# LAPORAN UJIAN AKHIR SEMESTER PEMROGRAMAN BERORIENTASI OBJEK KELOMPOK 1



**DISUSUN OLEH :**

1. Weko Abbror (G1A022025)
2. Citra Azzahra Al-Fatihah (G1A022057)
3. Ridho Herta Putra (G1A022061) Kelas : A

# Nama Asisten Dosen :

1. Randi Julian S (G1A019066)

# Dosen Pengampu :

* 1. Andang Wijanarko, S.Kom, M.Kom.
  2. Arie Vatresia, S.T., M.TI, Ph.D.

# PROGRAM STUDI INFORMATIKA FAKULTAS TEKNIK UNIVERSITAS BENGKULU

**2023**

# LANDASAN TEORI

Pemrograman Python adalah proses membuat program komputer menggunakan bahasa pemrograman Python. Python adalah bahasa pemrograman tingkat tinggi yang dikembangkan pada tahun 1991 oleh Guido van Rossum. Python dirancang dengan fokus pada keterbacaan kode dan kesederhanaan sintaks, yang memungkinkan pengembang untuk menulis kode yang lebih mudah dipahami dan dijalankan. Dalam pemrograman Python, Anda menggunakan sintaks dan struktur bahasa Python untuk membuat serangkaian instruksi yang dieksekusi oleh komputer. Python mendukung paradigma pemrograman yang beragam, termasuk pemrograman prosedural, pemrograman berorientasi objek, dan pemrograman fungsional. Bahasa ini juga memiliki sejumlah besar pustaka dan modul yang siap pakai, yang memudahkan pengembangan aplikasi yang kompleks. Salah satu fitur Python yang populer adalah penanganan otomatis pengelolaan memori.

Python menggunakan pengumpulan sampah (garbage collection) yang membebaskan pengembang dari kewajiban untuk secara manual mengelola alokasi dan dealokasi memori. Python digunakan secara luas dalam berbagai bidang, termasuk pengembangan perangkat lunak, analisis data, pembelajaran mesin (machine learning), kecerdasan buatan (artificial intelligence), pengembangan web, dan banyak lagi. Keunggulan Python dalam hal keterbacaan kode dan produktivitas membuatnya menjadi salah satu bahasa pemrograman yang sangat populer di kalangan pengembang. Dalam pemrograman Python, Anda dapat menggunakan berbagai alat dan lingkungan pengembangan yang tersedia, seperti IDLE (Integrated Development and Learning Environment), Jupyter Notebook, PyCharm, dan lainnya, untuk menulis, menguji, dan menjalankan kode Python Anda.

Python dikembangkan oleh seorang programmer bernama Guido van Rossum pada akhir 1980-an dan awal 1990-an. Guido van Rossum merancang Python sebagai bahasa pemrograman yang mudah dipahami dan digunakan oleh pemula. Dia terinspirasi oleh bahasa pemrograman ABC yang dikembangkan di Centrum Wiskunde & Informatica (CWI) di Belanda. Python pertama kali dirilis pada tahun 1991 dengan versi 0.9.0. Nama Python sendiri diambil dari judul serial komedi Inggris yang populer, "Monty Python's Flying Circus". Guido van Rossum adalah penggemar serial tersebut dan memilih nama "Python" untuk bahasanya karena dia ingin nama yang unik, cerdas, dan sedikit aneh.

Pada awalnya, Python dikembangkan sebagai bahasa pemrograman umum yang fokus pada keterbacaan kode. Namun, seiring berjalannya waktu, Python berkembang menjadi

bahasa pemrograman yang serbaguna dengan dukungan yang luas untuk berbagai bidang seperti pengembangan web, analisis data, kecerdasan buatan, dan lainnya. Pada tahun 2000, Python 2.0 dirilis dengan berbagai perbaikan dan fitur baru. Versi Python 2.0 menjadi versi yang paling banyak digunakan untuk waktu yang lama.

Pada tahun 2008, Python 3.0 dirilis dengan pembaruan yang signifikan, namun tidak sepenuhnya kompatibel dengan Python 2.x. Ini menghasilkan adopsi yang lambat dari Python 3 oleh komunitas pengembang. Pada tahun 2020, dukungan untuk Python 2.x dihentikan dan pengembang mendorong penggunaan Python 3.x yang lebih baru. Python 3.x terus mengalami pembaruan dan perbaikan, dengan versi terbaru pada saat pengetian saya adalah Python 3.9. Python 3.x memiliki perbaikan performa, sintaks yang lebih bersih, dukungan untuk fitur-fitur baru, dan keamanan yang ditingkatkan. Sejak awal kemunculannya, Python telah menjadi salah satu bahasa pemrograman yang paling populer di dunia. Python memiliki komunitas pengembang yang aktif dan ramah, serta banyak pustaka dan modul yang tersedia yang menjadikannya bahasa pemrograman yang kuat dan fleksibel.

Python memiliki berbagai fitur yang membuatnya menjadi bahasa pemrograman yang populer dan kuat. Berikut adalah beberapa fitur utama dari Python:

1. Mudah dipelajari dan mudah dibaca: Python dirancang dengan fokus pada keterbacaan kode. Sintaks yang bersih dan penggunaan indentasi sebagai bagian penting dari struktur kode membuatnya mudah dipahami, bahkan oleh pemula.
2. Bahasa pemrograman tingkat tinggi: Python adalah bahasa pemrograman tingkat tinggi, yang berarti Anda dapat mengekspresikan konsep pemrograman kompleks dengan kode yang lebih singkat dan lebih mudah dipahami. Ini memungkinkan pengembang untuk lebih fokus pada logika bisnis daripada detail implementasi.
3. Interpretatif: Python adalah bahasa pemrograman interpretatif, yang berarti kode Python dieksekusi secara langsung oleh interpreter tanpa perlu dikompilasi terlebih dahulu. Ini memungkinkan pengembang untuk menguji dan menjalankan kode dengan cepat tanpa proses kompilasi yang rumit.
4. Multi-paradigma: Python mendukung paradigma pemrograman yang beragam, termasuk pemrograman prosedural, pemrograman berorientasi objek, dan pemrograman fungsional. Ini memberikan fleksibilitas dalam memilih pendekatan pemrograman yang paling sesuai dengan kebutuhan proyek.
5. Pustaka dan modul yang kaya: Python memiliki ekosistem yang luas dengan banyak pustaka dan modul yang tersedia. Pustaka-pustaka ini menyediakan fungsionalitas tambahan yang siap pakai, seperti pemrosesan string, pengembangan web, analisis data, kecerdasan buatan, dan banyak lagi. Pengguna dapat mengimpor modul dan menggunakan kode yang sudah ada untuk mempercepat pengembangan.
6. Platform independen: Python dapat berjalan pada berbagai platform, termasuk Windows, macOS, Linux, dan sistem operasi lainnya. Kode Python yang ditulis dapat dijalankan tanpa perubahan pada berbagai lingkungan.
7. Pengelolaan memori otomatis: Python menggunakan pengumpulan sampah (garbage collection) yang otomatis mengelola alokasi dan dealokasi memori. Hal ini mengurangi beban pengembang dalam mengelola memori secara manual dan mengurangi risiko terjadinya kebocoran memori.
8. Komunitas yang aktif: Python memiliki komunitas pengembang yang besar dan aktif. Ada banyak sumber daya, forum, dan dokumentasi yang tersedia untuk membantu pengembang Python, sehingga memudahkan belajar dan mengatasi masalah.

Fitur-fitur ini membuat Python menjadi bahasa pemrograman yang kuat dan populer dalam berbagai bidang pengembangan perangkat lunak, analisis data, pembelajaran mesin, kecerdasan buatan, pengembangan web, dan lainnya.

Pemrograman Python memiliki sejumlah kelebihan yang membuatnya menjadi salah satu bahasa pemrograman yang paling populer dan digunakan secara luas. Berikut adalah beberapa kelebihan utama dari Python:

1. Mudah dipelajari dan digunakan: Python dirancang dengan fokus pada keterbacaan kode dan kesederhanaan sintaks. Ini membuatnya menjadi bahasa yang mudah dipelajari, baik bagi pemula maupun pengembang yang berpengalaman. Python memiliki sintaks yang bersih dan menggunakan indentasi untuk menentukan blok kode, sehingga memudahkan dalam membaca dan memahami kode.
2. Produktivitas yang tinggi: Python memungkinkan pengembang untuk menulis kode dengan lebih sedikit baris dibandingkan dengan bahasa pemrograman lain. Python memiliki banyak pustaka dan modul yang tersedia yang menyediakan fungsionalitas yang kaya dan siap pakai. Pengguna dapat menggunakan modul-modul ini untuk menghemat waktu dan upaya dalam mengembangkan aplikasi.
3. Dukungan untuk berbagai platform: Python dapat berjalan di berbagai platform, termasuk Windows, macOS, Linux, dan sistem operasi lainnya. Kode Python yang ditulis dapat dijalankan tanpa perubahan pada berbagai lingkungan. Ini memberikan fleksibilitas bagi pengembang untuk mengembangkan aplikasi yang dapat dijalankan di berbagai sistem operasi.
4. Ekosistem yang luas: Python memiliki ekosistem yang luas dengan banyak pustaka dan modul yang tersedia. Pustaka-pustaka ini menyediakan berbagai fungsionalitas tambahan seperti pemrosesan string, pengembangan web, analisis data, kecerdasan buatan, dan masih banyak lagi. Dengan adanya ekosistem yang kaya, pengembang dapat mempercepat pengembangan aplikasi dengan menggunakan kode yang sudah ada.
5. Integrasi dengan bahasa lain: Python dapat dengan mudah diintegrasikan dengan bahasa pemrograman lain seperti C, C++, dan Java. Hal ini memungkinkan pengembang untuk menggunakan kode yang sudah ada dalam bahasa lain atau memperluas fungsionalitas Python dengan menggunakan pustaka dari bahasa lain.
6. Dukungan untuk pemrograman berorientasi objek: Python mendukung paradigma pemrograman berorientasi objek (OOP) dengan fitur-fitur seperti pewarisan, polimorfisme, dan enkapsulasi. Ini memungkinkan pengembang untuk mengorganisir kode dalam objek-objek yang dapat digunakan kembali dan mempermudah pengembangan aplikasi yang kompleks.
7. Pengolahan data dan analisis: Python memiliki pustaka yang kuat untuk pengolahan data dan analisis seperti NumPy, pandas, dan matplotlib. Ini membuat Python menjadi pilihan yang populer untuk analisis data, pembelajaran mesin, dan kecerdasan buatan.
8. Komunitas yang aktif: Python memiliki komunitas pengembang yang besar, aktif, dan ramah. Komunitas ini menyediakan dukungan, sumber daya, dan dokumentasi yang kaya. Pengembang Python dapat dengan mudah mendapatkan bantuan, berbagi pengetahuan, dan terlibat dalam proyek bersama.

Kelebihan-kelebihan ini membuat Python menjadi bahasa pemrograman yang sangat populer diberbagai bidang pengembangan perangkat lunak, analisis data, kecerdasan buatan, dan pengembangan web.

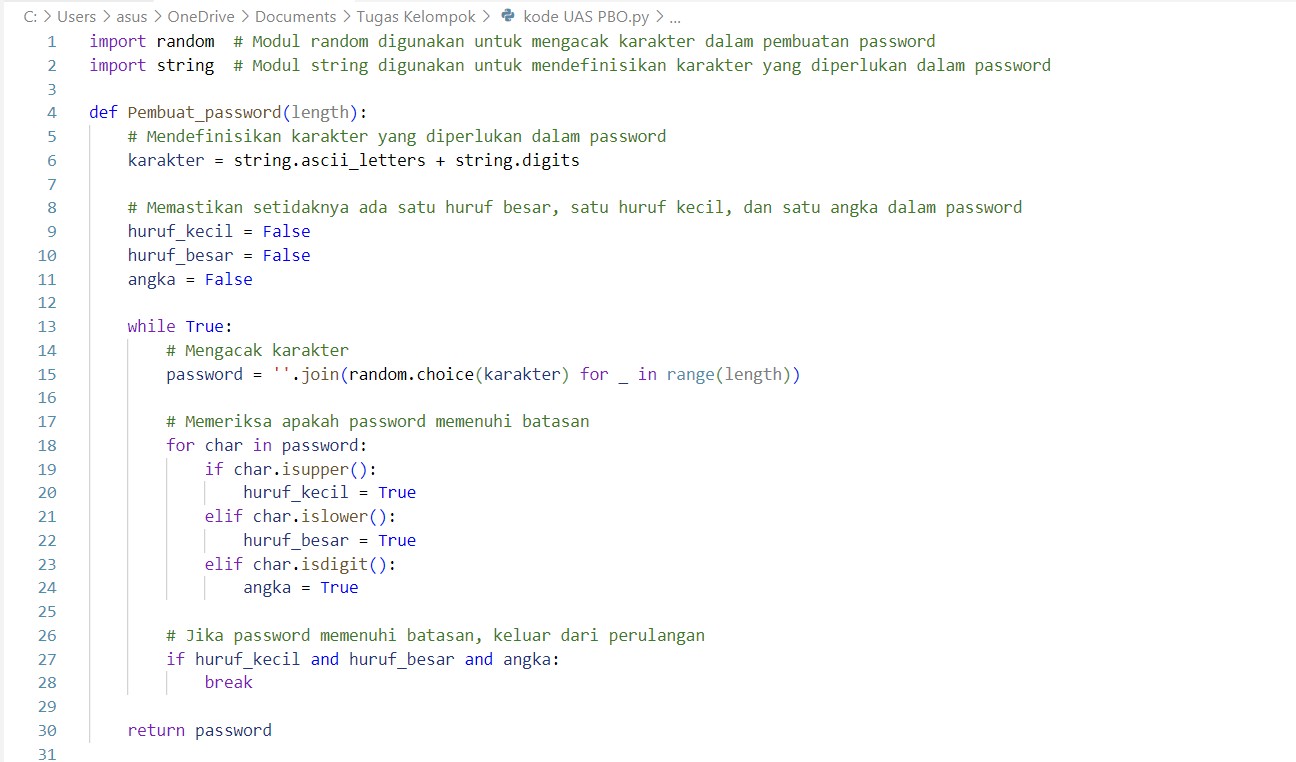
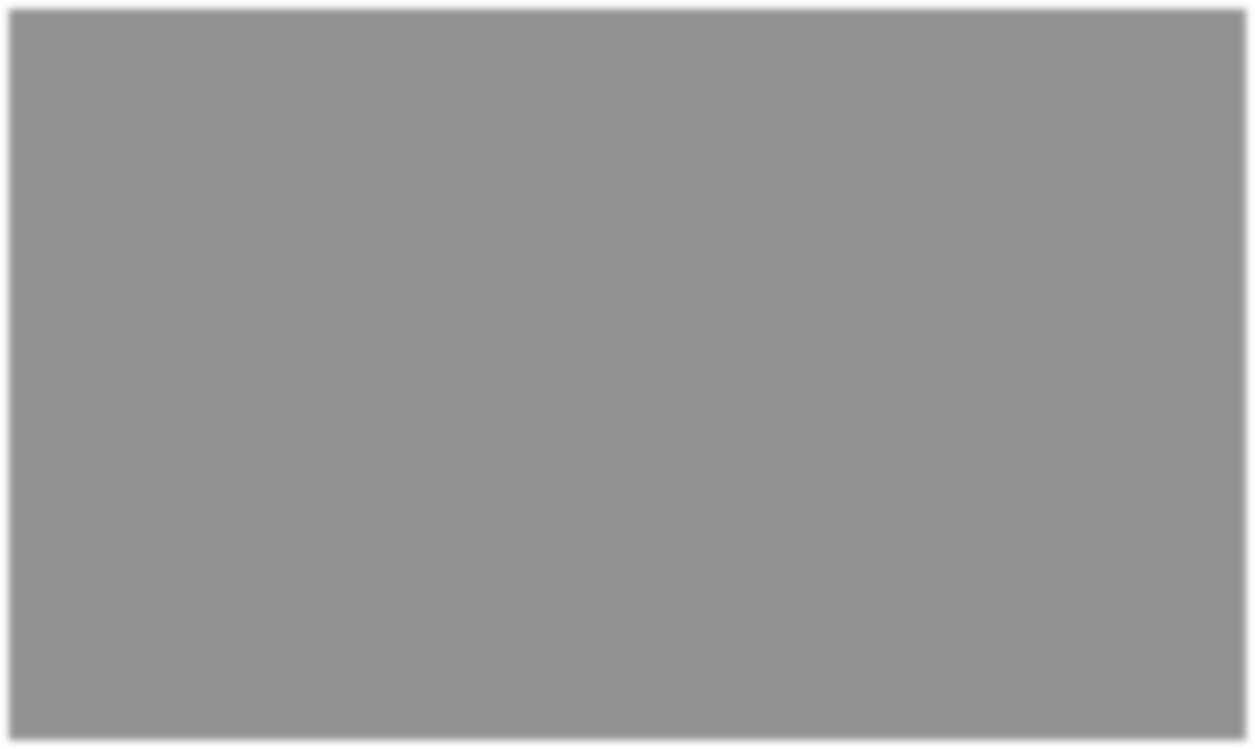
Meskipun Python memiliki banyak kelebihan, seperti yang telah dijelaskan sebelumnya, ada juga beberapa kekurangan yang perlu dipertimbangkan. Berikut adalah beberapa kekurangan pemrograman Python:

1. Performa relatif lambat: Meskipun Python telah mengalami peningkatan performa dari versi ke versi, Python secara umum dianggap memiliki performa yang lebih lambat dibandingkan dengan bahasa pemrograman lain seperti C atau C++. Hal ini terutama berlaku dalam kasus penggunaan CPU-bound yang membutuhkan komputasi intensif.
2. Penggunaan memori yang lebih tinggi: Python menggunakan pengelolaan memori otomatis melalui pengumpulan sampah (garbage collection), yang dapat menyebabkan penggunaan memori yang lebih tinggi dibandingkan dengan bahasa pemrograman lain yang mengharuskan pengelolaan memori manual. Hal ini dapat menjadi masalah dalam aplikasi yang membutuhkan penggunaan memori yang efisien.
3. GIL (Global Interpreter Lock): Python menggunakan GIL, yang merupakan fitur di interpreter CPython (implementasi referensi Python), untuk memastikan keamanan thread dalam lingkungan multi-threading. Namun, GIL juga mengakibatkan hanya satu thread yang dapat dijalankan pada satu waktu, sehingga dapat membatasi kinerja Python dalam aplikasi yang mengandalkan multi-threading untuk skala yang lebih besar.
4. Keterbatasan dalam pengembangan mobile: Python tidak sepenuhnya dioptimalkan untuk pengembangan aplikasi mobile. Meskipun ada pustaka seperti Kivy dan BeeWare yang memungkinkan pengembangan aplikasi mobile dengan Python, namun ekosistem Python masih lebih fokus pada pengembangan web, analisis data, dan kecerdasan buatan.
5. Kompatibilitas antara Python 2 dan Python 3: Pada awalnya, ada perubahan yang signifikan antara Python 2 dan Python 3, yang menyebabkan beberapa masalah kompatibilitas. Ini berarti kode yang ditulis dengan Python 2 mungkin tidak berjalan langsung di Python 3 tanpa modifikasi. Namun, dukungan untuk Python 2 telah dihentikan pada tahun 2020, dan sekarang disarankan untuk beralih ke Python 3.

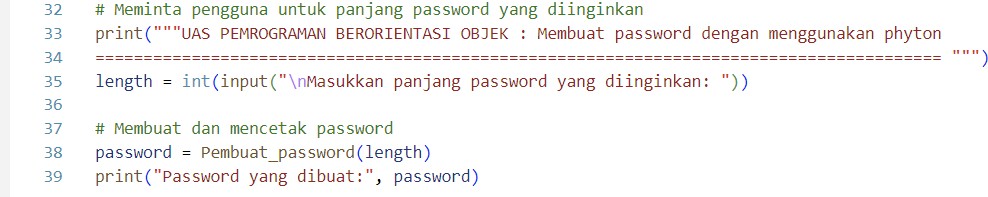
Meskipun ada kekurangan-kekurangan ini, Python tetap menjadi salah satu bahasa pemrograman yang paling populer dan digunakan secara luas di berbagai industri. Kekurangan-kekurangan tersebut seringkali dapat diatasi dengan pemilihan strategi pengembangan yang tepat dan menggunakan pustaka atau alat tambahan yang sesuai.

# HASIL DAN PEMBAHASAN

**1. Membuat password dengan menggunakan Python: Dengan menggunakan skrip python ini kita dapat membantu dalam pembuatan password dengan beberapa batasan seperti harus campuran huruf besar, huruf kecil, simbol, dan angka.**



Gambar 1.1 Print out source code



Gambar 1.2 Lanjutan printout source code

Source Code:

import random # Modul random digunakan untuk mengacak karakter dalam pembuatan password

import string # Modul string digunakan untuk mendefinisikan karakter yang diperlukan dalam password

def Pembuat\_password(length):

# Mendefinisikan karakter yang diperlukan dalam password karakter = string.ascii\_letters + string.digits

# Memastikan setidaknya ada satu huruf besar, satu huruf kecil, dan satu angka dalam password

huruf\_kecil = False huruf\_besar = False angka = False

while True:

# Mengacak karakter

password = ''.join(random.choice(karakter) for \_ in range(length))

# Memeriksa apakah password memenuhi batasan for char in password:

if char.isupper(): huruf\_kecil = True

elif char.islower(): huruf\_besar = True

elif char.isdigit(): angka = True

# Jika password memenuhi batasan, keluar dari perulangan if huruf\_kecil and huruf\_besar and angka:

break

return password

# Meminta pengguna untuk panjang password yang diinginkan

print("""UAS PEMROGRAMAN BERORIENTASI OBJEK : Membuat password dengan menggunakan phyton

===================================================================

===================== """)

length = int(input("\nMasukkan panjang password yang diinginkan: "))

# Membuat dan mencetak password

password = Pembuat\_password(length) print("Password yang dibuat:", password)

Penjelasan Source Code:

Program Python di atas adalah sebuah script untuk membuat password acak dengan menggunakan modul `random` dan `string`. Berikut adalah penjelasan dari setiap bagian program tersebut:

1. Pada baris pertama, dilakukan import modul `random` untuk mengacak karakter dalam pembuatan password.
2. Pada baris kedua, dilakukan import modul `string` untuk mendefinisikan karakter yang diperlukan dalam password.
3. Kemudian, fungsi `Pembuat\_password` didefinisikan dengan menerima satu parameter `length` yang merupakan panjang password yang diinginkan.
4. Pada baris ke-7, variabel `karakter` didefinisikan sebagai gabungan dari

`string.ascii\_letters` (huruf kecil dan huruf besar) dan `string.digits` (angka), sehingga karakter yang mungkin dalam password adalah huruf kecil, huruf besar, dan angka.

1. Selanjutnya, variabel `huruf\_kecil`, `huruf\_besar`, dan `angka` diinisialisasi sebagai

`False`. Variabel-variabel ini akan digunakan untuk memastikan bahwa setidaknya ada satu huruf kecil, satu huruf besar, dan satu angka dalam password yang dibuat.

1. Pada baris ke-10, dilakukan perulangan `while True` yang berarti akan terus berjalan sampai kondisi tertentu terpenuhi.
2. Di dalam perulangan, password diacak menggunakan `random.choice(karakter)` dan diulang sebanyak `length`. Hasilnya digabungkan menggunakan `''.join()` sehingga menghasilkan password acak.
3. Setelah password diacak, dilakukan pemeriksaan karakter-karakter dalam password menggunakan perulangan `for`. Jika ditemukan karakter huruf kecil, variabel

`huruf\_kecil` diubah menjadi `True`. Jika ditemukan karakter huruf besar, variabel

`huruf\_besar` diubah menjadi `True`. Jika ditemukan karakter angka, variabel `angka` diubah menjadi `True`.

1. Jika password memenuhi batasan yaitu memiliki setidaknya satu huruf kecil, satu huruf besar, dan satu angka, maka perulangan dihentikan dengan menggunakan

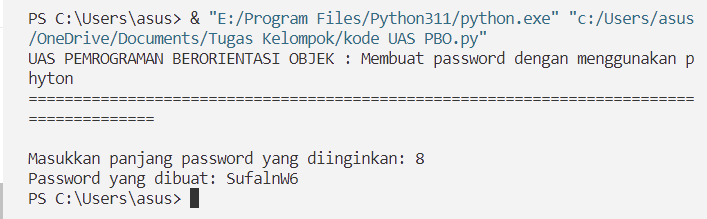
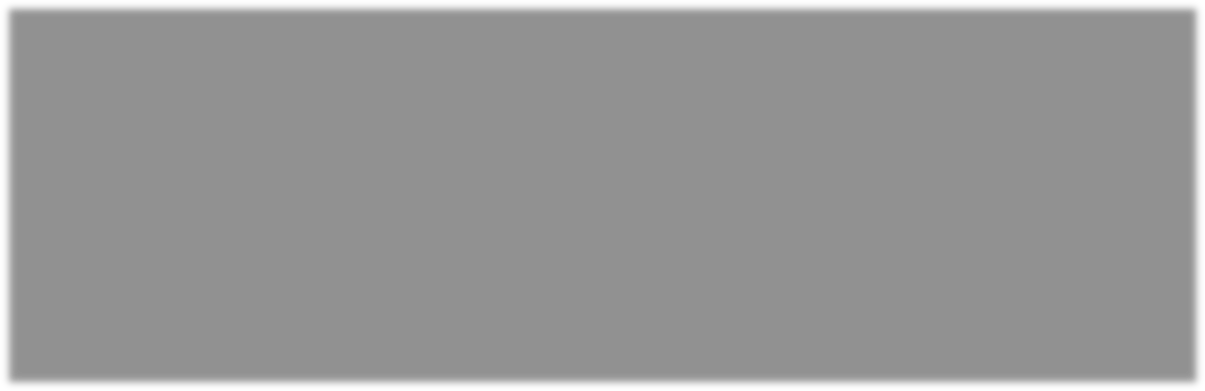
`break`.

1. Password yang memenuhi batasan kemudian dikembalikan sebagai hasil dari fungsi

`Pembuat\_password`.

1. Pada baris ke-20, program akan menampilkan pesan berupa instruksi kepada pengguna untuk memasukkan panjang password yang diinginkan.
2. Pengguna diminta untuk memasukkan panjang password melalui input.
3. Pada baris ke-23, fungsi `Pembuat\_password` dipanggil dengan mengirimkan panjang password yang dimasukkan pengguna sebagai argumen.
4. Password acak yang dibuat kemudian dicetak.
5. Program selesai.

Dengan menggunakan program ini, pengguna dapat membuat password acak dengan panjang yang diinginkan dan memastikan bahwa password tersebut memiliki setidaknya satu huruf kecil, satu huruf besar, dan satu angka.



Gambar 1.3 Print out ouput source code

Penjelasan Ouput:

Output yang dihasilkan adalah bentuk password sesuai jumlah karakter atau panjang karakter yang diinginkan. Seperti contoh pada output di atas, ketika kita meminta 8 panjang karakter, maka program akan menghasilkan output password berupa 8 karakter Password yang terdiri dari kombinasi huruf kapital, huruf kecil, dan angka.

# 2. Penjelasan Penerapan Konsep OOP pada program di atas.

Dalam program Python di atas, terdapat beberapa konsep penerapan pemrograman berorientasi objek (OOP). Berikut adalah konsep-konsep tersebut:

1. Modul: Modul `random` dan `string` yang diimpor digunakan untuk mengorganisir dan mengelompokkan fungsi-fungsi terkait dalam satu tempat. Dalam OOP, modul ini dapat dianggap sebagai contoh dari modul yang menyediakan fungsionalitas yang dapat digunakan secara terpisah.
2. Fungsi: Fungsi `Pembuat\_password` digunakan untuk membuat password dengan panjang tertentu. Fungsi ini dapat dianggap sebagai metode dari suatu kelas, meskipun dalam program ini tidak ada penggunaan kelas.
3. Encapsulation: Fungsi `Pembuat\_password` menerima panjang password sebagai argumen dan mengembalikan password yang dihasilkan. Ini menunjukkan penggunaan konsep encapsulation, di mana detail implementasi untuk membuat password tersembunyi dari pengguna dan hanya fungsi yang mengatur akses ke detail tersebut.
4. Looping: Dalam program ini, terdapat penggunaan perulangan `while True` untuk memastikan bahwa password yang dibuat memenuhi batasan yang diperlukan. Hal ini memanfaatkan konsep looping dalam OOP untuk melakukan iterasi dan mengulangi langkah-langkah tertentu sampai kondisi yang diinginkan terpenuhi.

Meskipun ada beberapa elemen yang mencerminkan konsep OOP, seperti penggunaan fungsi dan pemisahan fungsionalitas dalam modul-modul, program ini secara keseluruhan tidak sepenuhnya mengikuti paradigma OOP secara formal dengan penggunaan kelas dan objek. Namun, elemen-elemen tersebut dapat diterapkan lebih lanjut dengan pembuatan kelas yang memodelkan konsep pembuatan password dan mengorganisir fungsionalitas terkait di dalamnya.

# KESIMPULAN DAN SARAN

1. **Kesimpulan**

Tugas UAS PBO ini memberikan kesempatan kepada kami untuk mengembangkan keterampilan pemrograman Python Anda dan memahami praktik terbaik dalam pembuatan password yang aman. Dalam kesimpulan ini, kita akan melihat lebih dalam mengenai pentingnya keamanan password dan memberikan saran tambahan untuk meningkatkan kualitas skrip Python yang Anda buat.

# Saran

* 1. Pahami spesifikasi dan batasan tugas dengan jelas.
  2. Gunakan modul random untuk menghasilkan nilai acak dan modul string untuk karakter password.
  3. Buat fungsi validasi untuk memeriksa apakah password memenuhi kriteria keamanan.
  4. Gunakan konsep looping untuk membangun password secara bertahap.
  5. Lakukan uji coba dan pastikan password memenuhi persyaratan keamanan.
  6. Dokumentasikan kode dengan komentar yang jelas dan ikuti konvensi penamaan yang baik.
  7. Perbarui skrip secara berkala dan tingkatkan keamanannya.
  8. Pertimbangkan kebutuhan pengguna untuk mencapai keseimbangan antara keamanan dan kenyamanan.

# DAFTAR PUSTAKA

Lutz, M., & Ascher, D. (2016). Matplotlib A Versatile Plotting Library. Computing Science & Engineering, 13(3), 9 – 12.

Romzi, M., & Kurniawan, B. ( 2020 ). Pembelajaran Pemrograman Python dengan Pendekatan Logika Algoritma. Jurnal Teknik Informatika Mahakarya, 37 – 44.

Saragih, R. Rotuahta. (2018). Pemrograman dan Bahasa Pemrograman.

Van Rossum, G. (1995) Python Tutorial. Centrum voor Wiskunde en Informatica (CWI).

Amsterdam, The Netherlands.