Python Errors

Petr Svarny, 2020

Error

- Two kinds of errors in Python
  - Syntax error
    - Bad indentation, parenthesis is not closed, forgotten colon
  - Exceptions

```
>>> if x > 2
...     print('x is greater than 2')
File "<ipython-input-73-00cd78beb41b>", line 1
if x > 2
^
SyntaxError: invalid syntax
```

Indentation error and missing parenthesis error

```
>>> if x > 2:
... print('x is greater than 2')
File "<ipython-input-74-544106e52c6f>", line 2
print('x is more than 2')
^
IndentationError: expected an indented block
>>> print'Hello'
File "<ipython-input-75-352cd0cc9337>", line 1
print 'Hello'
^
SyntaxError: Missing parentheses in call to 'print'
```

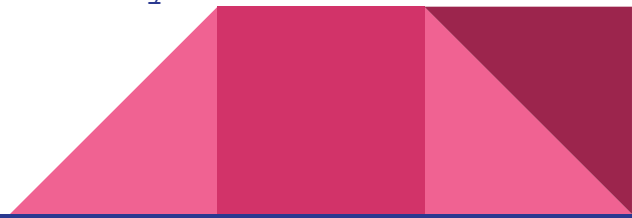
Exception

- Are syntactically correct
- List of built-in exceptions(Python 3.7)

```
>>> 3/0
Traceback (most recent call last):

File "<ipython-input-80-a0641230c7a8>", line 1, in <module>
3/0
ZeroDivisionError: division by zero
```

Exception

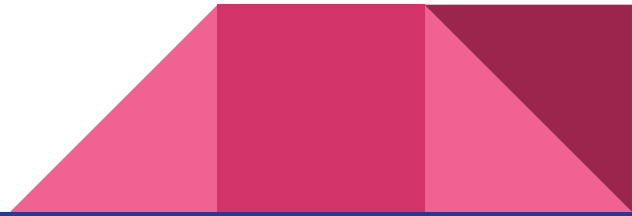
```
>>> Fruits
Traceback (most recent call last):
File "<ipython-input-82-2f4f0fa8bdfe>", line 1, in <module>
Fruits
NameError: name 'Fruits' is not defined

>>> f = open('frut.txt', 'r')
Traceback (most recent call last):
File "<ipython-input-83-bdcc5d8c7fd1>", line 1, in <module>
f = open('frut.txt', 'r')
FileNotFoundError: [Errno 2] No such file or directory: 'frut.txt'
```



Exercise

- Correct following code - try first without running code

```
seasons = ['Spring', 'Summer', 'Fall', 'Winter']
print('My favorite season is ', seasons[4])
```



Exercise

- Correct following code - try first without running code

```
for number in range(10):
    # use a if the number is a multiple of 3, otherwise use b
    if (Number % 3) == 0:
        message = message + a
    else:
        message = message + "b"
print(message)
```

Catching exceptions

- Use ***try - except***
- Generally not recommended, use conditions instead

```
try:
    code block where exception can occur
except:
    code block that will run if exception occurs
```

Catching exceptions

```
>>> try:
...     print(3 + 3)
...     print('x' + 3)
...     print(2 + 2)
... except:
...     print('Error occurred')

6
Error occurred
```

Raising Exception

- We can raise exception in our code using ***raise***

```
>>> raise Exception('Oh no, this is supposed to be number!')
Traceback (most recent call last):
File "<ipython-input-87-4471a9c5e463>", line 1, in <module>
raise Exception('Oh no, this is supposed to be number!')
Exception: Oh no, this is supposed to be number!
```

Catching exceptions

- Better to use except with specific exception than catch all exceptions

```
>>> try:
...     raise NameError
... except IOError:
...     print('Input/Output error')
... except NameError:
...     print('Object does not exist')
... except:
...     print('Some unknown exception')
... else:
...     print('Everything is OK!')
```

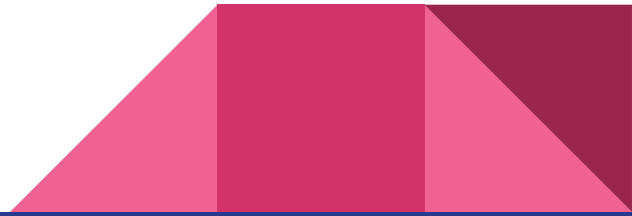
Exercise

- Ask user to type name
- Raise exception if name
  - Contains number
  - Has spaces
  - Does not start with uppercase letter

Syntax hint: `raise Exception('string')`

Exercise

- Create function that ask user to type two integers and return division result
- If user type other data type than integer, ask to type integer
- If second number is 0, ask user to type number again until number is not 0
- Hint: while and/or try - except



Code debugging

- Locating, analyzing, and correcting a bug (error)
- May be as twice time consuming as writing the code
- Creative and intellectually challenging part of programming

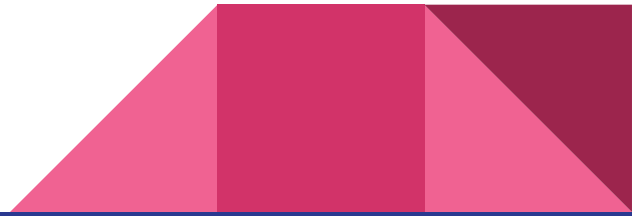
Debugging tips

- **Don’t panic**
- Get into debug mode
- Don’t look for complex explanations
- If you code was working a minute ago, but now it doesn’t—what was the last thing you changed?
- Reproduce problem
- **Understand the error message. Do not be afraid of errors, they are here to help** 🙂
- Be critical of your **beliefs** about your code
- Be systematic and persistent. **Don’t panic**

Exercise

- Debug following code

```
year == int.input("Greetings! What is your year of origin? "))

if year <= 1900
    print ('Woah, that's the past!')
elif year > 1900 && year < 2020:
    print ("That's totally the present!")
elif:
    print ("Far out, that's the future!!")
```



Debug tool even without IDE

Package ipdb

- import ipdb; ipdb.set_trace()