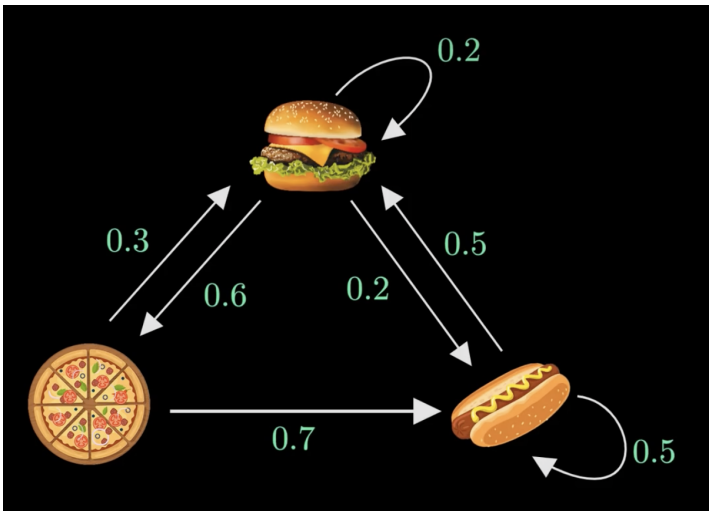
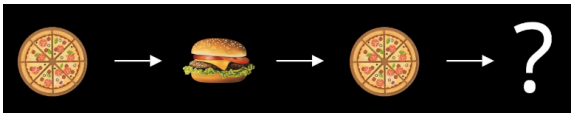
Markov's inequality

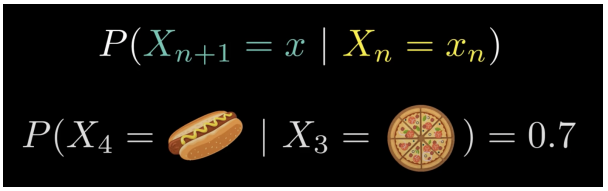
Let's say there is a restaurant that only serves 3 types of food: hamburger, pizza, and hot dog. But they follow a strange rule: They only serve one of these three items, and it depends on what they served yesterday. In other words, there is a way to predict what they will serve tomorrow if you know what they are serving today. For example, there is a 60% chance that tomorrow will be pizza day given that today is hamburger day (I will represent it as weighted arrows, the arrow comes from the current state and points to the future state).



There is a 20% chance they will be serving hamburgers again tomorrow, and this is represented by the self-guiding arrow. Each arrow is called a transition is called a transition from one state to another. The diagram you see is actually a Markov chain. *The most important property of a Markov chain is that the future state depends only on the current state and not on previous steps.* Mathematically, we can say that the probability of that $P\ (X_{n+1} = x\ =\ x\ \ |X_1 = x_1, X_2 = x_2, ..., X_n = x_n)$. Suppose the restaurant served pizza on the first day, a hamburger on the second, and again pizza on the third day. And now, what is the likelihood that they serve hot dogs on the fourth day?

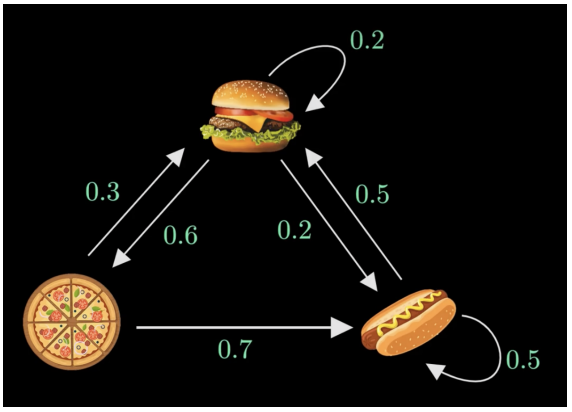


We need to look at the third day and it turns out to be 0.7.



Task 1

1)



Calculate with this matrix:

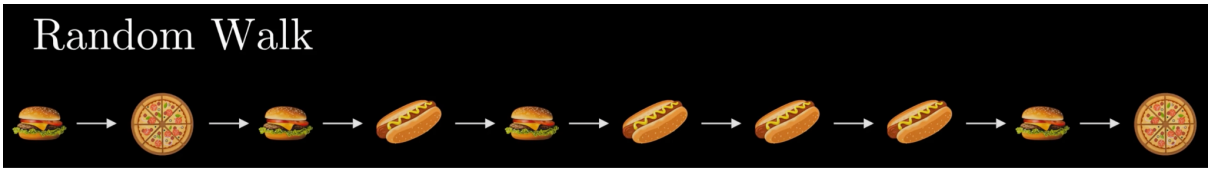
1) [0,5 0,5 0] (hot dog - hot dog - hot dog - pizza)
2) [0,3 0,2 0,5] (pizza - hamburger - hamburger - hot dog)
3) [0,2 0,2 0,6] (hamburger - hamburger - pizza)
4) [0,5 0,2 0,3] (hot dog - hamburger - hamburger - pizza)
5) [0,7 0 0,3] (pizza - hot dog - pizza - hamburger)

2) Do a Random walk like in step 1. What happens after 10000 steps

! we are interested in what happens in the end, whether these probabilities converge to fixed values or continue to change forever.
*you can write your answer and put screen of your code here

Let's explore our Markov's chain by doing a random walk. Imagine we're at the bottom of a hamburger. Let's see how they serve for 10 days. After 10 steps we have a situation similar to this.

[step 1]

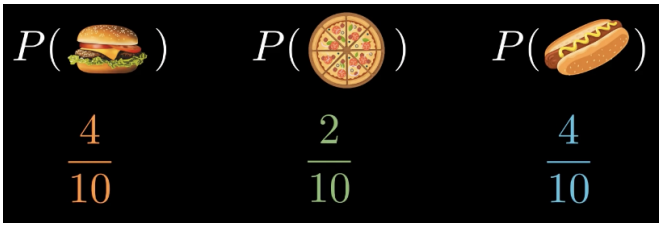


[step 2]

Now we want to find what is the probability corresponding to each of the food items or the probability distribution of the states.

[step 3]

It's pretty simple. Just divide the number of occurrences by the total number of days.



If we change the number of steps, then the probability will change.

Task 2

*Download the file cha.txt and put it in the same folder as the program code.

A Markov chain is a sequence of events where each new event depends only on the previous one. For example, one word may be followed by another word.

A Markov chain is a sequence of events where each new event depends only on the previous one. For example, one word may be followed by another word.

There are algorithms that are capable of generating text based on Markov chains. They study what connections can be between words, and then go through these connections and make up a new text.

For our work, the algorithm always needs the source text (aka corpus) - looking at this text, the algorithm will understand which words usually follow each other.

The larger the size of the source text, the more connections between the nets and the more varied the output text is.

The logic would be like this:

1) Take the file (cha.txt) and break it down into words.
2) We connect all the words that stand next to each other in pairs.
3) Using these pairs, we compose a dictionary of strings, where the first word is indicated and everything that can come after it.
4) Choose a random word to start.
5) We set the length of the output text and get the result. (100 words is enough)
6) Let's do everything step by step, as usual.

Hints

Splitting the source text. To quickly work with large data arrays, we will use the numpy library - it is written specifically for big dates, working with neural networks and processing large matrices. To install, you can use the pip3 install numpy command.

Generating pairs. To do this, we use a special generator command: yield. In functions, it works like a return - it returns some value, but we need it because of the peculiarities of its work. The fact is that yield does not store or remember any values - it just generates something, immediately forgets about

it and moves on to the next one. This is exactly how Markov chains work - they do not remember all previous states, but work only with specific pairs at the moment. As a result, we get all pairs of words that follow each other - with repetitions and in the order as they are located in the source text. Now you can compose a dictionary for strings.

Compiling a dictionary. Let's take the simplest path: we will not calculate the continuation probabilities for each word, but simply indicate with the second element in the pair all words that can be continuation. For example, we have such pairs in a variable:

It can be seen that "друг" occurs 3 times more often than other words, so the probability of its occurrence is ¾. But in order not to count the probability, we will do this:

1. Let's make a pair of "привет" → (это, друг, как, друг, друг).
2. When choosing, we will just randomly choose one of the values to continue.

Choosing a word to start. To make it completely unpredictable, we will also choose the initial word at random. The main requirement for an initial word is the first capital letter. Let's fulfill this condition like this:

- ➢ Let's pick the first word at random.
- ➢ Let's check if it contains big letters. For simplicity, let's say that if there is, then they are at the beginning and suit us.
- ➢ If there is - great, if not - select the word again and repeat all the steps.
- ➢ We do this until we find the right word.

Run the algorithm. We are almost ready to launch. The only thing left for us to do is set the number of words in the finished text (100 words is enough). After that, our algorithm will take the first word, add it to the chain, then choose a random continuation for this word, then choose a random continuation for the second word, and so on. He will do this until he has typed the required number of words, after which he will stop.

*For training, we took the eighth volume of Chekhov's complete works - stories and stories. It has about 150 thousand words, so the answer should be varied.

```
Глина, вода, а также вследствие некоторой степени выразительное лицо, руки… В большом красном футляре, похожем на м
оих руках. Ведь какая, подумаешь, нелепость! Мы стояли тучи он полагал, читатель не годится, что это в природе и тв
орчества Льва Николаевича Толстого. Все это были придуманы, чтобы готовить обед. — летом 1893 г. у вас ничто не ста
л рассказывать содержание этих черт в отсутствии определенного исторического периода, не бойся… Папа, коновал прише
л! — Посмотрите на сильном впечатлении, произведенном им, что Петр Михайлыч, идя к Орлову и, вспоминая ее грязью и
безобидное чудачество; но тотчас же. — Вы получите письмо, в ее тихом, ровном плаче,
```