Kursus 02100 — Ugeseddel-Uge-06-Eftermiddag

Opgave 1

Skriv en Java-klasse `Bil` med to private felter `maerke` og `aargang` af type hhv. `String` og `int`.

a) Tilføj en konstruktør, der tager en streng og et heltal som parametre og initialiserer de to felter tilsvarende.

b) Tilføj en `toString`-metode, der returnerer `maerke`, efterfulgt af ordet ␣`fra`␣ (med mellemrum foran og bag) og så tallet fra `aargang`.

c) Opret to `Bil`-objekter af valgfrit mærke og årgang i klient-kode.

d) Udvid din konstruktør således, at feltet `aargang` sættes til maksimumet af 1860 og værdien af heltalsparameteren.

e) Tilføj to mutatormetoder, der gør det muligt at ændre hhv. en bils årgang (dog skal for lave værdierne stadig sættes til 1860) og dens mærke. Tilføj også tilsvarende accessor-metoder. Udvid din klient-kode til at bruge disse metoder.

f) Tilføj en konstruktør, der kun tager en streng som parameter, initialiserer `maerke` med strengen og sætter `aargang` til 2025. Brug denne konstruktør til at oprette endnu en bil i klient-koden.

Opgave 2

Usually a list in Python consists of similar kind of elements, but it is not a requirement. A tuple consists of a number of values separated by commas and parentheses can be added to disambiguate. For example, consider the following list of three tuples:

```
[(1, 2), (3, 4), (5, 6)]
```

Without the three pairs of parentheses, the list would consist of 6 integers and no tuples at all.

It is possible to change the elements of a list, but the elements of a tuple cannot be changed. Tuples make the code safe from any accidental changes and are more time and memory efficient than lists. Here is a simple example with a list and a tuple in Python:

```
list = [1, 2, 3]
list[1] = 4
tuple = 1, 2, 3
print(list, tuple)
```

The lines print:

```
[1, 4, 3] (1, 2, 3)
```

Consider the following lines:

```
anotherlist = [[1, 2], [3], []]
anotherlist[1] = 4, 2
anotherlist[2] = 999
print(anotherlist)
```

What do the lines print?

Opgave 3

In arithmetical expressions * performs multiplication, but when used with a list, tuple or string in Python, it performs repetition:

```
string = "ABC" * 3
print(2 * string)
list = [1, 2, 3]
print(list * 3)
```

The lines print:

```
ABCABCABCABCABCABC
[1, 2, 3, 1, 2, 3, 1, 2, 3]
```

Consider the following lines:

```
print(2 * 0, 1 * 3)
print(2 * (0, 1) * 3)
```

What do the lines print?

Opgave 4

Here is a simple example of a so-called list comprehension in Python:

```
[2 ** i for i in range(8)]
```

Result:

```
[1, 2, 4, 8, 16, 32, 64, 128]
```

Here is another example:

```
[len(x) for x in ["elements", "not", "relevant"]]
```

Result:

```
[8, 3, 8]
```

The variable `x` is not visible outside of the list comprehension. It can happen that the variable only occurs once and therefore not really used:

```
[1 for x in ["elements", "not", "relevant"]]
```

Result:

```
[1, 1, 1]
```

In this case the variable `_` is often used instead:

```
[1 for _ in ["elements", "not", "relevant"]]
```

Here it would in fact have been smarter to obtain the repetition using `*` as in the previous exercise.

A final example with a list from a list of lists:

```
[len(a) for a in [[32, 8, 4, 1], [16, 2]]]
```

Result:

```
[4, 2]
```

Use list comprehension to print a list of square numbers:

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

Use the following subexpression: `[i + 1 for i in range(10)]`

Opgave 5

I Python kan funktioner returnere flere værdier ved hjælp af tupler, fx:

```
def minmax(x, y):
    return min(x, y), max(x, y)

def main():
    v, w = minmax(42, 13)
    print(v, w)

main()
```

Udskrift:

```
13 42
```

Udvid funktionen `minmax`, så 3 værdier returneres, nemlig minimum, gennemsnit og maksimum:

```
def main():
    v, a, w = minmax(42, 13)
    print(v, a, w)
```

Udskrift:

```
13 27.5 42
```

Opgave 6

Oversæt Case Study: Zip Code Lookup i bogens afsnit 6.5 side 432 og frem til Python.

Her er eksemplet fra bogen:

```
Welcome to the zip code database.
Give me a 5-digit zip code and a
proximity, and I'll tell you where
that zip code is located, along
with a list of other zip codes
within the given proximity.

What zip code are you interested in? 98104
And what proximity (in miles)? 1

98104: Seattle, WA
zip codes within 1.0 miles:
    98101 Seattle, WA, 0.62 miles
    98104 Seattle, WA, 0.00 miles
    98154 Seattle, WA, 0.35 miles
    98164 Seattle, WA, 0.29 miles
    98174 Seattle, WA, 0.35 miles
```

Hent filen `zipcode.txt` på Learn.

Benyt evt. filen `zip_lookup_skeleton.py` på Learn som udgangspunkt. Nøgleordet `pass` gør ingenting og fungerer som pladsholder, da funktionsdefinitioner ikke kan være tomme.

Hvis `1/3` ønskes med 2 decimaler, så benyt `format(1/3, ".2f")` (giver `0.33` i dette tilfælde).