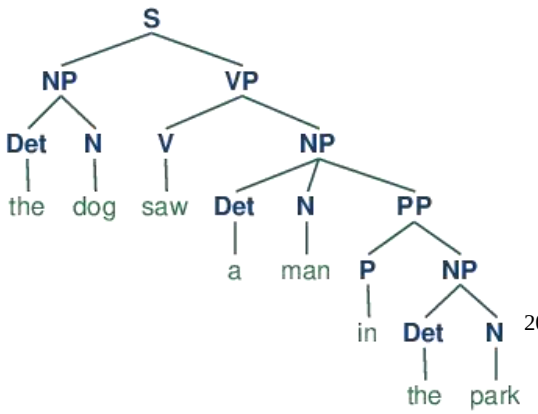
- Verb → "saw"|"ate"|"walked"
  – *Verb* berisi beberapa kata, yaitu *saw, ate* dan *walked*

Misalkan kita akan melakukan *parsing* kalimat `the dog saw a man in the park` menggunakan *grammar* di atas, maka hasilnya adalah.

a. Untuk kasus **VP → V NP PP**



b. Untuk kasus **VP → V NP**



Hasil program:

```
sentence = "the dog saw a man in the park".split()
rd_parser = RecursiveDescentParser(grammar)
for tree in rd_parser.parse(sentence):
    print tree
```

6.7 POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-02	"Mahasiswa belajar komputer di laboratorium pada jam kuliah" Pada kalimat diatas lakukan langkah membuat CFG secara bottom-up parsing! pada kalimat diatas lakukan langkah membuat CFG secara manual!	50 50

6.8 HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

Pertemuan ke 7

Total Alokasi Waktu: 90 menit

- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%

- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas.
CPMK-02	Mampu menjelaskan metode yang digunakan untuk penyelesaian task: language modeling, POS Tagging, parsing, representasi semantik, dan aplikasi PBA.

7.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:

1. Menjelaskan konsep NER dalam pemrosesan bahasa.
2. Menerapkan NER dalam pemrosesan bahasa.

7.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-01	CPMK-02	Kemampuan mahasiswa dalam menerapkan NER dalam suatudokumen.

7.3 TEORI PENDUKUNG

Named entity adalah Setiap kata yang mewakili seseorang, organisasi, lokasi, dll. Pengenalan *Named entity* adalah sub tugas dari Ekstraksi Informasi dan merupakan proses mengidentifikasi kata-kata yang diberi nama entitas yang diberikan dalam teks. Ini juga disebut identifikasi entitas atau *chunkin* entitas.

Named-Entity Recognition (NER) merupakan bagian dari riset Natural Language Processing (NLP) yang digunakan untuk mengekstrak informasi seperti nama orang, organisasi, lokasi, dan waktu. NER telah digunakan di banyak bidang pekerjaan salah satunya pada pengembangan chatbot. Penggunaan NLP dan machine learning (ML) pada chatbot dapat membuat chatbot lebih cerdas dengan analisis personal yang lebih baik ke pengguna. Ada 3 buah pendekatan untuk melakukan proses NER yaitu secara *Basic NLTK algorithm, Stanford NLP NER, dan Using Spacy*.

Contoh named entity dalam suatu kalimat:

"18.35: Banjir setinggi 60-100 cm di Jln Sultan Agung, dekat Pelabuhan 1 Cirebon, pengendara diharapkan berhati-hati".

Setelah proses NER dijalankan, akan didapat named-entity (NE) atau sering disebut mention beserta tipenya: "18:35" bertipe waktu; "banjir" bertipe kejadian; "60-100cm" bertipe ukuran "Jln. Sultan Agung", "pelabuhan 1" dan "Cirebon" bertipe lokasi. Dari contoh di atas dapat dilihat bahwa fungsi dari NER adalah mendeteksi kata atau kumpulan kata yang merupakan entitas dan mengkategorikan kata tersebut ke dalam tipe yang sesuai.

7.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Bahasa pemrograman Python
3. JupyterNotebook

7.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-02	Apa aitu NER?	20
2.	CPL-03	CPMK-02	Sebutkan kegunaan dan contoh pendekaatan NER!	30
3.	CPL-03	CPMK-02	Buatlah contoh kalimat dan tentukan NE nya!	50

7.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-02	Selesaikan langkah praktikum 1 – 9 (Basic NLT Algorithm)	Hasil praktikum langkah 1 – 9	60
2.	CPL-03	CPMK-02	Selesaikan langkah praktikum 1 – 6 (Using Spacy)	Hasil praktikum langkah 1 – 6	40

Langkah-Langkah Praktikum:

A. Menentukan Named Entity menggunakan Basic NLTK Algorithm

Pada Basic NLTK Algorithm terdapat dua buah cara untuk menentukan named entity yaitu dengan word segmentation dan sentence segmentation.

1. Import dependencies

Ketikkan perintah berikut untuk import dependencies NTLK:

```
import nltk
import pandas as pd
```

Kemudian run atau tekan tombol (shift+enter) pada keyboard.

2. Definisikan teks yang akan dilakukan proses named entity seperti berikut:

```
text = "Apple acquired Zoom in china on Wednesday 6th May 2020.\
This news has made Apple and Google stock jump by 5% on Dow Jones Index in the\
United States of America"
```

Kemudian run atau tekan tombol (shift+enter) pada keyboard.

3. Untuk proses pertama kita akan menggunakan word base segementation dimana kalima

t yang ada akan dipecah kedalam bentuk kumpulan kata. Ketikkan perintah berikut :

```
#tokenize to words
words = nltk.word_tokenize(text)
words
```

Kemudian run atau tekan tombol (shift+enter) pada keyboard.Maka akan muncul hasil tampilan sebagai berikut:

```
['Apple',
 'acquired',
 'Zoom',
 'in',
 'china',
 'on',
 'Wednesday',
 '6th',
 'May',
 '2020.This',
 'news',
 'has',
 'made',
 'Apple',
 'and',
 'Google',
 'stock',
 'jump',
 'by',
 '5',
 '%',
 'on',
 'Dow',
 'Jones',
 'Index',
 'in',
 'theUnited',
 'States',
 'of',
 'America']
```

4. kemudian kita akan memberikan label / tags pada setiap kata yang ada :

```
#Part of speech tagging
pos_tags =nltk.pos_tag(words)
pos_tags
```

Setelah di run maka akan memeberikan output sebagai berikut :

```
[('Apple', 'NNP'),
 ('acquired', 'VBD'),
 ('Zoom', 'NNP'),
 ('in', 'IN'),
 ('china', 'NN'),
 ('on', 'IN'),
 ('Wednesday', 'NNP'),
 ('6th', 'CD'),
 ('May', 'NNP'),
 ('2020.This', 'CD'),
 ('news', 'NN'),
 ('has', 'VBZ'),
 ('made', 'VBN'),
 ('Apple', 'NNP'),
 ('and', 'CC'),
 ('Google', 'NNP'),
 ('stock', 'NN'),
 ('jump', 'NN'),
 ('by', 'IN'),
 ('5', 'CD'),
 ('%', 'NN'),
 ('on', 'IN'),
 ('Dow', 'NNP'),
 ('Jones', 'NNP'),
 ('Index', 'NNP'),
 ('in', 'IN'),
 ('theUnited', 'JJ'),
 ('States', 'NNS'),
 ('of', 'IN'),
 ('America', 'NNP')]
```

5. Untuk mengetahui maksud dari tags tersebut (CD/IN/JJ/NNP/dll) maka dapat menjalankan perintah berikut:

```
#check nltk help for description of the tag
nltk.help.upenn_tagset('NNP')
```

```
NNP: noun, proper, singular
    Motown Venneboerger Czestochwa Ranzer Conchita Trumplane Christos
    Oceanside Escobar Kreisler Sawyer Cougar Yvette Ervin ODI Darryl CTCA
    Shannon A.K.C. Meltex Liverpool ...
```

6. kemudian ketikan function ne_chunck untuk memebrikan tags/label pada kata yang sudah diuraikan sebelumnya. Parameter binary = True hanya akan memebrikan 2 output berupa kata yang termasuk / berlabel NE (*Named entity*) ataupun bukan tidak NE.

```
chunks = nltk.ne_chunk(pos_tags, binary=True) #either NE or not NE
for chunk in chunks:
    print(chunk)
```

Hasil output :

```
(NE Apple/NNP)
('acquired', 'VBD')
('Zoom', 'NNP')
('in', 'IN')
(NE China/NNP)
('on', 'IN')
('Wednesday', 'NNP')
('6th', 'CD')
('May', 'NNP')
('2020.This', 'CD')
('news', 'NN')
('has', 'VBZ')
('made', 'VBN')
(NE Apple/NNP)
('and', 'CC')
(NE Google/NNP)
('stock', 'NN')
('jump', 'NN')
('by', 'IN')
('5', 'CD')
('%', 'NN')
('on', 'IN')
('Dow', 'NNP')
('Jones', 'NNP')
('Index', 'NNP')
('in', 'IN')
('the', 'DT')
(NE United/NNP States/NNPS)
('of', 'IN')
(NE America/NNP)
```

7. Membuat data frame berisikan kata hasil proses pencarian NE dengan parameterbinary=True:

```
entities =[]
labels =[]
for chunk in chunks:
    if hasattr(chunk,'label'):
        #print(chunk)
        entities.append(' '.join(c[0] for c in chunk))
        labels.append(chunk.label())

entities_labels = list(set(zip(entities, labels)))
entities_df = pd.DataFrame(entities_labels)
entities_df.columns = ["Entities","Labels"]
entities_df
```

Hasil output :

	Entities	Labels
0	Apple	NE
1	China	NE
2	Google	NE
3	United States	NE
4	America	NE

8. Ada beberapa data yang hilang atau tidak muncul pada data frame kemudian kita uji dengan

Parameter binary = False untuk mendapatkan label yang lebih bervariatif:

```
chunks = nltk.ne_chunk(pos_tags, binary=False) #either NE or not NE
for chunk in chunks:
    print(chunk)

entities =[]
labels =[]
for chunk in chunks:
    if hasattr(chunk,'label'):
        #print(chunk)
        entities.append(' '.join(c[0] for c in chunk))
        labels.append(chunk.label())

entities_labels = list(set(zip(entities, labels)))
entities_df = pd.DataFrame(entities_labels)
entities_df.columns = ["Entities","Labels"]
entities_df
```

Hasil output :

```
(PERSON Apple/NNP)
('acquired', 'VBD')
(PERSON Zoom/NNP)
('in', 'IN')
(GPE China/NNP)
('on', 'IN')
('Wednesday', 'NNP')
('6th', 'CD')
('May', 'NNP')
('2020.This', 'CD')
('news', 'NN')
('has', 'VBZ')
('made', 'VBN')
(PERSON Apple/NNP)
('and', 'CC')
(ORGANIZATION Google/NNP)
('stock', 'NN')
('jump', 'NN')
('by', 'IN')
('5', 'CD')
('%', 'NN')
('on', 'IN')
(PERSON Dow/NNP Jones/NNP Index/NNP)
('in', 'IN')
('the', 'DT')
(GPE United/NNP States/NNPS)
('of', 'IN')
(GPE America/NNP)
```

	Entities	Labels
0	Dow Jones Index	PERSON
1	Apple	PERSON
2	Google	ORGANIZATION
3	Zoom	PERSON
4	United States	GPE
5	America	GPE
6	China	GPE

9. Untuk sentence base kita akan mentokenize kedalam kata kemudian melakukanproses

Chunking. Kemudian menampilkan hasil NE pada dataframe:

```
entities = []
labels = []

sentence = nltk.sent_tokenize(text)
for sent in sentence:
    for chunk in nltk.ne_chunk(nltk.pos_tag(nltk.word_tokenize(sent)),binary=False):
        if hasattr(chunk,'label'):
            entities.append(' '.join(c[0] for c in chunk))
            labels.append(chunk.label())

entities_labels = list(set(zip(entities,labels)))

entities_df = pd.DataFrame(entities_labels)
entities_df.columns = ["Entities","Labels"]
entities_df
```

Output:

	Entities	Labels
0	Dow Jones Index	PERSON
1	Apple	PERSON
2	Google	ORGANIZATION
3	Zoom	PERSON
4	United States	GPE
5	America	GPE
6	China	GPE

B. Using Spacy

Spacy memberikan hasil yang lebih cepat dan akurasi yang baik. Untuk mendapatkan informasi lebih lanjut dapat mengunjungi Spacy library Link: https://spacy.io/.

1. Pada halam notebook baru lakukan import dependencies:

```
import nltk
import pandas as pd
import spacy
from spacy import displacy
```

2. perintah untuk menampilkan ver yang berjalan.

```
#SpaCy 2.x brough significant speed and accuracy improvements
spacy.__version__
```

```
'3.0.5'
```

3. Download spacy language models. Ada 3 versi yaitu small, medium dan large bergantung pada kompleksitas dan proses yang berjalan untuk proses ini kita gunakan ver. Small untuk mengurangi penggunaan storage yang besar dan agar proses berjalan lebih cepat. Ketikkan command berikut :

```
#Download spacy models
#!python -m spacy download en_core_web_sm
```

```
# Load SpaCy model
nlp = spacy.load("en_core_web_sm")
#nlp = spacy.load("en_core_web_md")
#nlp = spacy.load("en_core_web_lg")
```


4. Definisikan teks yang akan digunakan pada prose NER.

```
text = "Apple acquired Zoom in China on Wednesday 6th May 2020.\
This news has made Apple and Google stock jump by 5% on Dow Jones Index in the \
United States of America"
```

5. Untuk menampilkan hasil dari NER menggunakan spacy maka kita akan mengconvert data

dalam teks kedalam parameter doc.kemudian akan mengambil data dari setiapentitas dan

label list. Berikut perintah yang akan dijalankan:

```
doc = nlp(text)

entities = []
labels = []
position_start = []
position_end = []

for ent in doc.ents:
    entities.append(ent)
    labels.append(ent.label_)
    position_start.append(ent.start_char)
    position_end.append(ent.end_char)

df = pd.DataFrame({'Entities':entities,'Labels':labels,'Position_Start':position_start, 'Position_End':position_end})

df
```

Output menampilkan hasil dengan nama entitas yang variative :

	Entities	Labels	Position_Start	Position_End
0	(Apple)	ORG	0	5
1	(Zoom)	PERSON	15	19
2	(China)	GPE	23	28
3	(Wednesday, 6th)	DATE	32	45
4	(Apple)	ORG	74	79
5	(5, %)	PERCENT	105	107
6	(Dow, Jones)	ORG	111	120
7	(the, United, States, of, America)	GPE	130	158

6. Berikut function untuk mengetahui makna dari setiap label yang diberikan dari hasil output:

```
spacy.explain("ORG")
```

```
'Companies, agencies, institutions, etc.'
```

```
spacy.explain("GPE")
```

```
'Countries, cities, states'
```

7.7 POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01	Tentukan NE dari kalimat berikut menggunakan basic NLTK Algorithm Screenshot setiap langkah: 1. *Mark Zuckerberg is one of the founders of Facebook, a company from the United States* Meta Platforms saw its stock market value slump by more than $230bn (£169bn) on Thursday, in a record daily loss for a US firm. Meta also warned of slowing revenue growth in the face of competition from rival platforms including TikTok and YouTube, while advertisers were also cutting spending.	100

7.8 HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 8: TEXT SUMMARIZATION

Pertemuan ke 8

Total Alokasi Waktu: 90 menit
- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%
- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas.
CPMK-02	Mampu menjelaskan metode yang digunakan untuk penyelesaian task: language modeling, POS Tagging, parsing, representasi semantik, dan aplikasi PBA.

8.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:
1. Menjelaskan task peringkasan teks
2. Menerapkan sebuah contoh kasus.

8.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-03	CPMK-02	Kemampuan mahasiswa dalam menerapkan studi kasus.

8.3 TEORI PENDUKUNG

A. Pengertian *Text Summarization*

Text Summarization atau ringkasan teks pertama kali diperkenalkan pada tahun 1950 (Gaikwad & Mahender, 2016), merupakan suatu metode dalam membuat ringkasan yang singkat, akurat, dan dapat dicerna dari suatu dokumen teks yang panjang. Gagasan utama di balik *Text Summarization* adalah untuk dapat menemukan subset singkat dari informasi palin gpenting dari seluruh rangkaian dan menyajikannya dalam format yang dapat dibaca manusia. Seiring dengan pertumbuhan data tekstual online, metode *text summarization* berpotensi sangat membantu karena informasi yang lebih bermanfaat dapat dibaca dalam

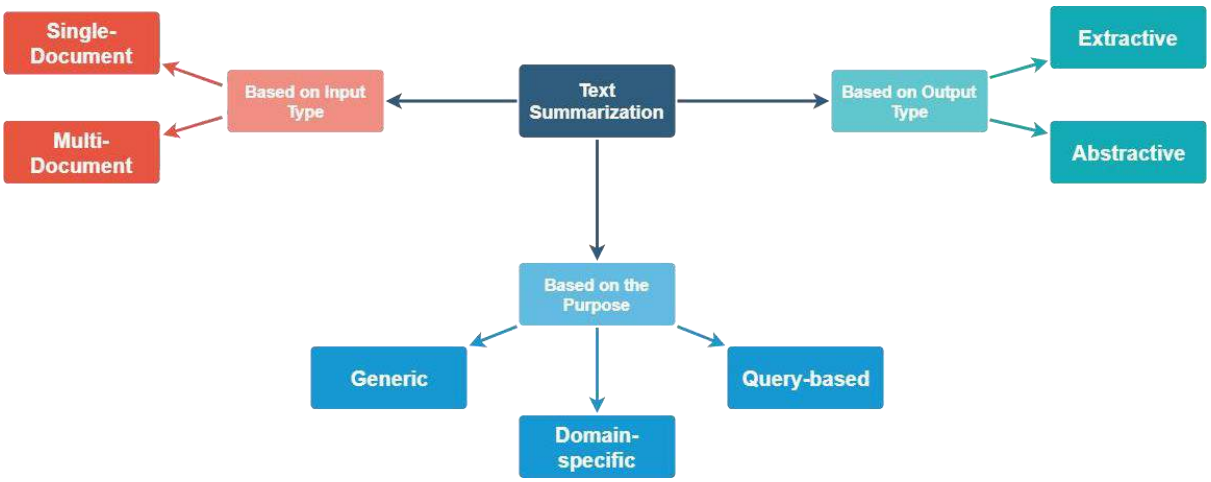
waktu singkat. Sedangkan *automatic text summarization* adalah suatu langkah untuk membuat ringkasan

secara otomatis dengan tidak mengubah inti dari suatu dokumen yang diringkas. *Automatic text summarization* atau biasa disebut *automatic summarization* sangat dibutuhkan pada era *big data* sekarang ini, di mana jumlah data teks setiap hari selalu bertambah dengan sangat banyak dan tidak terstruktur sehingga untuk membantu menemukan informasi yang relevan dengan lebih cepat dibutuhkan *automatic summarization*. Terdapat beberapa hal yang membuat *automatic text summarization* penting yaitu :

3. Ringkasan mengurangi waktu membaca
4. Saat meneliti dokumen, ringkasan membuat proses seleksi lebi mudah
5. *Automatic Text Summarization* meningkatkan efektifitas pengindeksan
6. Algoritma *Automatic Text Summarization* kurang bias dibandingkan peringkasan manusia
7. Ringkasan yang dipersonalisasi berguna dalam system penjawab pertanyaan karena memberikan informasi yang dipersonalisasi
8. Menggunakan system *Automatic Text Summarization* atau *semi-automatic* memungkinkan layanan abstrak komersial untuk meningkatkan jumlah dokumen teks yang dapat mereka proses

B. Tipe *Text Summarization*

Tipe – tipe *Text* Summarization dapat dilihat pada Gambar 1.1.



Gambar 2.1 Tipe *Text Summarization*

- Berdasarkan *Input Type :*
  1. *Single-Document*, dimana Panjang inputnya pendek. Banyak dari system peringkasan awal berurusan dengan peringkasan *single-document.*
  2. *Multi-Document*, dimana inputan dapat panjang secara cuma – Cuma.

- Berdasarkan *Purpose* (tujuan) :
  1. *Generic,* dimana model tidak membuat asumsi tentang domain atau konten teks yang akan diringkas dan memperlakukan semua input sebagai homogen. Sebagian besar pekerjaan yang telah dilakukan berkisar pada ringkasan umum.
  2. *Domain-specific*, dimana model menggunakan pengetahuan khusus domain untuk membentuk ringkasan yang lebih akurat. Misalnya, meringkas makalah penelitian dari domain tertentu, dokumen biomedis, dll.

3. *Query-based*, dimana ringkasan hanya berisi informasi yang menjawab pertanyaan bahasa alami tentang teks input.

- Berdasarkan *Output Type :*
  1. *Extractive*, dimana kalimat penting dipilih dari teks input untuk membentuk ringkasan. Kebanyakan pendekatan peringkasan saat ini bersifat *extractive*.
  2. *Abstractive*, dimana model membentuk frasa dan kalimatnya sendiri untuk menwarkan ringkasan yang lebih koheren, seperti yang akan dihasilkan manusia. Pendekatan ini jelas lebih menarik, tetapi jauh lebih sulit daripada ringkasan *extractive.*

C. Langkah – langkah dalam melakukan *Text Summarization*

1. *Text Cleaning*
2. *Sentence Tokenization*
3. *Word Tokenization*
4. *Word-Frequency Table*
5. *Summarization*

8.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

8.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-02	Jelaskan apa yang dimaksud dengan *TextSummarization* dan *Automatic Text Summarization* ?	25
2.	CPL-03	CPMK-02	Sebutkan beberapa hal yang membuat *AutomaticText Summarization* penting untuk dilakukan ?	25
3.	CPL-03	CPMK-02	Gambarkan bagan alir dari *Text Summarization* danjelaskan masing – masing dari bagan tersebut	25
4.	CPL-03	CPMK-02	Sebutkan langkah – langkah dari *Text Summarization ?*	25

8.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-02	Selesaikan langkah praktikum 1 – 21	Hasil praktikum langkah 1 – 21	

Langkah-Langkah Praktikum:
Menentukkan Text Summarization Menggunakan Spacy

1. Install Library Spacy

   Install spacy language models. Ada 3 versi yaitu small, medium dan large bergantung pada kompleksitas dan proses yang berjalan untuk proses ini kita gunakan ver. Small untuk mengurangi penggunaan storage yang besar dan agar proses berjalan lebih cepat.

Apabila anda belum menginstall library spacy, install terlebih dahulu library tersebut menggunakan perintah :

```
!pip install -U spacy
```

```
!python -m spacy download en_core_web_sm
```

2. Import Dependencies

   Ketikkan perintah berikut untuk mengimport dependencies spacy :

```
import spacy
from spacy.lang.en.stop_words import STOP_WORDS
from string import punctuation
```

   Kemudian run atau tekan tombol (shift+enter) pada keyboard.

3. Definisikan teks yang akan dilakukan proses summarization text seperti berikut:

```
text = """ There are broadly two types of extractive summarization tasks depending on what the summarization program focuses on.
The first is generic summarization, which focuses on obtaining a generic summary or abstract of the collection
(whether documents, or sets of images, or videos, news stories etc.). The second is query relevant summarization,
sometimes called query-based summarization, which summarizes objects specific to a query. Summarization systems are able to
create both query relevant text summaries and generic machine-generated summaries depending on what the user needs.
An example of a summarization problem is document summarization, which attempts to automatically produce an abstract from
a given document. Sometimes one might be interested in generating a summary from a single source document, while others
can use multiple source documents (for example, a cluster of articles on the same topic). This problem is called multi-document
summarization. A related application is summarizing news articles. Imagine a system, which automatically pulls together news
articles on a given topic (from the web), and concisely represents the latest news as a summary.
Image collection summarization is another application example of automatic summarization. It consists in selecting a
representative set of images from a larger set of images.A summary in this context is useful to show the most representative
images of results in an image collection exploration system. Video summarization is a related domain, where the system
automatically creates a trailer of a long video. This also has applications in consumer or personal videos, where one might
want to skip the boring or repetitive actions. Similarly, in surveillance videos, one would want to extract important and
suspicious activity, while ignoring all the boring and redundant frames captured."""
```

   Kemudian run atau tekan tombol (shift+enter) pada keyboard.

4. Perintah untuk menampilkan daftar kata umum yang memilki fungsi tapi tidak mempunyai arti

```
stopwords = list(STOP_WORDS)
```

5. Membuat model NLP menggunakan en_core_web_sm

```
nlp = spacy.load('en_core_web_sm')
```

6. Setelah membuat model NLP, selanjutnya melakukan tokenisasi terhadap teks yang telah di definisikan sebelumnya menggunakan perintah :

```
doc = nlp(text)
```

7. Perintah untuk menampilkan daftar token

```
tokens = [token.text for token in doc]
print(tokens)
```

8. Fungsi untuk menambahkan baris baru ke dalam tanda baca

```
punctuation = punctuation + '\n'
punctuation
```

9. Fungsi untuk menampilkan frekuensi kata

```
word_frequencies = {}
for word in doc:
    if word.text.lower() not in stopwords:
        if word.text.lower() not in punctuation:
            if word.text not in word_frequencies.keys():
                word_frequencies[word.text] = 1
            else:
                word_frequencies[word.text] += 1
```

```
print(word_frequencies)
```

10. Menampilkan jumlah frekuensi kata maksimal. Hasil yang ditampilkan nantinya disebut sebagai frekuensi normalisasi maksimum.

```
max_frequency = max(word_frequencies.values())
```

```
max_frequency
```

11. Kemudian bagi semua nilai dengan label yang dapat dilakukan dengan menggunakan for loop. Hasilnya ialah frequensi yang dinormalisasi dari masing – masing kata yang ada di dalam teks data.

```
for word in word_frequencies.keys():
    word_frequencies[word] = word_frequencies[word]/max_frequency
```

```
print(word_frequencies)
```

12. Perintah selanjutnya yaitu melakukan tokenisasi kalimat

```
sentence_tokens = [sent for sent in doc.sents]
print(sentence_tokens)
```

13. Menghitung score kalimat dengan membuat kamus, dengan cara menambahkan nilai sebenernya. Setiap kata memiliki frekuensi yang dinormalisasi, selanjutnya kita akan menambahkan jumlah frekuensi yang dinormalisasi pada setiap kalimat. Kemudian diperoleh nilai maksimum yang akan kita pilih kalimat yang paling penting dari nilai tersebut.

```
sentence_scores = {}
for sent in sentence_tokens:
    for word in sent:
        if word.text.lower() in word_frequencies.keys():
            if sent not in sentence_scores.keys():
                sentence_scores[sent] = word_frequencies[word.text.lower()]
            else:
                sentence_scores[sent] += word_frequencies[word.text.lower()]
```

```
santence_scores
```

14. Import fungsi nlargest(). Fungsi nlargest() merupakan fungs dari modul python heapq yang bertujuan untuk mengembalikan jumlah elemen terbesar yang ditentukan yang ditentukan dari iterable python seperti daftar, tupel, dan lainnya. Fungsi nlargest() dapat melewati fungsi kunci yang mengembalikan fungsi perbandingan untuk digunakan dalam pengurutan.

```
from heapq import nlargest
```

15. Perintah untuk mengetahui total panjang kalimat yang ada dan kemudian dikalikan dengan 0.3, dan hanya dipilih 30% dari total kalimat

```
select_length = int(len(sentence_tokens)*0.3)
select_length
```

16. Untuk meringkas kalimat dan teks menggunakan perintah summary. Dimana kalimat – kalimat tersebut merupakan kalimat terpenting yang nantinya akan digabungkan.

```
summary = nlargest(select_length, sentence_scores, key = sentence_scores.get)
summary
```

17. Untuk menggabungkan kalimat – kalimat menggunakan perintah :

```
final_summary = [word.text for word in summary]
summary = ' '.join(final_summary)
```

18. Perintah untuk menampilkan teks asli

```
print(text)
```

19. Perintah untuk menampilkan teks hasil ringkasan

```
print(summary)
```

20. Menampilkan panjang teks asli

```
len(text)
```

21. Menampilkan panjang teks setelah diringkas

```
len(summary)
```

21.1. POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-03	CPMK-01		
2.	dst			

21.2. HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-03	CPMK-01	20%		
2.	Praktik	CPL-03	CPMK-01	30%		
3.	Post-Test	CPL-03	CPMK-01	50%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 9: TRANSLATION

Pertemuan ke 9

Total Alokasi Waktu: 90 menit
- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%
- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas.
CPMK-02	Mampu menjelaskan metode yang digunakan untuk penyelesaian task: language modeling, POS Tagging, parsing, representasi semantik, dan aplikasi PBA.

9.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu:
1. Menjelaskan apa itu Text Translate
2. Menerapkan proses deep translator

9.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-02	CPMK-02	Kemampuan mahasiswa dalam menerapkan proses text translatepada deep translator

9.3 TEORI PENDUKUNG

Machine translation (MT) adalah proses penggunaan kecerdasan buatan (AI) untuk menerjemahkan konten secara otomatis dari satu bahasa (sumber) ke bahasa lain (target) tanpa masukan manusia. Proses menerjemahkan teks maupun kata dari suatu Bahasa pada Bahasa lainnya dapat dilakukan dengan memanfaatkan modul translator.

manfaat dari Terjemahan mesin memberikan pemahaman dokumen yang cepat dan komprehensif. Jika secara khusus melatih mesin sesuai dengan kebutuhan, MT akan memberikan kombinasi sempurna dalam proses terjemahan yang cepat dan hemat biaya. Dengan mesin yang terlatih khusus, MT dapat menangkap konteks kalimat lengkap sebelum menerjemahkannya, yang memberikan Anda kualitas tinggi dan keluaran yang terdengar seperti manusia. Dengan alat terjemahan mesin, tata letak teks akan dipertahankan, dan terjemahan segera dikembalikan.

Beberapa contoh List of Translator :

1. Google Translate
2. Mymemory Translator
3. DeepL Translator
4. QCRI Translator
5. Linguee Translator
6. PONS Translator
7. Yandex Translator
8. Microsoft Translator
9. Papago Translator

9.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Bahasa pemrograman Python
3. JupyterNotebook

9.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-0	CPMK-0	Apa itu MT dan text translation?	40
2.	CPL-0	CPMK-0	Sebutkan 6 contoh language translator publisher!	10
3.	CPL-0	CPMK-0	Sebutkan perbedaan PONS, LInguee, MyMemory dan Google Translator!	50

9.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-03	CPMK-02	Selesaikan langkah praktikum 1 – 3 (Basic NLT Algorithm)	Hasil praktikum langkah 1 – 3	25
2.	CPL-03	CPMK-02	Selesaikan langkah praktikum 1 – 2 (Using Spacy)	Hasil praktikum langkah 1 – 2	25
3.	CPL-03	CPMK-02	Selesaikan langkah praktikum 1 – 4 (Using Spacy)	Hasil praktikum langkah 1 – 4	25
4.	CPL-03	CPMK-02	Selesaikan langkah praktikum 1 – 2 (Using Spacy)	Hasil praktikum langkah 1 – 2	25

Langkah-Langkah Praktikum:

Untuk melakukan proses penerjemahan teks melalui beberapa tahapan dan metode berikut :

1. Download function deep translator seperti berikut:

```
!pip install deep-translator
```

```
Collecting deep-translator
  Downloading https://files.pythonhosted.org/packages/2a/4f/a89478fa7b9b67c5abf3468a8dc59bae336a10828aa79cd65a8064dffa/deep_translator-1.7.0-py3-none-any.whl
Collecting beautifulsoup4<5.0.0,>=4.9.1 (from deep-translator)
  Downloading https://files.pythonhosted.org/packages/69/bf/f0f194d3379d3f3347478bd267f754fc68c11cbf2fe302ada69447b1417/beautifulsoup4-4.10.0-py3-none-any.whl (97kB)
Collecting requests<5.0.0,>=2.23.0 (from deep-translator)
  Downloading https://files.pythonhosted.org/packages/2d/61/08076519c80041bc0ffa1a8af0c6d3bf3a2b62af10435d269a9d0f40564d/requests-2.27.1-py2.py3-none-any.whl (63kB)
Requirement already satisfied: soupsieve>1.2 in c:\users\acer\anaconda3\lib\site-packages (from beautifulsoup4<5.0.0,>=4.9.1->deep-translator) (1.9.3)
Collecting charset-normalizer~=2.0.0; python_version >= "3" (from requests<3.0.0,>=2.23.0->deep-translator)
  Downloading https://files.pythonhosted.org/packages/06/b3/24afc8868eba069a7f03650ac750a778862dc34941a4bebeb58706715726/charset_normalizer-2.0.12-py3-none-any.whl
Requirement already satisfied: idna<4,>=2.5; python_version >= "3" in c:\users\acer\anaconda3\lib\site-packages (from requests<3.0.0,>=2.23.0->deep-translator) (2.8)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\acer\anaconda3\lib\site-packages (from requests<3.0.0,>=2.23.0->deep-translator) (1.24.2)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\acer\anaconda3\lib\site-packages (from requests<3.0.0,>=2.23.0->deep-translator) (2019.9.11)
Installing collected packages: beautifulsoup4, charset-normalizer, requests, deep-translator
  Found existing installation: beautifulsoup4 4.8.0
    Uninstalling beautifulsoup4-4.8.0:
      Successfully uninstalled beautifulsoup4-4.8.0
  Found existing installation: requests 2.22.0
    Uninstalling requests-2.22.0:
      Successfully uninstalled requests-2.22.0
Successfully installed beautifulsoup4-4.10.0 charset-normalizer-2.0.12 deep-translator-1.7.0 requests-2.27.1
```

Google translator

Dalam deep translator sudah terintegrasi google translator didalamnya. Untuk informasi detail dapat mengunjungi link Language: [https://py-googletrans.readthedocs.io/en/latest/.](https://py-googletrans.readthedocs.io/en/latest/)

2. Untuk menggunakan Google translator dalam menerjemahkan suatu kata maka kita harus mengimport parameter GoogleTranslator dari deep translator. Berikut contoh translate dari English to Germany:

```python
from deep_translator import GoogleTranslator
to_translate = 'I want to translate this text'
translated = GoogleTranslator(source='auto', target='de').translate(to_translate)
translated
```

```
'Ich möchte diesen Text übersetzen'
```

3. Kemudian untuk menerjemahkan kalimat atau lebih dari 1 kata:

```python
from deep_translator import GoogleTranslator
to_translate = 'I want to translate this text'
translated = GoogleTranslator(source='auto', target='id').translate(to_translate)
translated
```

```
'Saya ingin menerjemahkan teks ini'
```

PONS

Juga merupakan salah satu publisher Bahasa seperti google translate yang berasal dari jerman.

1. PONS is one of Germany's leading language publisher
2. It is mostly famous for translating single words or small sentences.
3. It has a rich database of words and can even outperform google translate when itcomes to translating words and getting synonyms

1. Untuk menampilkan hasil translate satu kata tertentu dengan hasil mendekati dalam databae menggunakan return_all=False:

```
from deep_translator import PonsTranslator
word = 'good'
translated_word = PonsTranslator(source='english', target='french').translate(word, return_all=False)
translated_word
```

```
'bien '
```

2. Sedangkan saat return_all=True akan memberikan list kata dengan artian good ataurekomendasi sinonim.

```
#list all synonyms
from deep_translator import PonsTranslator
word = 'good'
translated_word = PonsTranslator(source='english', target='french').translate(word, return_all=True)
translated_word
```

```
['bien ',
 'le bien et le mal ',
 'faire le bien ',
 'mijoter ',
 'mal tourner ',
 'bien ',
 'pour le bien de la société ',
 "c'est pour ton bien ",
 "pour le peu de bien que ça m'a fait ",
 'grand bien lui fasse ',
 'elle est trop généreuse et ça lui jouera des tours ',
 'pour sa santé ',
 "tu crois que ça m'amuse de faire ça ",
 "ça n'a pas arrangé ma migraine ",
 "une grève n'arrangera pas les affaires de l'entreprise ",
 'la pluie fait du bien aux plantes ',
 'ça te fera du bien de dormir ',
 "rien de bon n'en sortira ",
 'attendre ne changera rien ',
```

Linguee

1. The Linguee translator is an online bilingual concordance that provides an online dictionary for a number of language pairs, including many bilingual sentence pairs
2. Link Linguee: https://www.linguee.com/

1. Pencarian hasil translate dengan publisher liinguee:

```
from deep_translator import LingueeTranslator
word = 'good'
translated_word = LingueeTranslator(source='english', target='french').translate(word)
translated_word
```

```
'bon'
```

2. Pada Linguee memberikan rekomendasi yang paling relevan pada saat menggunakanreturn_all=Ture

```
from deep_translator import LingueeTranslator
word = 'good'
translated_word = LingueeTranslator(source='english', target='french').translate(word, return_all=True)
translated_word
```

```
['bon', 'beau', 'droit', 'sain', 'correct', 'sage', 'gentil']
```

3. Implementasi pada 2 kata:

```
#More than one word
from deep_translator import LingueeTranslator
word = 'good person'
translated_word = LingueeTranslator(source='english', target='french').translate(word)
translated_word
```

```
'bonne personne'
```

4. Sedangkan apabila yang diterjemahkan sudah berbentuk klausa dengan kompleksitas makayang akan diterjemahkan hanya kata yang kuat:

```
#More than two words
from deep_translator import LingueeTranslator
word = 'I was looking for you at canteen'
translated_word = LingueeTranslator(source='english', target='french').translate(word)
translated_word
```

```
'cantine'
```

MyMemory

1. The MyMemory translator is the world's largest Translation Memory and it is 100%free to use
2. Link: https://mymemory.translated.net/

1. Untuk menggunakan API dari MyMemory Translator dapat menggunakan perintah berikut:

```
from deep_translator import MyMemoryTranslator
translated = MyMemoryTranslator(source='en', target='id').translate(text='superb')
translated
```

```
'MANTAP!'
```

2. Untuk penerjemahan dalam bentuk kalimat seperti berikut:

```
#Multi words
from deep_translator import MyMemoryTranslator
translated = MyMemoryTranslator(source='en', target='id').translate(text='i am starving, could you give me some cake')
translated
```

```
'Saya lapar, bisakah Anda memberi saya kue?'
```

9.7 POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor

1.	CPL-03	CPMK-01	Lakukanlah Translate dengan menggunakan GoogleTranslator, Linguee, dan MyMemory pada kalimat berikut : 1. English to Indonesia : “Do all the good you can, for all the people you can, in all the ways you can, as long as you can.” 2. English to German, then German to Indonesia : “My name is Mayang and currently study at AhmadDahlan University majoring informatics engineering3rd years.” Scroonshot setiap langakah dan jelasakn	100

			perbedaannya.	

9.8 HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-0	CPMK-0	20%		
2.	Praktik	CPL-0	CPMK-0	30%		
3.	Post-Test	CPL-0	CPMK-0	50%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

PRAKTIKUM 10: FLASK

Pertemuan ke 10

Total Alokasi Waktu: 90 menit
- Materi : 15 menit
- Pre-Test : 15 menit
- Praktikum : 45 menit
- Post-Test : 15 menit

Total Bobot Penilaian : 100%
- Pre-Test : 20 %
- Praktik : 30 %
- Post-Test : 50 %

Pemenuhan CPL dan CPMK:

CPL-03	Mampu menerapkan konsep teoritis bidang area Informatika terkait matematika dasar dan ilmu komputer untuk memodelkan masalah dan meningkatkan produktivitas.
CPMK-03	Mengembangkan aplikasi dengan metode-metode yang sudah dipelajari dalam penyelesaian permasalahan dalam pemrosesan bahasa alami.

1.1 DESKRIPSI CAPAIAN PEMBELAJARAN

Setelah mengikuti praktikum ini mahasiswa diharapkan mampu menerapkan flask.

1.2 INDIKATOR KETERCAPAIAN PEMBELAJARAN

Indikator ketercapaian diukur dengan:

CPL-02	CPMK-03	Kemampuan mahasiswa dalam menerapkan flask

1.3 TEORI PENDUKUNG

Micro tidak berarti semua aplikasi web dapat dijadikan kedalam satu file Python, seperti itu bukan berarti kekurangan dalam fungsionalitas. Micro framework (Flask) bertujuan menjaga core agar tetap simple akan tetapi dapat dikembangkan dalam skala yang besar. Flask tidak ingin membuat banyak aturan untuk seorang developer, seperti contoh database apa yang akan digunakan, aturan template engine apa yang harus digunakan. Sehingga developer bebas memilih apapun segalanya yang dibutuhkan dan apa yang tidak dibutuhkan.

Secara default, didalam Flask tidak tersedia database abstraction layer, validation form atau sesuatu yang lain dimana macam – macam library sudah tersedia dan dapat segera dipakai. Flask mendukung pengembangan untuk penambahan fungsionalitas kebutuhan.

Meskipun flask adalah “micro”, akan tetapi siap digunakan untuk berbagai macam kebutuhan.

Flask adalah web *microframework* dari bahasa python. Flask menyediakan kumpulankode yang dapat digunakan untuk membantu mempercepat pembuatan sebuah website

Template adalah sebuah wadah untuk menampilkan data-data dari flask ke halamanwebsite (html, css, js).

Installasi dependencies:

1. Installing Flask

Buka anaconda promt lalu ketikkan perintah berikut: pip install Flask

2. Installig pymongo

Buka anaconda promt lalu ketikkan perintah berikut: pip install pymongo

1.4 HARDWARE DAN SOFTWARE

Hardware dan software yang digunakan dalam praktikum ini yaitu:

1. Komputer.
2. Bahasa pemrograman Python
3. JupyterNotebook

1.5 PRE-TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-08	CPMK-03	Jelaskan apa itu flask dan template	40
2.	CPL-08	CPMK-03	Sebutkan cara mengimport package flask di python!	10

1.6 LANGKAH PRAKTIKUM

Aturan Penilaian (Total Skor: 100):

No	CPL	CPMK	Pertanyaan	Dokumen Pendukung	Skor
1.	CPL-08	CPMK-03	Selesaikan langkah praktikum	Hasil praktikum	100

Langkah-Langkah Praktikum:

1. Install *library* Flask

```
pip install flask
```

2. *Import package* flask

```
from flask import Flask
```

3. Inisialisasi fungsi Flask()

```
app = Flask(__name__)
```

4. Atur route

```
@app.route('/')
def index():
    return 'Halaman berhasil dibuat'
```

5. Buka terminal, jalankan server flask dengan perintah “python <nama_file>.py”. Kalau berhasil, maka akan diberikan link untuk membuka halaman website.

```
(base) E:\python\flask\latihan-flask>python app.py
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 545-238-950
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

6. Buka link yang diberikan di web browser dan lihat hasilnya

B. *Template*

1. *Import* Flask dan render template

```
from flask import Flask, render_template
```

2. Buat folder Bernama “templates” di direktori yang sama dengan file flask. Lalu buat sebuah file html

3. Inisialisasi fungsi Flask() dan buat route. Di bagian route, kita return halaman html yang sudah kita buat menggunakan render_template()

```python
# Inisialisasi fungsi Flask
app = Flask(__name__)

# root route ('/')
@app.route('/')
def index():
    return render_template('index.html')
```

4. Buka halaman index.html, lalu edit

```html
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width,
  initial-scale=1.0">
  <title>Latihan</title>
</head>
<body>
  <h1>INI LATIHAN FLASK</h1>
  <p>Belajarlah dengan giat</p>
</body>
</html>
```

5. Lihat halaman yang sudah dibuat di web browser

1.7 POST TEST

Jawablah pertanyaan berikut (**Total Skor: 100):**

No	CPL	CPMK	Pertanyaan	Skor
1.	CPL-08	CPMK-03	Buatlah biodata diri minimal 4 data yang ditampilkan	100

1.8 HASIL CAPAIAN PRAKTIKUM

Diisi oleh asisten setelah semua assessment dinilai.

No	Bentuk Assessment	CPL	CPMK	Bobot	Skor (0-100)	Nilai Akhir (Bobot x Skor)
1.	Pre-Test	CPL-0	CPMK-0	20%		
2.	Praktik	CPL-0	CPMK-0	30%		
3.	Post-Test	CPL-0	CPMK-0	50%		
					Total Nilai	

LEMBAR JAWABAN PRE-TEST DAN POST-TEST PRAKTIKUM

Nama : **NIM :**	**Asisten:** **Paraf Asisten:**	**Tanggal:** **Nilai:**

DAFTAR PUSTAKA

1. Jurafsky, D., & Martin, J. H. (2019). Speech and Language Processing (3rd (draft) ed.).
2. Manning, C. D., Manning, C. D., & Schütze, H. (1999). Foundations of statistical natural language processing. MIT press.


LABORATORIUM
TEKNIK INFORMATIKA


UNIVERSITAS AHMAD DAHLAN