Nama : GALIH RIDHO UTOMO

NIM : 4211421036

Sejarah Python

Python adalah Bahasa pemprogaman yang tidak rumit dan kuat yang memberikan kekuatan dan kompleksitas Bahasa kompilas trandisional Bersama dengan kemudahan penggunaan dan skrip yang lebih sederhana dan Bahasa yang ditaksirkan W. Chun (2001). Bahasa pemprogaman Python adalah Bahasa kompilas trandisional yang merupakan penerus dari Bahasa pemprogaman ABC. Bahasa pemprogaman pyhton mempunyai sejarah yaitu ditemukan pada akhir tahun 1980-an oleh Guido van Rossum di Centrum Wiskunde & Informatica (CWI) di Belanda, yang merupakan terinspirasi oleh SETL, yang mampu menangani pengecualian dan berinteraksi dengan sistem operasi Amoeba. Pemprogaman ini baru mengimplementasikannya pada Desember 1989. G. Van Rossum (2007). Pemprogaman Python mempunyai versi dan keunggulan setiap versi yang diberikan diantara

1. Python 2.0 dirilis pada 16 Oktober 2000, dengan banyak fitur baru utama, termasuk pengumpul sampah pendeteksi siklus (selain penghitungan referensi) untuk manajemen memori dan dukungan untuk Unicode.

2. Python 3.0 dirilis pada 3 Desember 2008. Itu adalah revisi besar dari bahasa yang tidak sepenuhnya kompatibel ke belakang. Banyak fitur utamanya di-backport ke seri versi Python 2.6.x dan 2.7.x. Rilis Python 3 menyertakan utilitas 2 ke 3, yang mengotomatiskan terjemahan kode Python 2 ke Python 3.

3. Tanggal akhir Python 2.7 awalnya ditetapkan pada 2015 kemudian ditunda hingga 2020 karena khawatir bahwa sebagian besar kode yang ada tidak dapat dengan mudah di-forward ke Python 3. Tidak ada lagi patch keamanan atau peningkatan lain yang akan dirilis untuk itu. Dengan akhir masa pakai Python 2, hanya Python 3.6.x dan yang lebih baru yang didukung.

4. Python 3.9.2 dan 3.8.8 dipercepat karena semua versi Python (termasuk 2.7) memiliki masalah keamanan yang mengarah ke kemungkinan eksekusi kode jarak jauh dan keracunan cache web.

Python adalah jenis baru dari bahasa *scripting*, dan seperti kebanyakan bahasa scripting dibangun di sekitar penerjemah. Banyak skrip tradisional dan bahasa yang ditafsirkan telah mengorbankan kejelasan sintaksis untuk menyederhanakan konstruksi pengurai pertimbangkan misalnya sintaks menyakitkan yang diperlukan untuk menghitung nilai ekspresi sederhana seperti a+b*c di Lisp, Smalltalk atau Bourne shell. (van Rossum, 1993). Python pada dasarnya bersifat modular. Kernel sangat kecil dan dapat diperpanjang dengan mengimpor modul ekstensi. Distribusi Python mencakup perpustakaan ekstensi standar yang beragam (beberapa ditulis dengan

Python, yang lain dalam C atau C++) untuk operasi mulai dari manipulasi string dan ekspresi reguler seperti Perl, hingga Grafik Generator Antarmuka Pengguna (GUI) dan termasuk utilitas terkait web, layanan sistem operasi, debugging dan alat profil, dll. Sanner, M. F. (1999). Berikut merupakan contoh dari pemprogaman pyhton yang saya buat sebagai gambaran mengenai suasana Python

```
import matplotlib.pyplot as plt
import numpy as np

def operation(vy):
```

- Pernyataan import adalah "setara moral" Python dari pernyataan #include C. Fungsi dan variabel yang didefinisikan dalam modul yang diimpor dapat direferensikan dengan mengawalinya dengan nama modul.
- Sebagian besar kode program terkandung dalam definisi fungsi. Ini adalah cara umum penataan
  program Python yang lebih besar, karena memungkinkan untuk menulis kode secara top-down. Fungsinya operation() dijalankan oleh panggilan pada baris terakhir program.
- Ekspresi sys.argv[1:] mengiris daftar sys.argv: ia mengembalikan daftar baru dengan elemen pertama dihilangkan.

Setelah mendapatkan gambaran mengenai pemprogaman Python berikut adalah penyataan yang melengkap dan bagian fungsi dalam pyhton, Smith, dkk (2004)

1. *The assignment statement*, menggunakan tanda sama dengan tunggal “=”.
2. *The if statement*, yang secara kondisional mengeksekusi blok kode, bersama dengan else dan elif (kontraksi dari *else-if*).
3. *The for statement*, yang mengulangi objek yang dapat diubah, menangkap setiap elemen ke variabel lokal untuk digunakan oleh blok terlampir.
4. *The while statement*, yang mengeksekusi blok kode selama kondisinya benar.
5. *The try statement*, yang memungkinkan pengecualian yang diangkat dalam blok kode terlampir untuk ditangkap dan ditangani oleh klausa pengecualian hal itu juga memastikan bahwa kode pembersihan di blok akhirnya akan selalu dijalankan terlepas dari bagaimana blok keluar.
6. *The raise statement*, digunakan untuk menaikkan pengecualian tertentu atau menaikkan kembali pengecualian yang tertangkap.
7. *The class statement*, yang mengeksekusi blok kode dan melampirkan namespace lokalnya ke kelas, untuk digunakan dalam pemrograman berorientasi objek.
8. *The def statement*, yang mendefinisikan fungsi atau metode.
9. *The with statement*, yang membungkus blok kode dalam manajer konteks (misalnya, memperoleh kunci sebelum blok kode dijalankan dan melepaskan kunci setelahnya, atau membuka file dan kemudian menutupnya), memungkinkan resource-acquisition-is-initialization (RAII) seperti perilaku dan menggantikan idiom percobaan atau akhirnya yang umum.
10. *The break statement*, keluar dari loop.
11. *The continue statement*, melewatkan iterasi ini dan melanjutkan dengan item berikutnya.

12. *The del*, menghapus variabel, yang berarti referensi dari nama ke nilai dihapus dan mencoba menggunakan variabel itu akan menyebabkan kesalahan. Variabel yang dihapus dapat dipindahkan.
13. *The pass statement*, yang berfungsi sebagai NOP. Secara sintaksis diperlukan untuk membuat blok kode kosong.
14. *The assert statement*, digunakan selama debugging untuk memeriksa kondisi yang harus diterapkan.
15. *The yield statement*, yang mengembalikan nilai dari fungsi generator dan hasil juga merupakan operator. Formulir ini digunakan untuk mengimplementasikan coroutine.
16. *The return statement*, digunakan untuk mengembalikan nilai dari suatu fungsi.
17. *The import statement*, yang digunakan untuk mengimpor modul yang fungsi atau variabelnya dapat digunakan dalam program saat ini.

Daftar Pusaka

Chun, W. (2001). *Core python programming* (Vol. 1). Prentice Hall Professional.

Sanner, M. F. (1999). Python: a programming language for software integration and development. *J Mol Graph Model*, *17*(1), 57-61.

Smith, Kevin D.; Jewett, Jim J.; Montanaro, Skip; Baxter, Anthony (2 September 2004). *"PEP 318 – Decorators for Functions and Methods".* Python Enhancement Proposals. Python

Van Rossum, G. (2007, June). Python Programming language. In USENIX annual technical conference (Vol. 41, No. 1, pp. 1-36).

van Rossum, G. (1993). *Python for Unix/C Programmers An Introduction to Python for UNIX/C Programmers*.