
# Python: Destructor

Rudiman, S.Kom.,M.Sc.
Teknik Informatika FST-UMKT

**Nurul Huda – Jago Goding**

# Apa itu Destruktor?

- Destruktor pada python adalah sebuah fungsi yang akan dipanggil ketika suatu objek dihancurkan (destroyed) [1], atau dalam bahasa yang lebih sederhana: ketika suatu objek di-delete.
- Jadi bisa kita katakan bahwa destruktor adalah kebalikan dari konstruktor [2]. Jika konstruktor dipanggil saat ketika sebuah objek dibuat / diinstantiasi, maka destruktor akan dipanggil ketika sebuah objek dihapus atau ketika program selesai berjalan

# Apa Fungsi Adanya Destruktor?

- Fungsi adanya destruktor adalah untuk melakukan final cleaning up atau “bersih-bersih” terakhir sebelum sebuah objek benar-benar dihapus dari memori [3].
- Yang dimaksud “bersih-bersih” di sini bisa berupa: menutup koneksi database, menutup file yang sedang terbuka, atau mungkin menghapus cache selama aplikasi dijalankan.

# Mengenal Perintah del

- Sebelum kita membahas tentang destruktor, mari kita pelajari dulu bagaimana cara menghapus suatu variabel pada python.
- Cara menghapusnya adalah menggunakan perintah del. Perhatikan contoh berikut:

Pada contoh di atas kita bisa lihat bahwa setelah variabel a kita hapus, ia sudah tidak ada lagi di memori.

NB: meskipun tidak dihapus, semua variabel atau objek yang kita definisikan akan tetap terhapus secara otomatis ketika sudah kehilangan referensi atau ketika program berhenti berjalan.
Hal ini terjadi karena python memiliki fitur garbage collector yang menangani alokasi memori dengan baik.

```
>>> a = 3
>>> print(a)
3
>>> del a
>>> print(a)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
NameError: name 'a' is not defined
>>>
```

# Membuat Destruktor

- Tepat ketika sebuah variabel akan dihapus dari memori, sistem akan menjalankan proses destruksi.
- Kita bisa mendefinisikan aksi apa yang harus dilakukan pada fase ini dengan mengimplementasikan fungsi destruktor.
- Fungsi destruktor pada python bernama __del__(), ia merupakan sebuah magic method yang proses pemanggilannya dilakukan secara otomatis oleh sistem.
- Seperti biasa, mari kita praktikkan secara langsung.

# Contoh 1: Desktrutor + hapus variabel secara manual

- Contoh yang pertama kita akan membuat kelas bernama Mahasiswa dengan satu buah atribut yaitu atribut nama. Kita akan coba menerapkan fungsi __del__().
- Perhatikan kode program berikut:

Dari output di atas kita bisa lihat bahwa fungsi __init__() akan dipanggil ketika proses konstruksi terjadi, dan sebaliknya fungsi __del__() akan dipanggil ketika proses destruksi terjadi.

```
class Mahasiswa: def __init__(self, nama):
self.nama = nama print(f'mahasiswa
{self.nama} dibuat') def __del__(self):
print(f'mahasiswa {self.nama} dihapus')
Lalu buat instan kemudian hapus:
budi = Mahasiswa('budi') andi =
Mahasiswa('andi') del andi del budi
Output:
mahasiswa budi dibuat mahasiswa andi dibuat
mahasiswa andi dihapus mahasiswa budi
dihapus
```

# Contoh 2: Destruktor + biarkan program selesai

- Contoh berikutnya masih dengan kelas Mahasiswa, akan tetapi kali ini kita akan biarkan program selesai dengan sendirinya. Maka proses destruksi akan dilakukan tepat ketika program akan selesai. Perhatikan kode program berikut:

```
class Mahasiswa:
  def __init__(self, nama):
    self.nama = nama
    print(f'mahasiswa {self.nama} dibuat')

  def __del__(self):
    print(f'mahasiswa {self.nama} dihapus')

print('Halo dunia')
budi = Mahasiswa('budi')
print('Halo semuanya')
andi = Mahasiswa('andi')
print('1 + 1 = 2')
print('3 + 3 = 6')
```

Output:

```
Halo dunia
mahasiswa budi dibuat
Halo semuanya
mahasiswa andi dibuat
1 + 1 = 2
3 + 3 = 6
mahasiswa budi dihapus
mahasiswa andi dihapus
```

Pada program disamping, kita tidak melakukan proses penghapusan variabel secara manual. Kita hanya membiarkan program mengalir begitu saja hingga selesai. Dan tepat sebelum program benar-benar selesai, fungsi destruktor akan dipanggil

# Contoh 3: Destruktor + garbage collector

- Contoh berikutnya adalah kita akan menghapus referensi suatu variabel terhadap suatu objek.
- Kita tahu bahwa pada python, variabel secara umum hanya berisi sebuah referensi terhadap sebuah data yang disimpan pada memori. Sehingga ketika sebuah data kehilangan referensinya, ia tidak bisa lagi diakses. Oleh karena itu garbage collector pada python akan menghapusnya dari memori.
- Perhatikan contoh berikut:

```
x = 10
y = 5
x = y # <-- x merujuk ke 5 bukan lagi ke 10
```

Kira-kira berapa nilai x?

Nilai x pada kode di atas akan bernilai 5, karena ia sekarang merujuk ke nilai 5. Sedangkan nilai 10 sudah tidak bisa kita akses lagi karena ia kehilangan referensinya.

Nah, ketika proses inilah destruktor pada python akan dipanggil.

Perhatikan contoh kelas Mahasiswa berikut:

```python
class Mahasiswa:
  def __init__(self, nama):
    self.nama = nama
    print(f'mahasiswa {self.nama} dibuat')

  def __del__(self):
    print(f'mahasiswa {self.nama} dihapus')

mahasiswaX = Mahasiswa('Budi')
mahasiswaX = 10 # Budi kehilangan referensi

print(f'mahasiswaX = {mahasiswaX}')
```

Output:

```
mahasiswa Budi dibuat
mahasiswa Budi dihapus
mahasiswaX = 10
```

- Dari output di atas terbukti bahwa ketika suatu objek kehilangan referensinya, ia akan masuk ke dalam proses “penghancuran”. Dan sebelum proses tersebut dijalankan, sistem masih akan memanggil desktruktor terlebih dahulu demi memberikan kesempatan terakhir sebelum benar-benar hilang.

# Kesimpulan

- Dari pertemuan kali ini kita bisa simpulkan beberapa poin di antaranya:
- Variabel sebenarnya hanya berisi referensi terhadap letak suatu objek pada memori
- Ketika sebuah variabel dihapus (baik secara manual atau pun secara natural), proses destruktor akan dieksekusi
- Destruktor adalah kesempatan terakhir untuk “bersih-bersih” sebelum objek benar-benar dihapus dari memori
- Kita bisa mendefinisikan aksi destruktor pada python dengan mengimplementasikan magic method bernama __del__()


- Referensi
- [1] https://www.geeksforgeeks.org/destructors-in-python/ – diakses tanggal 3 Juni 2021
- [2] https://pythonprogramminglanguage.com/destructor/ – diakses tanggal 3 Juni 2021
- [3] https://www.studytonight.com/python/destructors-in-python – diakses tanggal 3 Juni 2021
- Python: Destructor 🐍 | Jago Ngoding