Computing for Mathematics: Handout 4

This handout contains a summary of the topics covered and an activity to carry out prior or during your lab session.

At the end of the handout is a specific coursework like exercise.

For further practice you can do the exercises available at the combinatorics chapter of Python for Mathematics.

1 Summary

The purpose of this handout is to cover combinatorics which corresponds to the combinatorics chapter of Python for Mathematics.

The topics covered are:

- Generating and counting permutations and combinations of elements.
- Directly computing binomial coefficients.

2 Activity

We will be tackling the problem from the tutorial of the combinatorics chapter of Python for Mathematics.

The digits 1, 2, 3, 4 and 5 are arranged in random order, to form a five-digit number.

1. How many different five-digit numbers can be formed?
2. How many different five-digit numbers are:
   (a) Odd
   (b) Less than 23000

There are instructions for how to do all of this is in the combinatorics chapter of Python for Mathematics.

1. Create a variable `digits` which has value the collection of integers 1, 2, 3, 4 and 5.
2. Use the `itertools.permutations` tool to create the variable `permutations` which has value the permutations of the integers 1, 2, 3, 4 and 5.
3. Convert the `permutations` variable to a `tuple` and use the `len` command to count the number of different permutations that exist.
4. Use the `sum` command to calculate:
$$\sum_{\pi \in \Pi} \pi_5 \mod 2$$
where Π is the collection of all permutations.
5. Use the `sum` command to calculate:
$$\sum_{\pi \in \Pi \text{ if } \pi_1 10^4 + \pi_2 10^3 + \pi_3 10^2 + \pi_4 10 + \pi_5 \leq 23000} 1$$
where Π is the collection of all permutations.

3 Coursework like exercise

1. Create a variable `number_of_permutations` that gives the number of permutations of size 4 of:

   `pets = ("cat", "dog", "fish", "lizard", "hamster")`

   Do this by generating and counting them.

2. Create a variable `direct_number_of_permutations` that gives the number of permutations of `pets` of size 4 by direct computation.

4 Summary examples

Create the collection: (A, A, B).

```
collection = ("A", "A", "B")
```

Obtain the 2nd element in the collection (A, A, B).

```
collection = ("A", "A", "B")
collection[1]
```

Check a boolean condition like if "C" $\in (A, A, B)$.

```
collection = ("A", "A", "B")
"C" in collection
```

Create the collection of integers from 1 to 11:

```
range(1, 12)
```

Create the permutations of the collection (A, A, B) of size 2.

```
import itertools
collection = ("A", "A", "B")
itertools.permutations(collection, r=2)
```

Create the combinations of the collection (A, A, B) of size 2.

```
import itertools
collection = ("A", "A", "B")
itertools.combinations(collection, r=2)
```

Calculate $\sum_{i=1 \text{ if } i \text{ even}}^{20} i^2$

```
sum(i ** 2 for i in range(1, 21) if (i % 2 == 0))
```

Obtain 5!:

```
import math
math.factorial(5)
```

Obtain $\binom{5}{2}$:

```
import scipy.special
scipy.special.comb(5, 2)
```

Obtain ${}^5P_2$:

```
import scipy.special
scipy.special.perm(5, 2)
```