পাইথন নিয়ে একটু প্যাঁচাল পাড়ি।

Python Code :

```
class Params:
        def __init__(self, lambd, omega, k):
        self.lambd = lambd
        self.omega = omega
        self.k = k
```

Corresponding C++ code:

```
Class Params{
        double lamd;
        double omega;
        double k;
Public:
        Params(double lamd,double omega,int k)
        {
                this.lamd = lamd;
                this.omega= omega;
                this.k = k;
        }
}
```

এখন আসল কথায় আসি। একটা ব্যাংকে মানুষ আসবে সার্ভিস নিবে সেটাকে সিমুলেট করতে হবে। কোড স্যার পাইথনে দিয়ে দিসেন চাইলে কেউ অন্য ল্যাংগুয়েজেও করতে পারবা কিন্তু পাইথনে করাটাই সহজ হবে কারণ পুরা কাজ স্যার নিজেই করে দিসেন। তোমারে গিয়া খালি বিভিন্ন জায়গায় দুই তিন লাইন করে ঢুকাইতে হবে। একেকটা স্নিপেট নিয়ে আলোচনা করা যাক।

Params ক্লাস নিয়ে কিছু বলার নাই। States ক্লাসে একটা ব্যাংকের অবস্থা বুঝানোর জন্য যত ভেরিয়েবল লাগবে সব আছে। স্যার যেটা দিসেন তার বাইরেও অনেক কিছু লাগতে পারে। (আসলে লাগবে কারণ আমি কোড করে ফেলেছি)

আমাদের প্রথম কাজ States ক্লাসের update function এ। ধর একটা এরাইভাল হইল। তাহলে যদি সার্ভার কোনটা ফাঁকা থাকে তাহলে সে ডিরেক্ট সার্ভারে ঢুকে যাবে আর সব বিজি থাকলে তাকে কিউ এর শেষে ঢুকাতে হবে। সো বুঝা যাইতেসে States ক্লাসে যেই ভেরিএবল গুলা আছে সেগুলা আপডেটের কাজ এই ফাংশনে করতে হবে।

অবশ্যই arrival হইসে যেভাবে আপডেট হবে departure হইলে সেভাবে আপডেট হবে না। সো আপডেট করার আগে event.eventType চেক করে নিতে হবে।

States এর finish function এ কী করতে হবে? Update function এ আমরা সব কাজ করতে পারি না। যেমন এভারেজ বের তো আমরা সব ইভেন্টের টাইমে করতে পারব না।

একেবারে সিমুলেশন শেষ হয়ে গেলে এরপর আমরা এভারেজ বের করব। সেই কাজ finish এ করতে হবে।

এরপর আসি নতুন ইভেন্ট schedule করা নিয়ে। event schedule করার কাজ প্রতিটা ইভেন্টের process function এ করতে হবে। যেমন একটা arrival হলে নতুন একটা interarrival time generate করে নতুন arrival schedule করতে হবে। আবার একটা departure হলে যদি কিউ ফাঁকা না থাকে তাহলে সামনের মানুষটাকে ধরে এনে তার জন্য service টাইম generate করে আরেকটা departure event schedule করতে হবে।

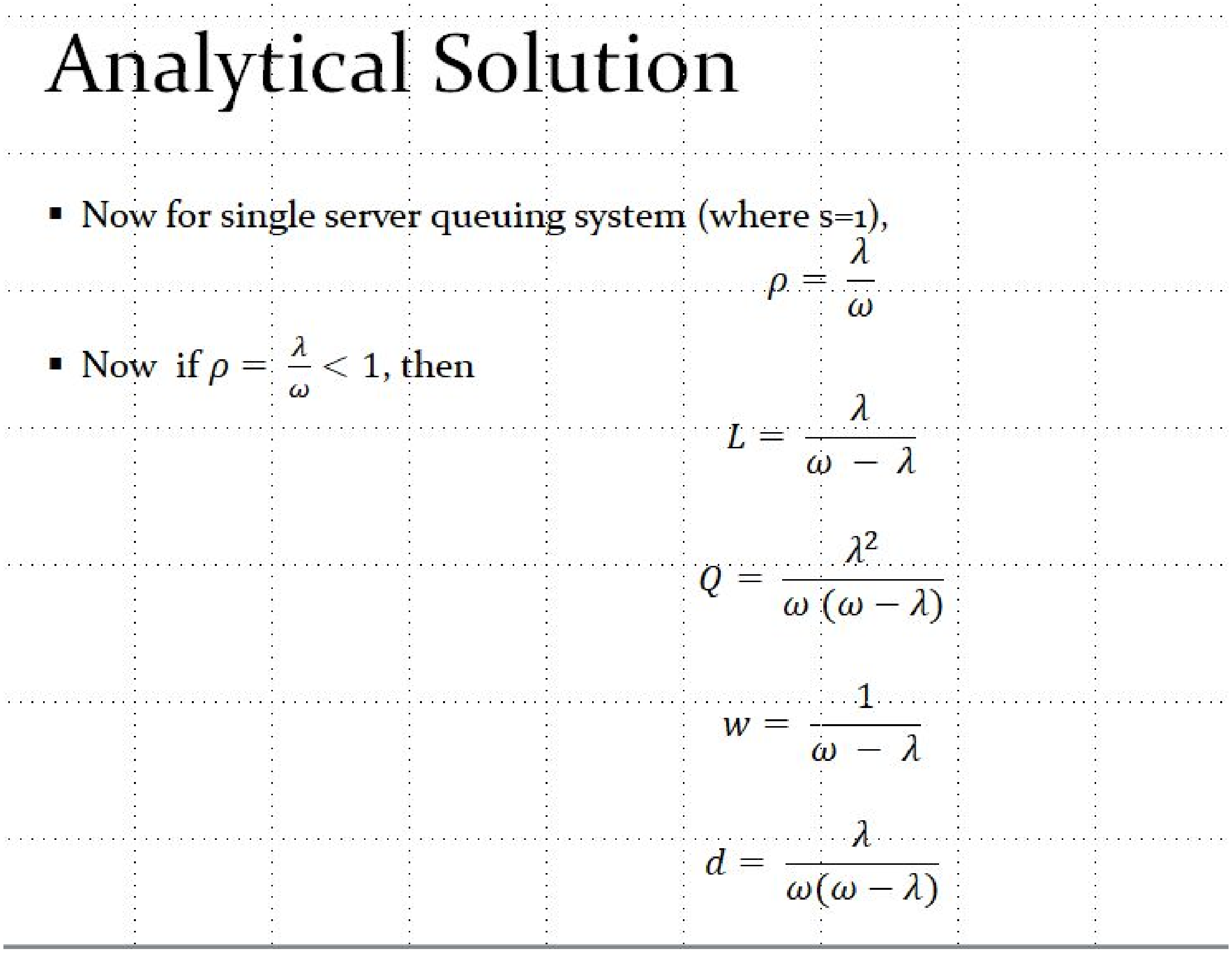
সো আমাদেরকে ArrivalEvent, DepartureEvent আর StartEvent এই তিনটা ক্লাসের process function implement করতে হবে। StartEvent এও নতুন এরাইভাল ইভেন্ট schedule করতে হবে। একদম শুরুর arrival টা।

এখানে একটা বিষয় বলে রাখি। আমাদেরকে যেই সার্ভিস টাইম আর inter arrival time generate করতে হবে সেটা অবশ্যই exponentially distributed হতে হবে। কিন্তু rand() function দিয়ে আমরা uniform distribution পাব। সেটা থেকে কীভাবে exponential এ যাইতে হয় সেটা স্যার ক্লাসে বলে দিসেন। সেই ফর্মুলা এপ্লাই করতে হবে।

এতটুকু ঠিক ভাবে করলে অনেক খানি কাজ শেষ। কোডের একদম নীচে যেখানে তিনটা experiment আছে সেটার মধ্যে খালি experiment1() run কর রেজাল্ট আসবে। এটা ঠিক আসসে নাকি সেটা বুঝার জন্য নীচের ছবির ফর্মুলা এপ্লাই করে সেটাও কনসোল এ প্রিন্ট করে রাখো।

Analytical Solution

- We want to estimate, L, Q, w, d
- d: steady-state average delay
- w: steady-state average waiting time
- Q: steady-state time-average number in queue
- L: steady-state time-average number in system



Analytical Solution

- Now for single server queuing system (where s=1),

$$\rho = \frac{\lambda}{\omega}$$

- Now if $\rho = \frac{\lambda}{\omega} < 1$, then

$$L = \frac{\lambda}{\omega - \lambda}$$

$$Q = \frac{\lambda^2}{\omega(\omega - \lambda)}$$

$$w = \frac{1}{\omega - \lambda}$$

$$d = \frac{\lambda}{\omega(\omega - \lambda)}$$

আশা করি হয়ে গেসে :D .

একটা প্রশ্ন আসতে পারে যে সিমুলেশন শেষ হবে কখন। এটা তোমার ইচ্ছা। হয় সময়ের উপর constraint দাও। যেমন সিমুলেশন ১২০ মিনিট চলবে। (তোমার কোড তাড়াতাড়িই শেষ হবে just sim.simclock = 120* 60 হয়ে গেলে শেষ) । অথবা ব্যাংকে ১০০০০ ঢুকে গেলে আর ঢুকতে দিবা না এভাবেও শেষ হতে পারে।

এরপর experiment2() টা রান করে দেখ। একটা প্লট আসবে। ওরকম করে তোমাকে আরও কিছু প্লট আকতে হবে। সেই কাজ তুমি experiment3() তে করবা। সেম কাজ খালি সার্ভারের সংখ্যা বাড়ায়া দিয়ে দিয়ে করতে হবে। যতই সার্ভার বাড়াবা দেখবা delay আর queue length কমতেসে।

N.B: States এ variable গুলা কীভাবে চেঞ্জ করতে হবে সেটা স্যার এর স্লাইডে details describe করা আছে।