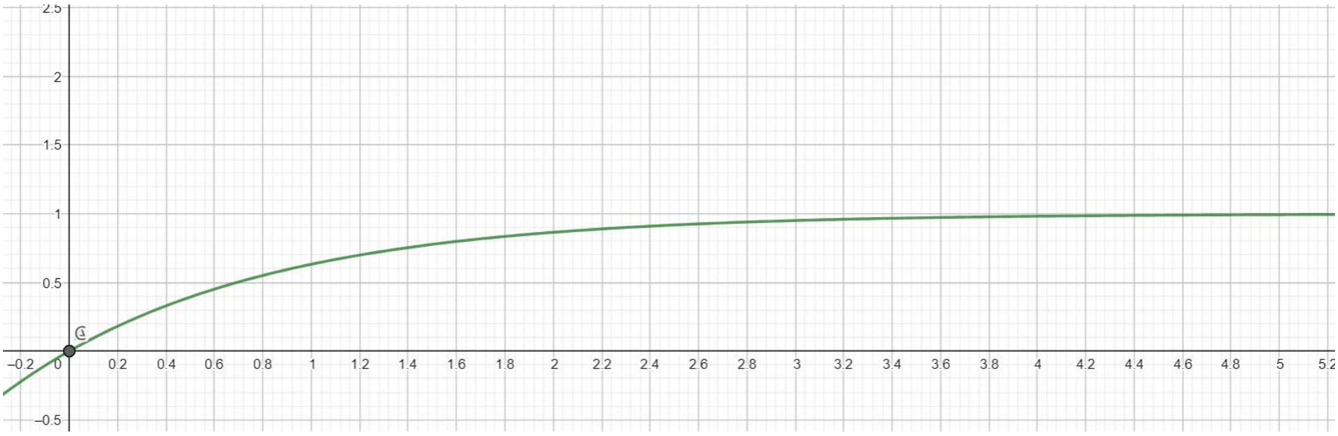
Problem1:

This is the CDF of exp(lambda=1):



The CDF of exponential random variable is $F_X(x)=1-e^{-x}$. With the definition of it , we can assume that the number we return from gen_exp has the similar property. To get the random variable that has $1-e^{-x}$ property, I first generate 1~10000 random variable and subtract it with 10000 to get a number that is less than 1. With this number I can g(x)=[-ln(1-x)] /lambda, whitch is the inverse of $F_X(x)$. The number I get will follow the exponential random variable.

The table:

x	1	2	3	4	5
F(x)	0.66541	0.87741	0.95396	0.98327	0.99395
$F_X(x)=1-e^{-x}$	0.63212	0.86466	0.950212	0.98168	0.99326

Problem2:

I first generate all the arrival time (exp(lambda)) random variable and add them every time I generate a new one to let the array(ta[5000]) be the time line.

```
for(i=1;i<5000;i++){
    a=gen_exp(lambda)+a;
    ta[i]=a;
    //printf("ta[%d]=%lf \n", i, ta[i]);
}                           ///Now I get all the arrival time
for(j=0;j<5000;j++){
    ts[j]=gen_exp(1.0);
    //printf("ts[%d]=%lf \n", j, ts[j]);
}                           ///Now I get all the service time
```

Then I generate the service time(ts[5000]). Let t be the time line. By using t, I can get the time a package spent in the node(T) and the number of packages left in the node when the package left.

There will be two situation when I have to manage package x. (x=1,2,3..)

Case 1:If the package came in with no packages waiting in the queue, then I set t to the time when it came in. The time it spend in the node will be its services time(ts[x]).To get N is to find how many packages came in while it is being served.

```
if(ta[x]>=t){                    //when there is no package waiting in the queue
    Ttotal= ts[x]+Ttotal;
    t=ta[x]+ts[x];
    for(y=x;y<5000;y++){
        if(ta[y]>t){
            break;
        }
    }
    Ntotal=y-x-1+Ntotal;
    //printf("n[%d]=%d \n", x, y-x);
}
```

Case2: When there are packages waiting in the node, the time it spend in the node will be the difference of the time it came in and the time it leaves(t-ta[x]+ts[x]). To get N is to find how many packages came in while it is being served(same as case 1).

```
else{                    //when there are packages waiting int the queue
    Ttotal=t-ta[x]+ts[x]+Ttotal;
    t=t+ts[x];
    for(y=x;y<5000;y++){
        if(ta[y]>t){
            break;
        }
    }
    Ntotal=y-x-1+Ntotal;
    //printf("n[%d]=%d\n", x, y-x);
}
```

After running all the package, I get the total of T and N. I divide it with 5000 to get the average and that will be the output.

The output table:

lambda	0.2	0.4	0.6	0.8	1.0	1.2
N	0.2972	0.7836	1.6422	3.60104	18.1428	328.069
T	1.190995	1.636482	2.404885	4.180995	17.369044	296.902

Problem:

There will be and overflow if the number of packages is larger than 5000. I've used static double in all my variables and %lf in my output line. There's still got something wrong (The picture below is when x=100000). I have to change the package number to 5000 so my code can run without error. In other words, the tolerance of my output is big.

```
Input lambda: 0.2
Ttotal=1.#INF00
N=111.049429      T=1.#INF00

--------------------------------
Process exited after 4.497 seconds with return value 0
請按任意鍵繼續 . . .
```