Exceptions

BIN-OWE2

Studiewijzer

Les	Onderwerp	
	Algemeen	**Python**
1	Debuggen en testen	Exceptions
2		Matplotlib
3		Datastructuren
4		Regular Expressions
5		Object Oriëntatie
6		Recursie
7		GUI programmeren

Doelstellingen

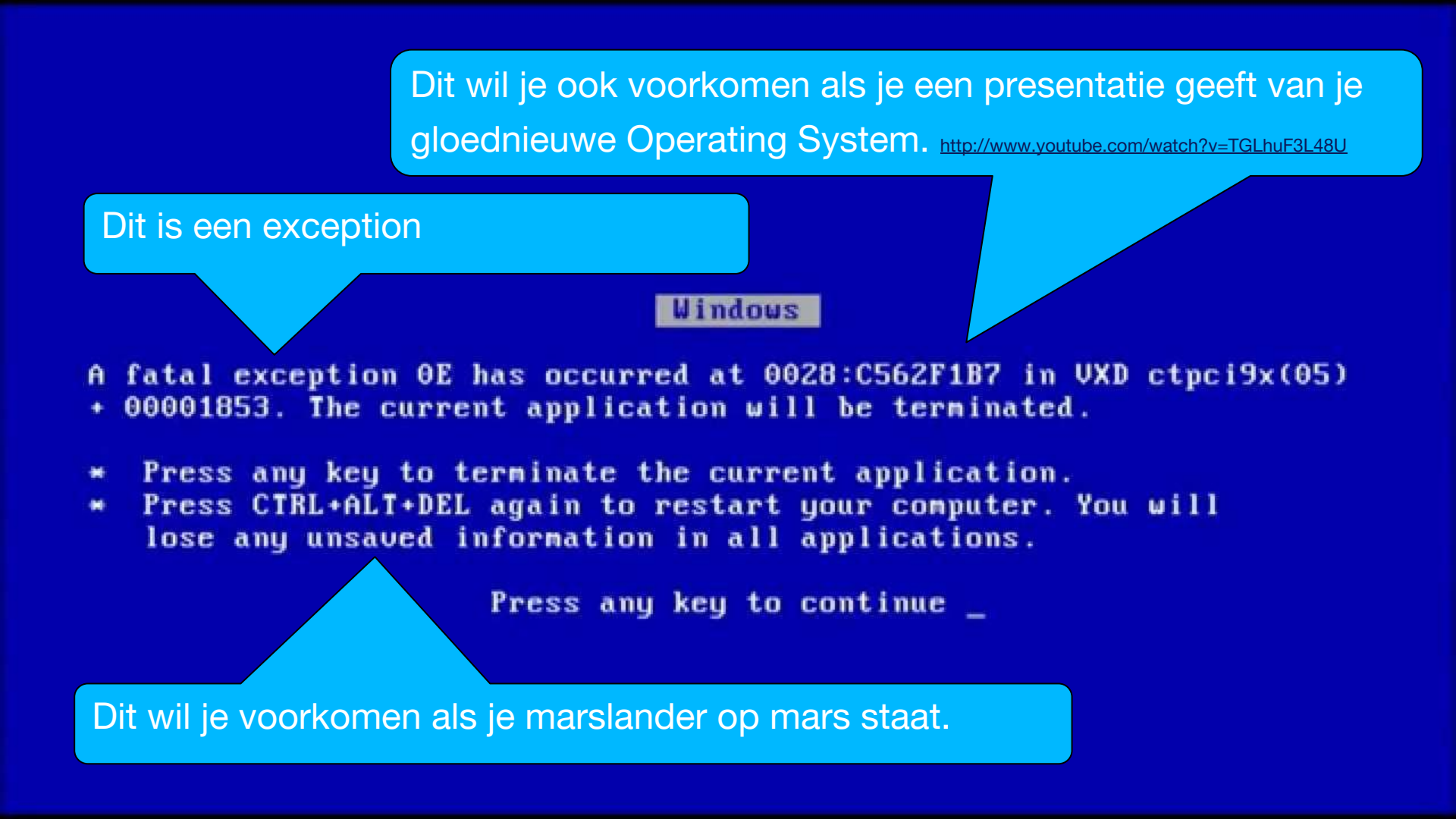
- Aan het eind van deze les kan je
  - Een try/except structuur toevoegen aan je code
  - Errors opgooien op het moment dat er iets mis gaat

- Aan het eind van deze week begrijp je
  - Hoe built in errors werken

Inhoud

- Introductie Exceptions
- Built in errors
- Try/except structuur
- Zelf errors opgooien
- Custom errors maken


Windows
A fatal exception 0E has occurred at 0028:C562F1B7 in VXD ctpci9x(05)
+ 00001853. The current application will be terminated.
* Press any key to terminate the current application.
* Press CTRL+ALT+DEL again to restart your computer. You will
lose any unsaved information in all applications.
Press any key to continue _


Dit wil je ook voorkomen als je een presentatie geeft van je gloednieuwe Operating System. http://www.youtube.com/watch?v=TGLhuF3L48U
Dit is een exception
Windows
A fatal exception 0E has occurred at 0028:C562F1B7 in VXD ctpci9x(05)
+ 00001853. The current application will be terminated.
* Press any key to terminate the current application.
* Press CTRL+ALT+DEL again to restart your computer. You will
lose any unsaved information in all applications.
Press any key to continue _
Dit wil je voorkomen als je marslander op mars staat.

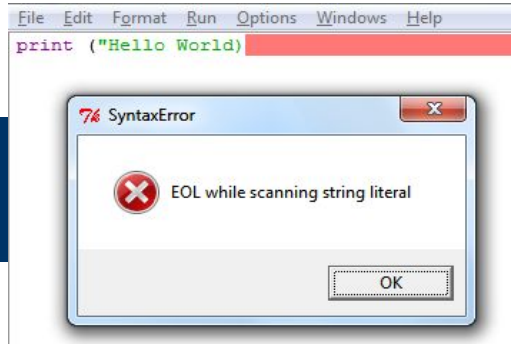

CNN
Microsoft

Compile errors vs. Exceptions

Compile error

- Treedt op tijdens compileren
- Programma kan niet runnen
- De compile error wordt verholpen door de programmeur

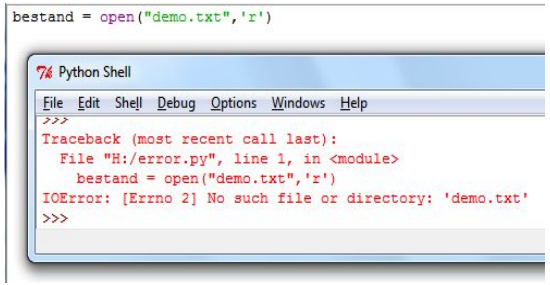
- Is altijd ongewenst



Exception

- Treedt op tijdens uitvoer
- Programma kan runnen totdat de exception op zal treden
- De exception zal niet altijd optreden

- Is soms gewenst



HAN

Exception Handling

- Bij de uitvoer van een programma kunnen er fouten optreden
- Deze fouten die je verwacht kun je afvangen in een exception handling

```
# AUTHOR:       Martijn van der Bruggen <Martijn.vanderBruggen@han.nl>
# DESCRIPTION:  Twee manieren om door bestanden heen te loopen


# While loop
bestand = open ("
                Open file and return a stream.  Raise IOError upon failure.
```

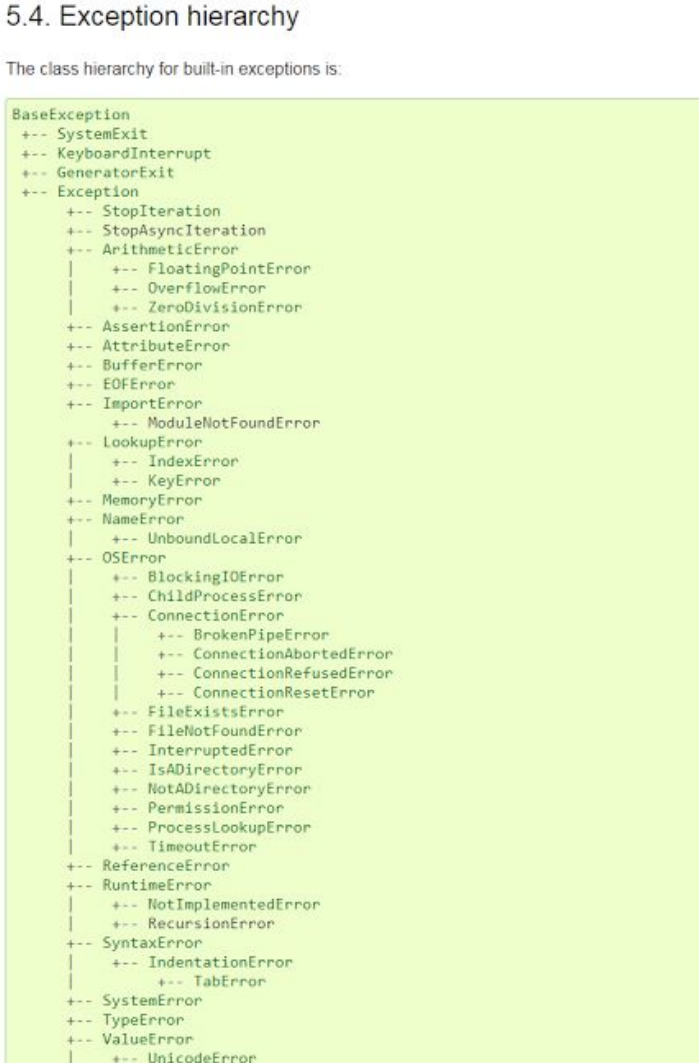
Inhoud

- Introductie Exceptions
- Built in errors
- Try/except structuur
- Zelf errors opgooien
- Custom errors maken

Documentatie exceptions

- https://docs.python.org/3/library/exceptions.html

(14-11-2017)



5.4. Exception hierarchy

The class hierarchy for built-in exceptions is:

```
BaseException
 +-- SystemExit
 +-- KeyboardInterrupt
 +-- GeneratorExit
 +-- Exception
      +-- StopIteration
      +-- StopAsyncIteration
      +-- ArithmeticError
      |    +-- FloatingPointError
      |    +-- OverflowError
      |    +-- ZeroDivisionError
      +-- AssertionError
      +-- AttributeError
      +-- BufferError
      +-- EOFError
      +-- ImportError
           +-- ModuleNotFoundError
      +-- LookupError
      |    +-- IndexError
      |    +-- KeyError
      +-- MemoryError
      +-- NameError
      |    +-- UnboundLocalError
      +-- OSError
      |    +-- BlockingIOError
      |    +-- ChildProcessError
      |    +-- ConnectionError
      |    |    +-- BrokenPipeError
      |    |    +-- ConnectionAbortedError
      |    |    +-- ConnectionRefusedError
      |    |    +-- ConnectionResetError
      |    +-- FileExistsError
      |    +-- FileNotFoundError
      |    +-- InterruptedError
      |    +-- IsADirectoryError
      |    +-- NotADirectoryError
      |    +-- PermissionError
      |    +-- ProcessLookupError
      |    +-- TimeoutError
      +-- ReferenceError
      +-- RuntimeError
      |    +-- NotImplementedError
      |    +-- RecursionError
      +-- SyntaxError
      |    +-- IndentationError
      |         +-- TabError
      +-- SystemError
      +-- TypeError
      +-- ValueError
      |    +-- UnicodeError
```

HAN

Deze wellicht al eens gezien?

- Deling door nul: ZeroDivisionError

exception **`ZeroDivisionError`**

Raised when the second argument of a division or modulo operation is zero. The associated value is a string indicating the type of the operands and the operation.

- Bestand openen dat niet bestaat: IOError

exception **`IOError`**

Raised when an I/O operation (such as a **`print`** statement, the built-in **`open()`** function or a method of a file object) fails for an I/O-related reason, e.g., "file not found" or "disk full".

Exceptions: bestand lezen

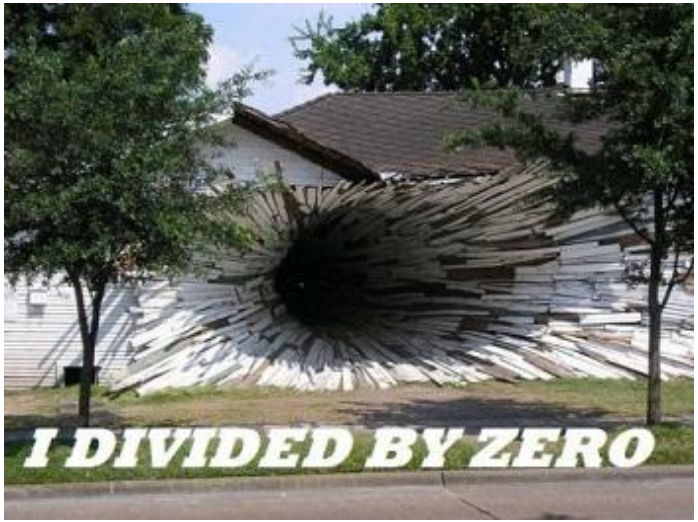
- Een bekend voorbeeld is het openen van een bestand dat niet bestaat

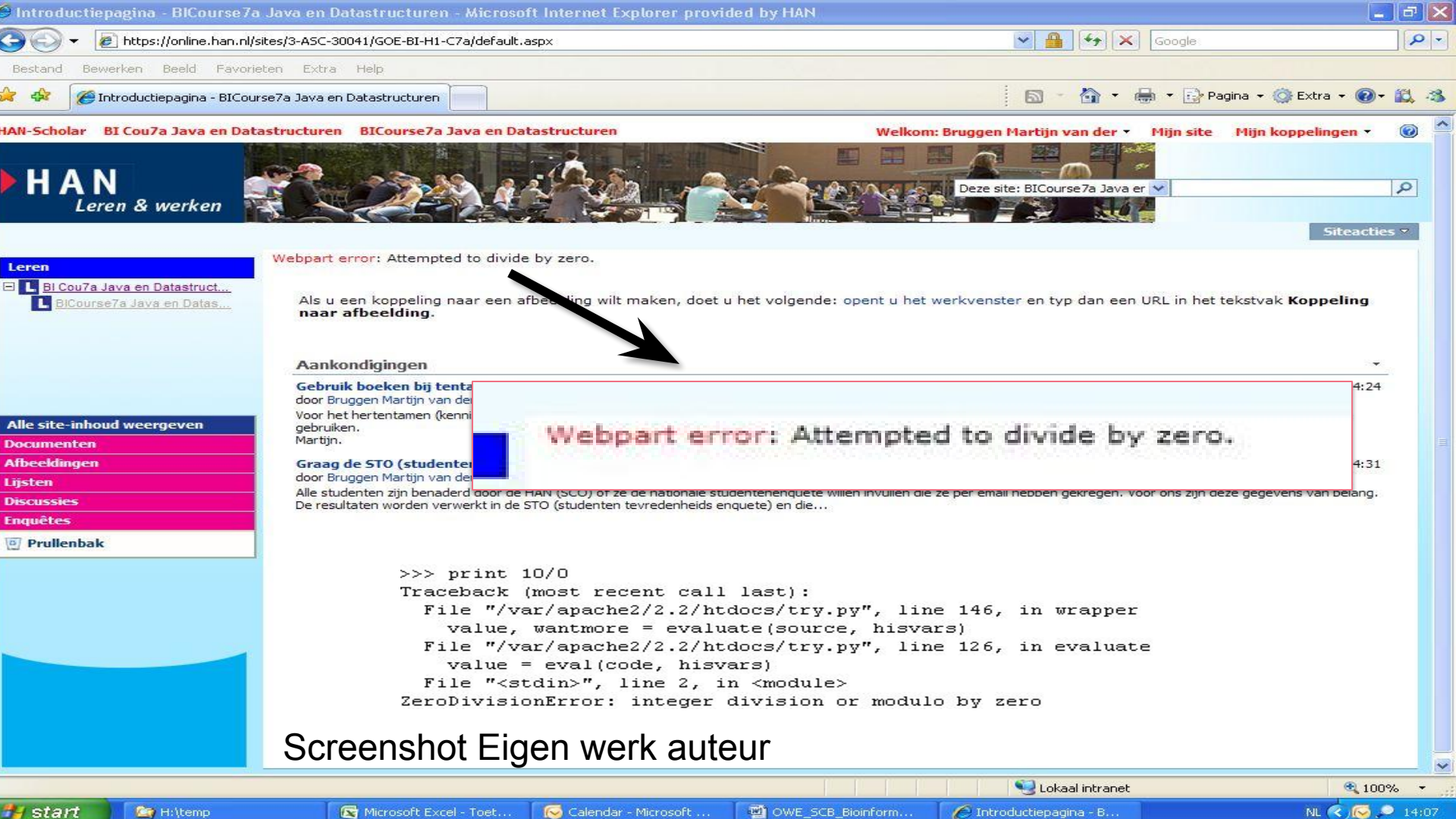
```
>>> bestand = open("demo.fa")
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    bestand = open("demo.fa")
IOError: [Errno 2] No such file or directory: 'demo.fa'
```

Divide by zero

- Delingen door nul mogen niet

```
>>> a = 2
>>> b = 0
>>> print (a/b)
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    print (a/b)
ZeroDivisionError: division by zero
>>>
```





Screenshot Eigen werk auteur

Inhoud

- Introductie Exceptions
- Built in errors
- Try/except structuur
- Zelf errors opgooien
- Custom errors maken

Exception Handling

- Het reageren op fouten vind plaats in de try en except

Probleem

```
def main():
        jaren = invoer()
        print("U bent", str(jaren),"jaren oud")
        print("Dat is",str(jaren*12),"maanden")


def invoer():
    getal = int(input("Voer uw leeftijd in (jaren): "))
    return getal

main()
```

```
Voer uw leeftijd in (jaren): 9
U bent 9 jaren oud
Dat is 108 maanden
```

```
Voer uw leeftijd in (jaren): Piet
Traceback (most recent call last):
  File "/home/esther/Desktop/les1.py", line 14, in <module>
    main()
  File "/home/esther/Desktop/les1.py", line 3, in main
    jaren = invoer()
  File "/home/esther/Desktop/les1.py", line 11, in invoer
    getal = int(input("Voer uw leeftijd in (jaren): "))
ValueError: invalid literal for int() with base 10: 'Piet'
```

HAN

Oplossing

```
def main():
    try:
        jaren = invoer()
        print("U bent", str(jaren),"jaren oud")
        print("Dat is",str(jaren*12),"maanden")
    except ValueError:
        print("Deze invoer is geen getal. Probeer het opnieuw")

def invoer():
    getal = int(input("Voer uw leeftijd in (jaren): "))
    return getal

main()
```

Plek ontstaan error

Aanroep functie in try

Wordt hier afgevangen

Ontstaat hier

```
def main():
    try:
        jaren = invoer()
        print("U bent", str(jaren),"jaren oud")
        print("Dat is",str(jaren*12),"maanden")
    except ValueError:
        print("Deze invoer is geen getal. Probeer het opnieuw")


def invoer():
    getal = int(input("Voer uw leeftijd in (jaren): "))
    return getal


main()
```

Meerdere excepts mogelijk

```
try:
   bestand = open("getallen.txt","r")
   a = bestand.readline()
   b = bestand.readline()
   c = float(a)/float(b)
   print c
except IOError:
    print "Kan het bestand niet vinden"
except ZeroDivisionError:
    print "Deling door nul is niet toegestaan"
except ValueError:
    print "Conversie van een getal mislukt"
```

Default Exception Handling

```
try:
   bestand = open("getallen.txt","r")
   a = bestand.readline()
   b = bestand.readline()
   c = float(a)/float(b)
   print c
except IOError:
    print "Kan het bestand niet vinden"
except ZeroDivisionError:
    print "Deling door nul is niet toegestaan"
except ValueError:
    print "Conversie van een getal mislukt"
except:
    print """Onbekende fout 13443,
             raadpleeg uw systeembeheerder!"
```

Syntax

```
try:
   <statement>
except <built-in exception>:
   <statement>
except <built-in exception>:     #optioneel
   <statement>
except:                          #optioneel
   <statement>
```

Opdracht (10 min)

- Samen met je buurman/buurvrouw
- Raadpleeg de Python documentatie op built in errors
- Welke error ontstaat er op het moment dat ik een programma handmatig onderbreek? (Ctrl + C tijdens runtime)
- Schrijf een kort programma met een oneindige while loop dat deze error netjes afvangt

Mogelijke oplossing

```
try:
    while True:
        print("Hello")

except KeyboardInterrupt:
    print("Interrupted by user")
```

Inhoud

- Introductie Exceptions
- Built in errors
- Try/except structuur
- Zelf errors opgooien
- Custom errors maken

Zelf opgooien

- Kan met `raise`

- Sytax:

```
raise <eenofandereerror>
```

- Vaak in combinatie met een if else structuur

```
if <conditie van toepassing>:
    raise <error>
```

Opdracht (5-10 min)

- Samen met je buurman/buurvrouw
- Raadpleeg de Python documentatie op built in errors
- Wat zou een goede error zijn om te gebruiken op het moment dat de ingevoerde sequentie niet uit aminozuren bestaat?

Voorbeeld

```
def main():
    try:
        seq = invoer()
        print("Dit is uw sequentie:", seq)
    except TypeError:
        print("Deze invoer bestaat niet uit
              aminozuren. Probeer het opnieuw")


def invoer():
    sequentie = input("Voer uw eiwit
                sequentie in: ")
    if check(sequentie) == False:
        raise TypeError
    return sequentie
```

```
def check(seq):
    seq = seq.upper()
    for aa in seq:
    if aa in
          ["B","J","O","U","X","Z"]:
            return False
    return True

main()
```

Opdracht (20 min)

- Samen met je buurman/buurvrouw
- Raadpleeg de Python documentatie op built in errors
- Welke error ontstaat er op het moment dat ik een index uit een lijst opvraag die niet bestaat?

Opdracht (20 min)

- Schrijf een kort programma dat deze error afvangt
- Vraag om 5 items gescheiden op komma als invoer
- Op het moment dat de invoer te weinig items bevat gooi je de error op

Mogelijke oplossing

```
def main():
    try:
        lijst = invoer()
        print(lijst)
    except IndexError:
        print("De ingevoerde rij klopt niet met het aantal items in de lijst.")


def invoer():
    invoer = input("Geef 5 items gescheiden door komma.")
    lijst = invoer.split(",")
    if len(lijst)!=5:
        raise IndexError
    return lijst


main()
```

Inhoud

- Introductie Exceptions
- Built in errors
- Try/except structuur
- Zelf errors opgooien
- Custom errors maken

Zelf errors maken

- Over het algemeen zullen de built-in errors volstaan...
- ...maar zelf maken is wel mogelijk
- Object oriëntatie nodig
- Pas in week 5

Samenvatting

- Python kent een groot aantal ingebouwde exceptions
- Deze kan je afvangen met een try/except structuur

- Compile errors ontstaan door fouten in de code/syntax
- Deze moet je oplossen om het programma te laten werken (debuggen)

- Exceptions ontstaan door onverwachte problemen tijdens het draaien van het programma
- Deze kan je oplossen door plannen en testen

Opdracht

Afvinkopdracht 1

Verantwoording

- In deze uitgave is géén auteursrechtelijk beschermd werk opgenomen
- Alle teksten © Martijn van der Bruggen/Esther Kok/HAN tenzij expliciet externe bronnen zijn aangegeven
- Screenshots op basis van eigen werk auteur en/of vernoemde sites en/of fair use
- Eventuele images zijn opgenomen met vermelding van bron