# Python Brain Teasers

EXERCISE YOUR MIND

```
1 class Player:
2     # Number of players in the Game
3     count = 0
4
5     def __init__(self, name):
6         self.name = name
7         self.count += 1
8
9
10 p1 = Player('Parzival')
11 print(Player.count)
```

What will this code print?

30 mind bending teasers & solutions

Miki Tebeka

# Teaser 3. When in Kraków

*city.py*

```
1 city = 'Kraków'
2 print(len(city))
```

Try to guess what the output is before moving to the next page.

This code will print: 7

If you count the number of characters in Kraków, it'll come out to 6. So why 7? The reason is… history.

In the beginning, computers were developed in English speaking countries - the UK and the US. When early developers wanted to encode text in computers that only understand bits, they came out with the following scheme. Use a byte (8 bits) [5] to represent a character. For example a is 97 (01100001), b is 98, etc. One byte is enough for the English alphabet that contains 26 lower case letters, 26 upper case letters and, 10 digits. There is even some space left for other special characters (e.g. 9 for tab). This is known as ASCII encoding.

After a while, other countries started to use computers and they wanted to write using their native language. ASCII wasn't good enough, a single byte can't to hold all the numbers we need to represent letters in different languages. This lead to several different encoding schemes, the most common one is UTF-8.

Some of the characters in UTF-8 are control characters. In this case we have the character o at position 4 and after it a control character saying "add an umlaut to the previous character". This is why the length of the string is 7.

In Python 3 you have str which is an immutable sequence of Unicode code points and bytes which is an immutable sequence of bytes. At the edges of your program you when you get bytes, convert it to a str using the decode method. When you send data, use the str encode method to convert it to bytes. Internally, use strin your code.

## 3.1. Further Reading

- Unicode HOWTO
- Unicode and You
- Unicode entry on Wikipedia
- Pragmatic Unicode, or, How do I stop the pain? video. It's about Python 2 but still very good.

[1] To be precise, ASCII uses only 7 bits, and LATIN-1 extends it to 8 bits.