

# Pertemuan 10

## Penganganan Eksepsi

Pada saat menulis dan menjalankan program, kita sering dihadapkan pada munculnya kesalahan atau error. Seringkali error menyebabkan program berhenti sendiri. Error dapat terjadi akibat kesalahan struktur (sintaks) program. Hal ini disebut syntax error. Contohnya adalah seperti berikut:

```
>>> if x < 5   File "<stdin>", line 1
if x < 5
SyntaxError: invalid syntax
```

Kita bisa melihat bahwa penyebabnya adalah lupa titik dua pada pernyataan if. Error juga dapat terjadi pada saat runtime (saat program berjalan). Error seperti ini disebut eksepsi. Misalnya, bila kita membuka file yang tidak ada, maka akan muncul pesan kesalahan FileNotFoundError. Bila kita membagi bilangan dengan nol akan muncul ZeroDivisionError, dan lain sebagainya.

Pada saat terjadi eksepsi, Python akan menampilkan traceback dan detail dimana kesalahan terjadi.

```
>>> 1/0
```

```
Traceback (most recent call last):
```

```
File "<stdin>", line 1, in
```

```
<module>
```

```
ZeroDivisionError: division by zero
```

Eksepsi	Penyebab Error
AssertionError	Muncul pada saat pernyataan <code>assert</code> gagal
AttributeError	Muncul pada saat penugasan terhadap attribute atau referensi gagal
EOFError	Muncul saat fungsi <code>input()</code> mendapatkan kondisi akhir file (end-of-file)
FloatingPointError	Muncul saat operasi terhadap bilangan float gagal
GeneratorExit	Muncul saat metode <code>close()</code> generator dipanggil
ImportError	Muncul saat modul yang hendak diimport tidak ditemukan
IndexError	Muncul saat indeks dari sequence berada di luar range
KeyError	Muncul saat suatu key tidak ditemukan di dalam dictionary
KeyboardInterrupt	Muncul saat user menekan tombol interupsi (Ctrl + C)
MemoryError	Muncul saat operasi kehabisan memori
NameError	Muncul saat variabel tidak ditemukan

NotImplementedError	Muncul oleh metode abstrak
OSError	Muncul saat sistem operasi bersangkutan mengalami error
OverflowError	Muncul saat hasil operasi perhitungan terlalu besar untuk direpresentasikan
ReferenceError	Muncul saat <i>weak reference</i> digunakan untuk mengakses referensi sampah program
RuntimeError	Muncul saat error yang terjadi di luar semua kategori eksepsi lain
StopIteration	Muncul oleh fungsi <code>next()</code> untuk menunjukkan bahwa tidak ada lagi item yang tersisa pada iterator
SyntaxError	Muncul oleh parser saat terjadi kesalahan sintaks
IndentationError	Muncul saat ada indentasi yang salah
TabError	Muncul saat indentasi memiliki jumlah spasi atau tab yang tidak konsisten
SystemError	Muncul saat interpreter mendeteksi kesalahan internal
SystemExit	Muncul oleh fungsi <code>sys.exit()</code>

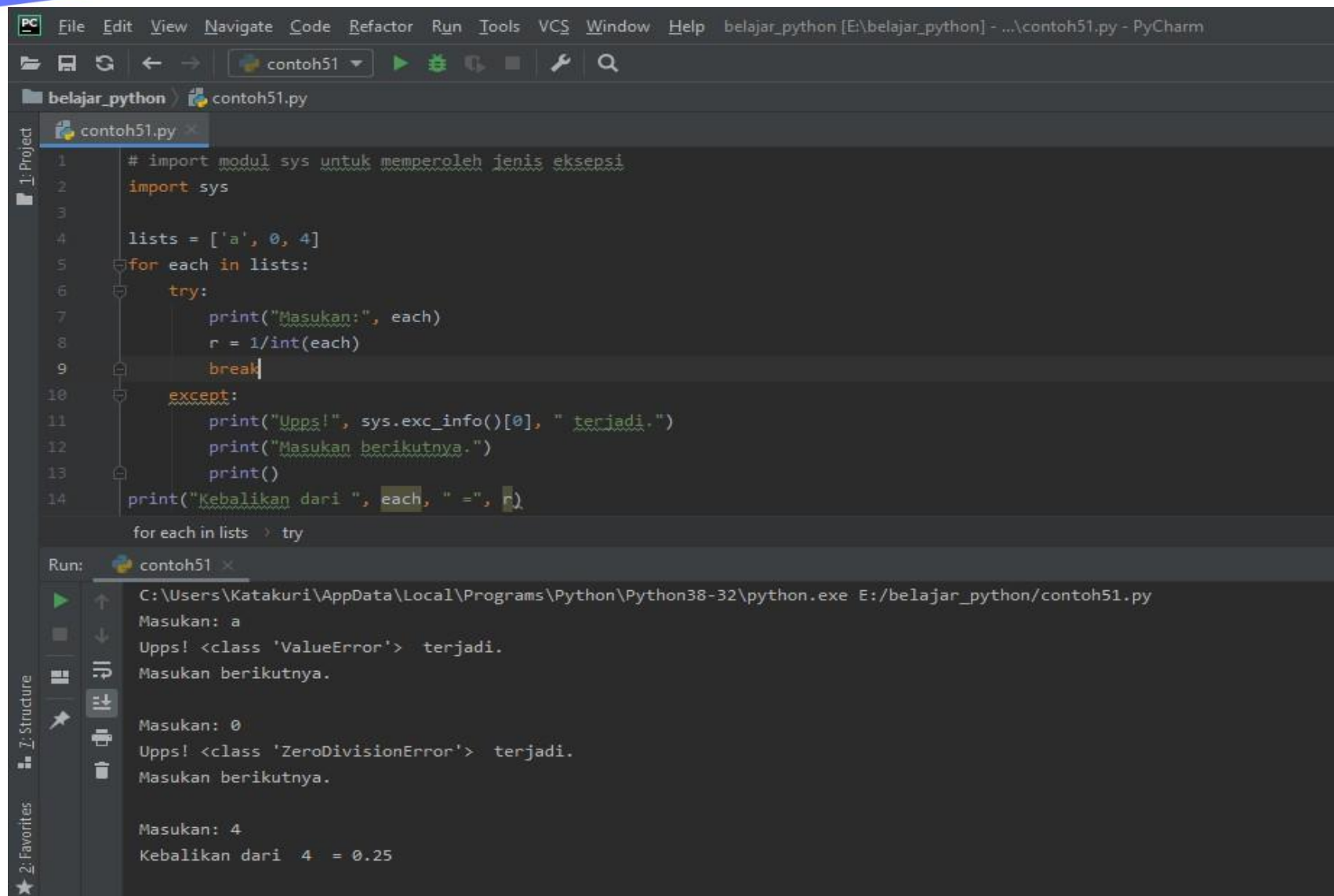
TypeError	Muncul saat melakukan operasi pada tipe data yang tidak sesuai
UnboundLocalError	Muncul saat referensi dibuat untuk variabel lokal dari fungsi, tapi tidak ada nilainya.
UnicodeError	Muncul saat terjadi kesalahan berkenaan dengan encoding dan decoding unicode
UnicodeEncodeError	Muncul saat terjadi kesalahan pada proses encoding
UnicodeDecodeError	Muncul saat terjadi kesalahan pada proses decoding
UnicodeTranslateError	Muncul saat terjadi kesalahan berkenaan dengan penerjemahan unicode
ValueError	Muncul saat fungsi menerima argumen yang tipe datanya salah
ZeroDivisionError	Muncul saat terjadi operasi pembagian bilangan dengan nol

# Menangani Eksepsi Dengan Try, Except, dan Finally

Terjadinya eksepsi pada program dapat menyebabkan program terhenti. Untuk mencegah hal tersebut, kita harus mengantisipasi hal tersebut. Python menyediakan metode penanganan eksepsi dengan menggunakan pernyataan try dan except.

Di dalam blok try kita meletakkan baris program yang kemungkinan akan terjadi error. Bila terjadi error, maka penanganannya diserahkan kepada blok except. Berikut adalah contoh penanganan eksepsi pada operasi pembagian bilangan.





```
1 # import modul sys untuk memperoleh jenis eksepsi
2 import sys
3
4 lists = ['a', 0, 4]
5 for each in lists:
6     try:
7         print("Masukan:", each)
8         r = 1/int(each)
9         break
10    except:
11        print("Upps!", sys.exc_info()[0], " terjadi.")
12        print("Masukan berikutnya.")
13        print()
14    print("Kebalikan dari ", each, " =", r)
```

Run: contoh51 x

C:\Users\Katakuri\AppData\Local\Programs\Python\Python38-32\python.exe E:/belajar\_python/contoh51.py

Masukan: a

Upps! <class 'ValueError'> terjadi.

Masukan berikutnya.

Masukan: 0

Upps! <class 'ZeroDivisionError'> terjadi.

Masukan berikutnya.

Masukan: 4

Kebalikan dari 4 = 0.25

Pada program di atas kita mencari kebalikan dari bilangan, misalnya 4, maka kebalikannya adalah  $1/4 = 0.25$ .



Pembagian dengan huruf 'a', dan juga dengan 0 tidak bisa dilakukan, sehingga muncul error. Bila tidak dilakukan penanganan eksepsi, maka program akan langsung terhenti pada saat terjadi error.

## Menangani Eksepsi Tertentu

Pada contoh di atas kita hanya menangani error secara umum. Tidak dikelompokkan, apakah dia adalah `TypeError`, `ValueError`, `SyntaxError`, dan lain sebagainya. Sebuah pernyataan `try`, bisa memiliki sejumlah pernyataan `except` untuk menangani jenis – jenis eksepsi secara lebih spesifik. Kita juga bisa mendefinisikan beberapa error sekaligus menggunakan tuple. Contohnya adalah seperti berikut:

```
try:
```

```
    # lakukan sesuatu    pass
```

```
except ValueError:
```

```
    # tangani eksepsi ValueError    pass
```

```
except (TypeError, ZeroDivisionError):    #
```

```
    menangani multi eksepsi
```

```
    # TypeError dan ZeroDivisionError    pass
```

```
except:
```

```
    # menangani eksepsi lainnya  
    pass
```

Pernyataan `pass` adalah pernyataan yang tidak melakukan apa-apa. Istilahnya adalah statemen kosong. `pass` sering digunakan untuk mengisi blok fungsi atau kelas yang masih kosong.

## Memunculkan Eksepsi

Eksepsi muncul bila terjadi error pada saat runtime atau saat program berjalan. Akan tetapi, kita juga bisa memunculkan eksepsi dengan sengaja untuk maksud tertentu dengan menggunakan kata kunci `raise`. Contohnya adalah seperti berikut:

```
>>> raise KeyboardInterrupt
```

```
Treaceback (most recent call last):
```

```
...
```

```
KeyboardInterrupt
```

```
>>> try:
```

```
    a = int(input("Masukkan sebuah bilangan positif: "))
```

```
    if a <= 0:
```

```
        raise ValueError("Itu bukan bilangan positif!")
```

```
except ValueError as ve:  print(ve)
```

```
Masukkan sebuah bilangan positif: -3  Itu
```

```
bukan bilangan positif!
```