Practica 1:

Colas:

**Generar tiempos entre llegadas y tiempos de servicio**

RndExp[rate\_]:=-Log[RandomReal[]]/rate //N

interArrivals=Table[RndExp[mu],nm]

**Acumulamos llegadas:**

Arrivals = Accumulate[interArrivals]

**Función FIFO (returns departures)**

Fifo[arrivals\_,service\_]:=Module[{n, checktime},

n=0;

checktime=arrivals[[1]];

Map[(n++;If[checktime>=#,checktime+=service[[n]],checktime=#+service[[n]]])&,arrivals]

]

**Representación del Resultado**

ListStepPlot[{arrivals,departures}]

* Use Manipulate[f(x),{x,0,5,1}]

**Creamos “Montaña”**

1. Marcamos arrays Llegadas y salidas (1,-1) 🡪 Mark[arrivals,1]
2. Ordenamos 🡪 Event=Short[Join[arrivals1,departures1]]
3. Creamos la función ({tInidicio,tFinal,Nº Paquetes}):

Mont[lst\_]:=Module[{n,time},

n=0;

time=0;

Map[(time,time=#[[1]],If[#[[2]]==1,n++,n--])&,lst]

]

1. \*\*\*Recordatorio que con ##&[] será como un return en blanco\*\*\*

Podemos realizar el calculo de las probabilidades seleccionando los tiempos de inicio y final de la función anterior para un valor de n concreto, sumar esa diferencia de tiempos, y dividirla entre el tiempo total del experimento.

S&W , GBN, etc …

**Recordatorios iniciales:**

1. Carga de la librería: Get[“path”]
2. Visualización de las funciones:

Set Inicial + Funciones Importantes:

1. SetIniParDraw[tPropagacion,tProcesadoACK]
2. FIFO básico S&W , GBN:

Igual que el Fifo original, solo que deberemos de tener en cuenta el sumar al checktime un valor de 2\*tp+ts antes del condicional, y acordarse de inicializar el checktime antes del map con un valor de –(2\*tp+ts) y así comenzar en checktime =0 en el primer calculo.