COMP105 Lecture 5

Tuples

Tuples

Functions always return something, but what if we want to return more than one thing from a function?

A **tuple** allows us to bind two or more values together

Examples:

```
(1, 2)

("A", "few", "words")

(6, "six")
```

Tuples

Tuples can have any size (at least 2)

```
("One", "Two", "Three", "Four", "Five", "Six")
```

The size should be thought of as being **fixed**

- It is not easy to change the length of a tuple

Tuples can mix types

```
(1, "Two", 3, "Four", 5, "Six")
```

Tuples in action

```
area_and_perimeter r =
            (pi * r * r, 2 * pi * r)

ghci> area_and_perimeter 1
(3.141592653589793,6.283185307179586)
```

```
rect_area (width, height) = width * height

ghci> rect_area (2, 4)
8
```

Tuples and function syntax

As we've just seen, you can pass a tuple to a function:

```
f (a, b) = a + b
```

This seems to give a (more familiar) alternative to the standard syntax:

```
f a b = a + b
```

- Both formulations will work
- There are good reasons to prefer the standard syntax whenever possible
- We will come back to this later in the course

Exercises

1. Write a function `exercise1` that takes one parameter `x` and returns `x/10` and `100*x` in a tuple.

2. Write a function `exercise2` that takes a tuple with three numbers and returns their sum

3. Write a function `exercise3` that takes a tuple with three elements, and returns a three element tuple with the elements in the opposite order