Python programming for beginners

Stefan Zhauryd
Instructor

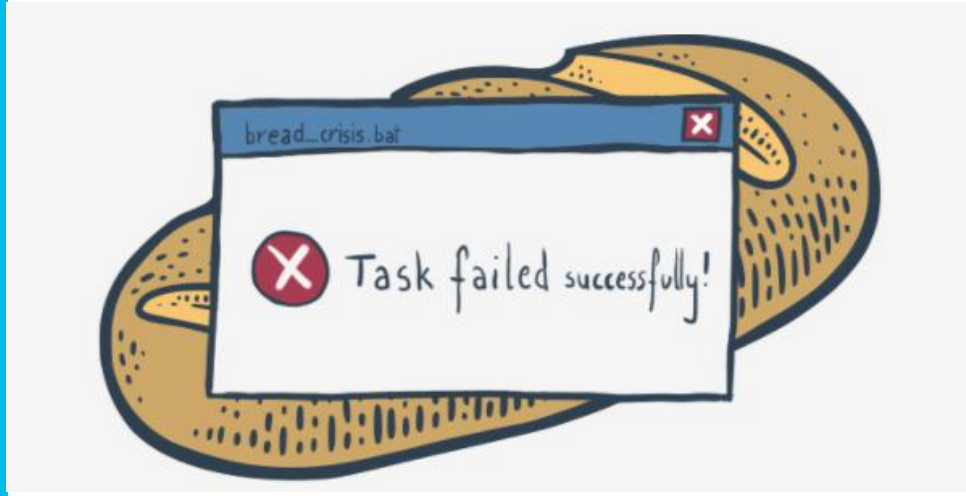
Module 5

Exceptions

In this module, you will learn about:

- **exceptions – the try statement and the except clause, Python built-in exceptions, code testing and debugging.**

Mistakes: Developer's Daily Bread



Errors in the data versus errors in the code

```
1  value = int(input('Enter a natural number: '))
2  print('The reciprocal of', value, 'is', 1/value)
3
```

```
Enter a natural number: 0
Traceback (most recent call last):
  File "main.py", line 2, in <module>
    print('The reciprocal of', value, 'is', 1/value)
ZeroDivisionError: division by zero
```

When data is not what it should be

```
type(value) is int
```

```
Enter a natural number: вав
Traceback (most recent call last):
  File "main.py", line 1, in <module>
    value = int(input('Enter a natural number: '))
ValueError: invalid literal for int() with base 10: 'вав'
```

```
>>> TYPES_FOR_OPERATION = [int, float]
>>> a = 10
>>> type(a) in TYPES_FOR_OPERATION
True
>>> a = ''
>>> type(a) in TYPES_FOR_OPERATION
False
```

The **try-except** branch

```
try:
    # It's a place where
    # you can do something
    # without asking for permission.
except:
    # It's a spot dedicated to
    # solemnly begging for forgiveness.
```


You can see two branches here:

- first, starting with the **try** keyword – this is the place where you put the code you suspect is risky and may be terminated in case of error; note: this kind of error is called an exception, while the exception occurrence is called **raising** – we can say that an exception is (or was) **raised**;
- second, the part of the code starting with the **except** keyword is designed to handle the exception; it's up to you what you want to do here: you can clean up the mess or you can just sweep the problem under the carpet (although we would prefer the first solution).

```
1 try:
2     value = int(input('Enter a natural number: '))
3     print('The reciprocal of', value, 'is', 1/value)
4 except:
5     print('I do not know what to do.')
```

The exception proves the rule

```
Enter a natural number: 2
The reciprocal of 2 is 0.5
Enter a natural number: 0
I do not know what to do.
Enter a natural number:

I do not know what to do.
Enter a natural number: aaa
I do not know what to do.
```

```
try:
    value = int(input('Enter a natural number: '))
    print('The reciprocal of', value, 'is', 1/value)
except ValueError:
    print('I do not know what to do.')
except ZeroDivisionError:
    print('Division by zero is not allowed in our Universe.')
```

How to deal with more than one exception

```
Enter a natural number: 0
Division by zero is not allowed in our Universe.
Enter a natural number: nnn
I do not know what to do.
Enter a natural number: 2
The reciprocal of 2 is 0.5
```

Note: that both branches have exception names specified. In this variant, each of the expected exceptions has its own way of handling the error, but it must be emphasized that only one of all branches can intercept the control – if one of the branches is executed, all the other branches remain idle.

Additionally, the number of except branches is not limited – you can specify as many or as few of them as you need, but don't forget that none of the exceptions can be specified more than once.

The default exception and how to use it

```
try:
    value = int(input('Enter a natural number: '))
    print('The reciprocal of', value, 'is', 1/value)
except ValueError:
    print('I do not know what to do.')
except ZeroDivisionError:
    print('Division by zero is not allowed in our Universe.')
except:
    print('Something strange has happened here... Sorry!')
```

The default **except** branch must be the last except branch. **Always!**

Some useful exceptions

ZeroDivisionError - /, // и%.

ValueError - int() or float()

TypeError -

short_list = [1]

one_value = short_list[0.5]

AttributeError -

short_list = [1]

short_list.append(2)

short_list.depend(3)

SyntaxError

Why you can’t avoid testing your code

```
1  temperature = float(input('Enter current temperature:'))
2
3  if temperature > 0:
4      print("Above zero")
5  elif temperature < 0:
6      print("Below zero")
7  else:
8      print("Zero")
```

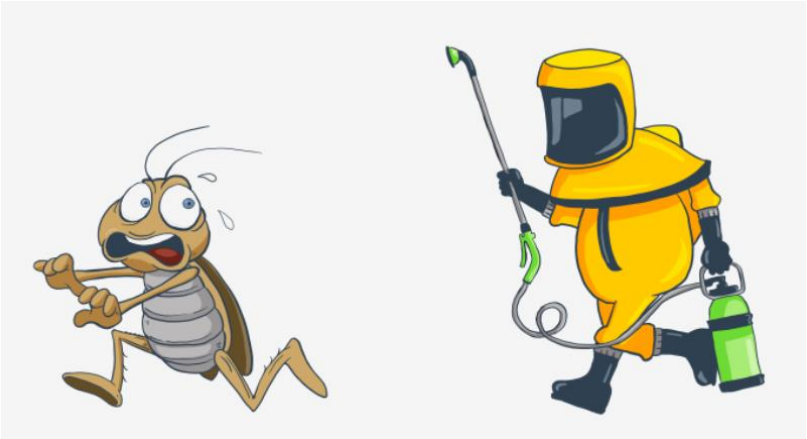
```
1  temperature = float(input('Enter current temperature:'))
2
3  if temperature > 0:
4      print("Above zero")
5  elif temperature < 0:
6      prin("Below zero")
7  else:
8      print("Zero")
```

When Python closes its eyes

```
Enter current temperature:1
Above zero
Enter current temperature:-1
Traceback (most recent call last):
  File "main.py", line 6, in <module>
    prin("Below zero")
NameError: name 'prin' is not defined
Enter current temperature:0
Zero
```

Tests, testing, and testers

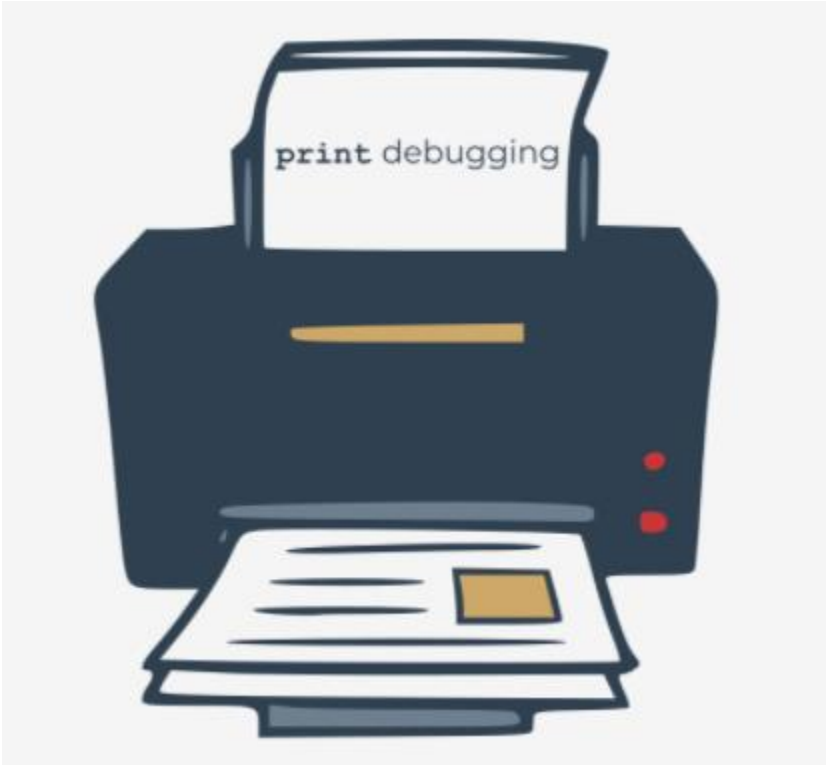
Bug vs. debug



https://www.cs.uky.edu/~keen/help/debug-tutorial/debug.html

https://docs.python.org/3/library/idle.html

print debugging



Some useful tips Unit testing – a higher level of coding

- Try to tell someone
- Try to isolate the problem
- Analyze all the changes you've introduced into your code
- Take a break ~40min ~5-10
- Be optimistic

https://en.wikipedia.org/wiki/Code_review

```
while True:
    try:
        number = int(input("Enter an integer number: "))
        print(number/2)
        break
    except:
        print("Warning: the value entered is not a valid number. Try again...")
```

Key takeaways – Exceptions

```
print(1/0)
```

```
Traceback (most recent call last):
  File "main.py", line 1, in
    print(1/0)
ZeroDivisionError: division by zero
```

```
print("Hello, World!)
```

```
  File "main.py", line 1

    print("Hello, World!)
                         ^
SyntaxError: EOL while scanning string literal
```

Key takeaways – Exceptions

KeyboardInterrupt

Ctrl+C /Z

```
while True:
    try:
        number = int(input("Enter an int number: "))
        print(5/number)
        break
    except (ValueError, ZeroDivisionError):
        print("Wrong value or No division by zero rule broken.")
    except:
        print("Sorry, something went wrong...")
```

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

```
while True:
    try:
        number = int(input("Enter an int number: "))
        print(5/number)
        break
    except ValueError:
        print("Wrong value.")
    except ZeroDivisionError:
        print("Sorry. I cannot divide by zero.")
    except:
        print("I don't know what to do...")
```

ЗАДАНИЯ

1) Прорешать всю классную работу
2) Выполнить все домашние задания

Почитать:

1) Byte of Python - повторяем

****) Structuring Your Project:**

Крайний срок сдачи 12/10 в 21:00 (можно раньше, но не позже)

https://docs.python-guide.org/writing/structure/

ЗАДАНИЯ

Название файлов, которые вы отправляете мне в telegram:
Vasia_Pupkin_class_work_Exception_L8_P0.py

Формат сообщения которое вы присылаете мне
(после полного выполнения домашнего задания, только один раз) в Telegram:
Добрый день/вечер. Я Вася Пупкин, и это мои домашние задания к лекции 8 часть 0 про основы исключений.
И отправляете файлы

Крайний срок сдачи 12/10 в 21:00 (можно раньше, но не позже)

https://docs.github.com/articles/using-pull-requests

Tap to links if you want to know more

Work with files:
https://www.youtube.com/watch?v=oRr_bEXJbV0
https://www.w3schools.com/python/python_ref_file.asp

Books for great peoples:
992 pages of "real" python

993 pages of "real" python

Watch this channel, useful things:
https://www.youtube.com/c/egoroffchannel/playlists

https://www.w3schools.com/python/default.asp

https://www.youtube.com/channel/UCr-KbmZWfDyTbqT_clZmhfw/videos

Create your possibilities. Bye bye.