Combinations With Repetition

Combinations represent the number of different possible ways we can **pick** a number of elements. In special cases we can have repetition in combinations and for those we use a different formula.

$$\bar{C}(n,p) = \frac{(n+p-1)!}{(n-1)!\,p!}$$

Combinations with repetition

Total number of elements in the sample space

Number of elements we need to select

Now that you know what the formula looks like, we are going to walk you through the process of deriving this formula from the Combinations *without* repetition formula. This way you will be able to fully understand the intuition behind and not have to bother memorizing it.

Applications of Combinations with Repetition

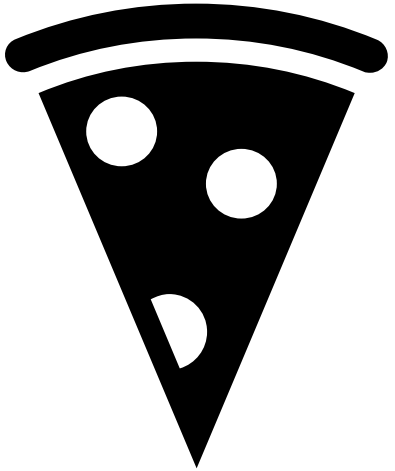
To understand how combinations with repetition work you need to understand the instances where they occur.

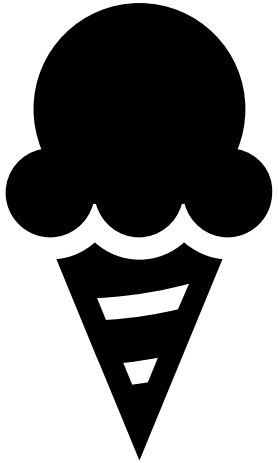
We use combinations with repetition when the events we are dealing with, have sufficient quantity of each of the distinct values in the sample space.

One such example is the toppings on a pizza.

We can order extra of any given topping, so **repetition is allowed**. However, we **do not care about the order** in which the toppings are put on top of the pizza, so we cannot use variations.

Similar examples include picking the ice-cream flavours for a Sundae melt or the players for a Fantasy Football Team.





Pizza Example

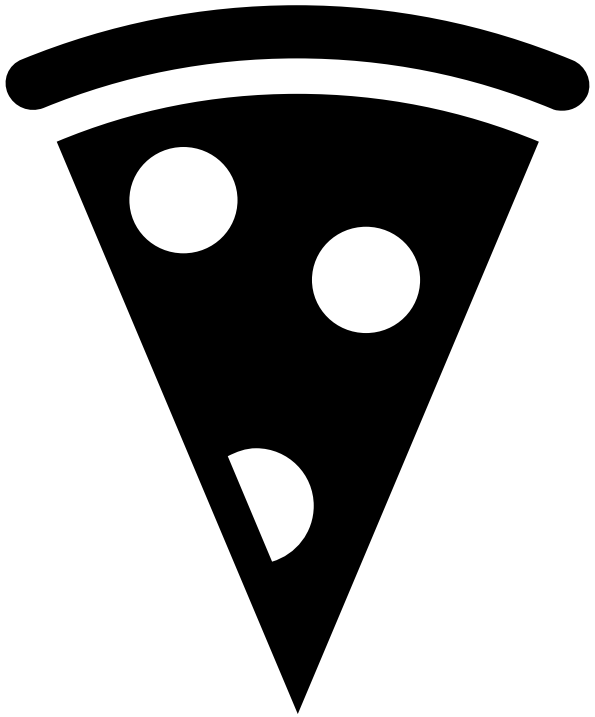
To get a better grasp of the number of combinations we have, let us explore a specific example.

You are ordering a 3-topping pizza from your local pizza place but they only have 6 topping left because its late.

The toppings are as follows:
cheddar **c**heese, **o**nions, **g**reen peppers, **m**ushrooms, **p**epperoni and **b**acon.

Your pizza can have 3 different toppings, or you can repeat a topping up to 3 times.

You can either order a pizza with 3 different toppings, a pizza with 3 identical toppings or a pizza with 2 different toppings but having a double dose of one of them.



Methodology

The methodology we use for such combinations is rather abstract. We like to represent each type of pizza with a special sequence of 0s and 1s. To do so, we first need to select a specific order for the available ingredients.

We can reuse the order we wrote down earlier:
cheddar **c**heese, **o**nions, **g**reen peppers, **m**ushrooms, **p**epperoni and **b**acon.

For convenience we can refer to each ingredient by the associated letter we have highlighted (e.g "c" means cheese, and "o" means onions).

To construct the sequence for each unique type of pizza we follow 2 rules as we go through the ingredients in the order we wrote down earlier.

1. If we want no more from a certain topping, we write a **0 and move to the next topping**.
2. If we want to include a certain topping, we write a **1 and stay on the same topping**.
    - Not going to the next topping allows us to indicate if we want extra by adding another 1, before we move forward. Say, if we want our pizza to have extra cheese, the sequence would begin with "1, 1".
    - Also, we always apply rule 1 before moving on to another topping, so the sequence will actually start with "1, 1, 0".

Pizzas and Sequences

If we need to write a "0" after each topping, then every sequence would consist of 6 zeroes and 3 ones.

Let's look at some pizzas and the sequences they are expressed with.

A pizza with **cheese** and **extra peperoni** is represented by the sequence 1,0,0,0,0,1,1,0,0.

A vegan variety pizza with **onions, green peppers and mushrooms** would be represented by the sequence 0,1,0,1,0,1,0,0,0.

Now, what kind of pizza would the sequence **0,0,1,0,0,0,1,1,0** represent?

We can put the sequence into the table and see that it represents a cheese pizza with extra bacon.

C	O	G	M	P	B
1,0	0	0	0	1,1,0	0
0	1,0	1,0	1,0	0	0
0	0	1,0	0	0	1,1,0

Always Ending in 0

Notice how all the sequences we have examined end on a 0:

- 1,0,0,0,0,1,1,0,**0**
- 0,1,0,1,0,1,0,0,**0**
- 0,0,1,0,0,0,1,1,**0**

This is no coincidence, since according to rule 1 of our algorithm, we need to add a "0" at the end of the sequence, regardless of whether we wanted bacon or not.

That means that only the first 8 elements of the sequence can take different values.

Each pizza is characterized by the positions of the 3 "1"s in the sequence. Since only 8 of the positions in the sequence can take a value of "1", then the number of different pizzas would be the combination of any 3 of the 8 positions.

C	O	G	M	P	B
1,0	0	0	0	1,1,0	0
0	1,0	1,0	1,0	0	0
0	0	1,0	0	0	1,1,0

Positions

As stated before, we have 3 “1s” and 8 different positions. Therefore, the number of pizzas we can get would be the number of combinations of picking 3 elements out of a set of 8. This means we can transform combinations **with** repetition to combinations **without** repetition.

$$\bar{C}(6,3) = \mathrm{C}(8,3)$$

Let’s observe the values 3 and 8 for a moment and try to generalize the formula. “3” represents the amount of toppings we need to pick, so it is still equivalent to p.

“8” represents the number of positions we have available for the ones. We had 3 + 6, or 9 positions in total, but we knew the last one could not contain a “1”. Thus, we had “n + p - 1” many positions that could contain a 1.

$$\bar{C}(n,p) = \mathrm{C}(\mathrm{n} + \mathrm{p} - 1, \mathrm{p})$$

The Final Step

Now that we know the relationship between the number of combinations **with** and **without** repetition, we can plug in "n+p-1" into the combinations without repetitio formula to get:

$$\bar{C}(n,p) = \mathrm{C}(\mathrm{n}+\mathrm{p}-1,\mathrm{p}) = \frac{(\mathrm{n}+\mathrm{p}-1)!}{((\mathrm{n}+\mathrm{p}-1)-\mathrm{p})!\,\mathrm{p}!} = \frac{(\mathrm{n}+\mathrm{p}-1)!}{(\mathrm{n}-1)!\,\mathrm{p}!}$$

This is the exact same formula we showed you at the beginning.
Before we you continue to the next lecture, let's make a quick recap of the algorithm and the formula.

1. We started by ordering the possible values and expressing every combinations as a sequence.
2. We examined that only certain elements of the sequence may differ.
3. We concluded that every unique sequence can be expressed as a combination of the positions of the "1" values.
4. We discovered a relationship between the formulas for combinations with and without repetition.
5. We used said relationship to create a general formula for combinations with repetition.