لگاریتم با نگاهی دیگر :

عبارت لگاریتمی زیر را در نظر بگیرید

$$\log_{10} 1000 = 3$$

یعنی اگر عدد ده را سه بار در خود ضرب کنیم به عدد ۱۰۰۰ می‌رسیم. سری توانی زیر را در نظر بگیرید:

$$10^0, 10, 100, 1000$$

در سری توانی فوق عدد 10^0 پس از 3 گام به 1000 رسید. **بعبارتی لگاریتم گام شمار در سری توانی است. برای مثال دیگر :**

$$\log_2 1024 = 10$$

خب عبارت فوق چه می‌گوید؟ با ده بار تقسیم عدد ۱۰۲۴ به عدد ۱ می‌رسیم.

$$\mathbf{1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024}$$

وقتی که می‌گوییم جستجویی دودویی برای آرایه‌ای با طول 1024 در بدترین سناریو ۱۰ مقایسه دارد یعنی هر بار باید عدد درخواستی با عنصر میانه مقایسه شده و نیمه‌ای از آن دوباره بازخواست شود و این کار آنقدر صورت گیرد که طول آرایه به یک برسد؛ پس دقیقا بدنبال سری فوق هستیم. و تعداد مقایسه ها دقیقا برابر با گام‌های شکاندن آرایه است.

معمولا توابع لگاریتمی در درخت‌های بازگشتی بدلیل ساختار لگاریتمی درخت خودش را نشان می‌دهد:

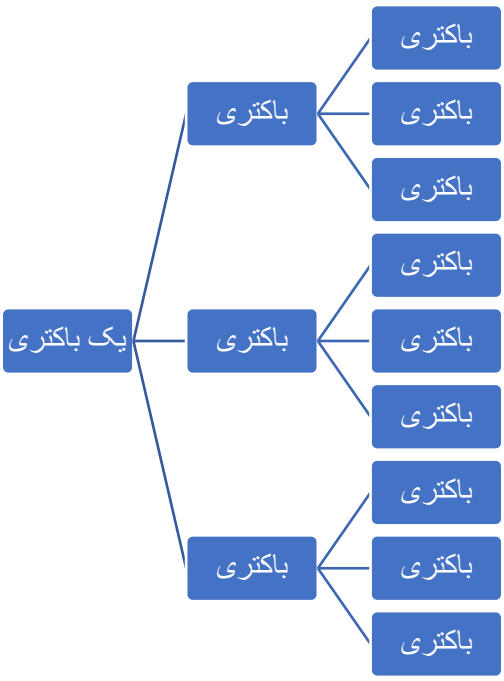
سری زیر را در نظر بگیرید :

$$\mathbf{1, 3, 9, 27, 81, 243}$$

اگر یک باکتری با هر تقسیم میوز فرض کنید سه باکتری شود (معمولا دو تا میشن 😊) بعد از ۳ گام چند باکتری خواهیم داشت ؟ بگذارید شکل را بکشیم ؟ بعد از دو بار تقسیم ۹ و پس از سه بار ۲۷، میبینیم که از سری توانی فوق پیروی می‌کند.

یک باکتری پس از چند گام به 2187 باکتری تبدیل می‌شود ؟ آیا میتوان شکل آن را کشید؟ خیر اما در ۷ گام تقسیم یک باکتری به این تعداد خواهد رسید چرا که ما تنها داریم گام‌های سری توانی را میشماریم.

$$\mathbf{\log_3 2187 = 27}$$



در رابطه با ساختارهای بازگشتی fun fact هایی نیز در رابطه با سری فیبوناچی، سری لوکاس و نسبت طلایی وجود دارد که در مخذن array-like یک فایل ipython با این مضمون وجود دارد. اما آنچه مهم است تحلیل درخت و تحلیل unfolding برای تحلیل ساختارهای بازگشتی است.