

# MR. TOLDOS

Daniel Montemayor Carlin, Carlos Alberto Torres Cano, Oscar Medina Juárez, Arturo Emiliano Hernández Guajardo, Oscar Eduardo Fernández Maldonado

## RESUMEN

Mr Toldos es una tienda que su funcionalidad es proveer distintos tipos de productos como lonas, toldos entre otros, en donde cada uno de estos ofrece una gran variedad de combinaciones de colores, medidas muy convencionales para cualquier tipo de eventos que solicite el comprador.

Este proyecto buscamos poder visualizar mejor nuestro inventario de nuestro día siguiente de nuestra tienda al momento de simular como se venden nuestros toldos.

## PALABRAS CLAVE

Toldos, Inventario, Ventas, Tienda, Simulación, Local, Fabricación

## INTRODUCCIÓN

Para la realización de este proyecto nos estaremos enfocando en la distribución de toldos por medio de la tienda física y en línea con un tiempo de venta de 24hrs, en el cual la tienda física solo tiene una línea de atención y la línea de venta de la tienda en línea es del tiempo de 24hrs. Para al momento de simular poder obtener nuestro inventario del día siguiente y estar anticipados con el producto a reestablecer.

## DESCRIPCIÓN DEL SISTEMA

En este proyecto haremos una simulación del sistema de inventarios de un negocio de venta y entrega de lonas de toldos en distintos tamaños que surten de diferentes formas tanto en un local de manera presencial y en las distintas tiendas online como “Mercado Libre” entre otras empresas. Esta simulación nos ayudara a poder proveer restock de inventario para el día siguiente de nuestros toldos a fabricar.



Figura 1.- Toldo

## JUSTIFICACIÓN

Nosotros como equipo creemos que una simulación puede ser beneficiosa para la administración de este local que fabrica lonas ya que podríamos visualizar mejor el posible inventario sin necesidad de adquirirlo, pudiendo de esta forma manipular mejor el inventario del establecimiento sin tener pérdidas de

materiales para la fabricación de lonas o hasta las mismas lonas.

## FORMULACIÓN DEL SISTEMA

Conocer con un rango aproximado lo más cerca posible el stock de los días posteriores para así en base en eso poder nosotros proveer a nuestra clientela nuestros productos sin necesidad de comprar de más y que llegue a ser una pérdida. Las ventas de estas lonas es una página en línea 24 horas abierta. Si la demanda que se tiene ese día es mayor que la cantidad de lonas actuales, entonces no se hace el intercambio y se pierde el día. A su vez, la demanda es distinta dependiendo del color de lona. Finalmente, cada día se hace una renovación del inventario, agregando la misma cantidad de lonas diariamente al stock, esto depende de cada color.

## ELEMENTOS DEL SISTEMA

- Venta en línea abierta 24 horas.
- 5 clasificaciones de lonas divididas por color (azul, rojo, gris, blanco y negro).
- Se empieza con un stock de lonas inicial, al finalizar el día llegan nuevas lonas y son agregadas al stock total. Siempre llega la misma cantidad de lonas por día.
- Cada día llega una demanda de lonas, estas varían dependiendo del color de lona.

## PARÁMETROS

- $T_{limite}$  = Cantidad de días máximos a simular.

- $Renovacion$  = Cantidad de lonas nuevas que llegan por día.

## COMPONENTES

- Venta en línea (24hrs)

## VARIABLES DEL SISTEMA

### Endógenas

- $stockAzul, stockRojo, stockGris, stockBlanco, stockNegro$ : Cantidad de lonas que se encuentran en el inventario para ser vendidas.
- $ventasAzul, ventasRojo, ventasGris, ventasBlanco, ventasNegro$ : Cantidad de lonas que fueron vendidas según el color.

### Exógenas

- Demanda: Cantidad de lonas que clientes solicitan ese día. Esta es una lista que tiene N cantidad de elementos correspondientes a la cantidad de días simulados. Cada día contiene la cantidad de lonas demandas en ese particular día.

### Estado

- Reloj: Contador de días de la simulación.
- $T_{muerto}$ : Cantidad de días en los que no se pudo cumplir con la demanda.

## ANÁLISIS DE LOS DATOS DE ENTRADA

### Recopilación

Para nuestro proyecto obtuvimos la información desde la base de datos de la pagina oficial del negocio, la cual nos arrojó la información de la venta de cuantos toldos se vendieron en ese día y de qué medida.

A partir de estos datos nosotros consideramos usar lo de 30 días para el conteo de los toldos

*Revisar anexo 1 para revisar los datos*

### Análisis de datos

Para la realización de este proceso se utilizó la herramienta Stat::Fit, donde colocamos nuestras muestras de ventas de toldos en 30 días, procesando las ventas de 30 días para cada color de toldo.

### Resultados:

#### Azul:

autofit of distributions		
distribution	rank	acceptance
Poisson[2.77]	100	do not reject

#### Rojo:

autofit of distributions		
distribution	rank	acceptance
Poisson[0.733]	100	do not reject

#### Gris:

autofit of distributions		
distribution	rank	acceptance
Poisson[0.933]	100	do not reject

#### Blanco:

autofit of distributions		
distribution	rank	acceptance
Poisson[2.47]	100	do not reject

#### Negro:

autofit of distributions		
distribution	rank	acceptance
Poisson[0.667]	100	do not reject

### Estadísticos descriptivos:

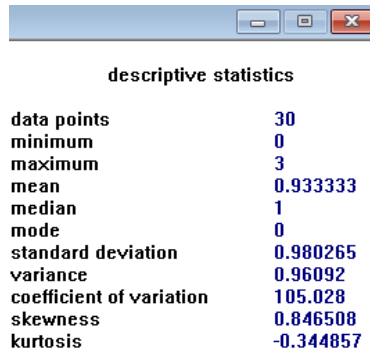
#### Azul:

descriptive statistics	
data points	30
minimum	0
maximum	7
mean	2.76667
median	3
mode	1
standard deviation	1.71572
variance	2.94368
coefficient of variation	62.0138
skewness	0.783741
kurtosis	0.415656

#### Rojo:

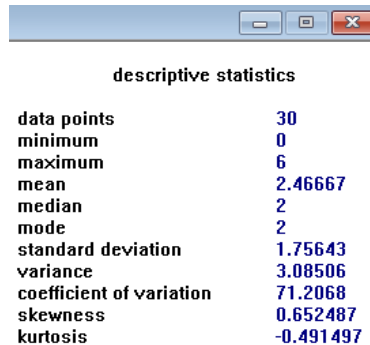
descriptive statistics	
data points	30
minimum	0
maximum	4
mean	0.733333
median	0
mode	0
standard deviation	0.980265
variance	0.96092
coefficient of variation	133.673
skewness	1.52427
kurtosis	2.25362

### Gris:



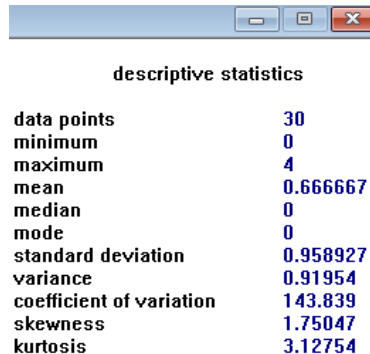
descriptive statistics	
data points	30
minimum	0
maximum	3
mean	0.933333
median	1
mode	0
standard deviation	0.980265
variance	0.96092
coefficient of variation	105.028
skewness	0.846508
kurtosis	-0.344857

### Blanco:



descriptive statistics	
data points	30
minimum	0
maximum	6
mean	2.46667
median	2
mode	2
standard deviation	1.75643
variance	3.08506
coefficient of variation	71.2068
skewness	0.652487
kurtosis	-0.491497

### Negro:



descriptive statistics	
data points	30
minimum	0
maximum	4
mean	0.666667
median	0
mode	0
standard deviation	0.958927
variance	0.91954
coefficient of variation	143.839
skewness	1.75047
kurtosis	3.12754

### Definición de características de operación:

#### Azul:

Utilizando nuestra muestra de datos de ventas de toldos de color azul en 30 días se obtuvo como resultado una distribución Poisson, con una lambda de 2.76667.

### Rojo:

Tomando en cuenta nuestra muestra de datos de ventas de toldos de color rojo en 30 días se obtuvo como resultado una distribución Poisson, con una lambda de 0.73333.

#### Gris:

Usando nuestra muestra de datos de ventas de toldos de color gris en 30 días, se obtuvo como resultado una distribución Poisson, con una lambda de 0.93333.

#### Blanco:

Utilizando nuestra muestra de datos de ventas de toldos de color blanco en 30 días, obtuvimos como resultado una distribución Poisson con una lambda de 2.46667.

#### Negro:

Utilizando nuestra muestra de datos de ventas de toldos de color negro en 30 días, se obtuvo como resultado una distribución Poisson con una lambda de 0.666667.

### Generación de los valores

La generación de números aleatorios está basada en la media obtenida en el análisis de datos, de tal forma que la media de cada color será utilizada como su valor de lambda, además de esto se procurará mantener el stock actual para nuestras muestras en 0 y los días a simular se establecerán como 30.

#### Azul:

Para generar las ventas de toldos azules se utilizó una lambda de 2.7666666 y con esta se pudieron generar nuestras variables

aleatorias de Poisson, como se puede ver en la Figura 2.

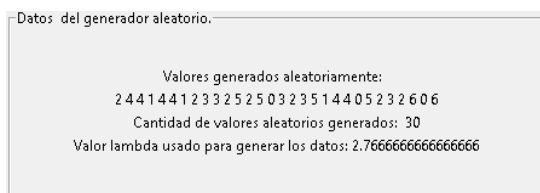


Figura 2.- Poisson de Azul

### Rojo:

Para la generación de ventas de toldos de color rojo se utilizó una lambda de 0.7333333 y con esta pudimos generar nuestras variables aleatorias de Poisson, como podemos observar en la Figura 3.

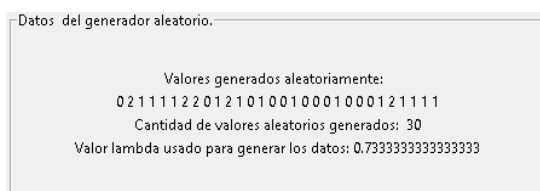


Figura 3.- Poisson de Rojo

### Gris:

Para generar las ventas de toldos grises utilizamos una lambda de 0.9333333 y con esta se pudieron generar nuestras variables aleatorias de Poisson, como se puede ver en la Figura 4.

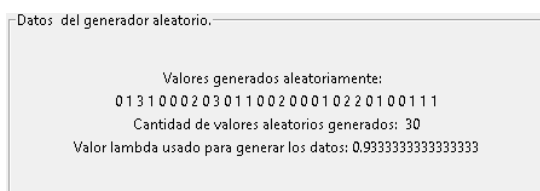


Figura 4.- Poisson de Gris

### Blanco:

Para generar las ventas de toldos blancos se utilizó una lambda de 2.4666666 y con esta se pudo generar nuestras variables aleatorias de Poisson, como se puede observar en la Figura 5.

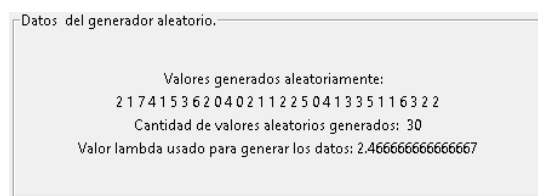


Figura 5.- Poisson de Blanco

### Negro:

Para la generación de las ventas de toldos negros se utilizó una lambda de 0.6666666 y con esta se pudieron generar nuestras variables aleatorias de Poisson, como se observa en la Figura 6.

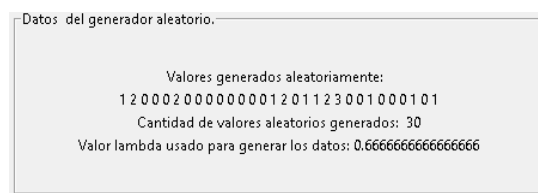
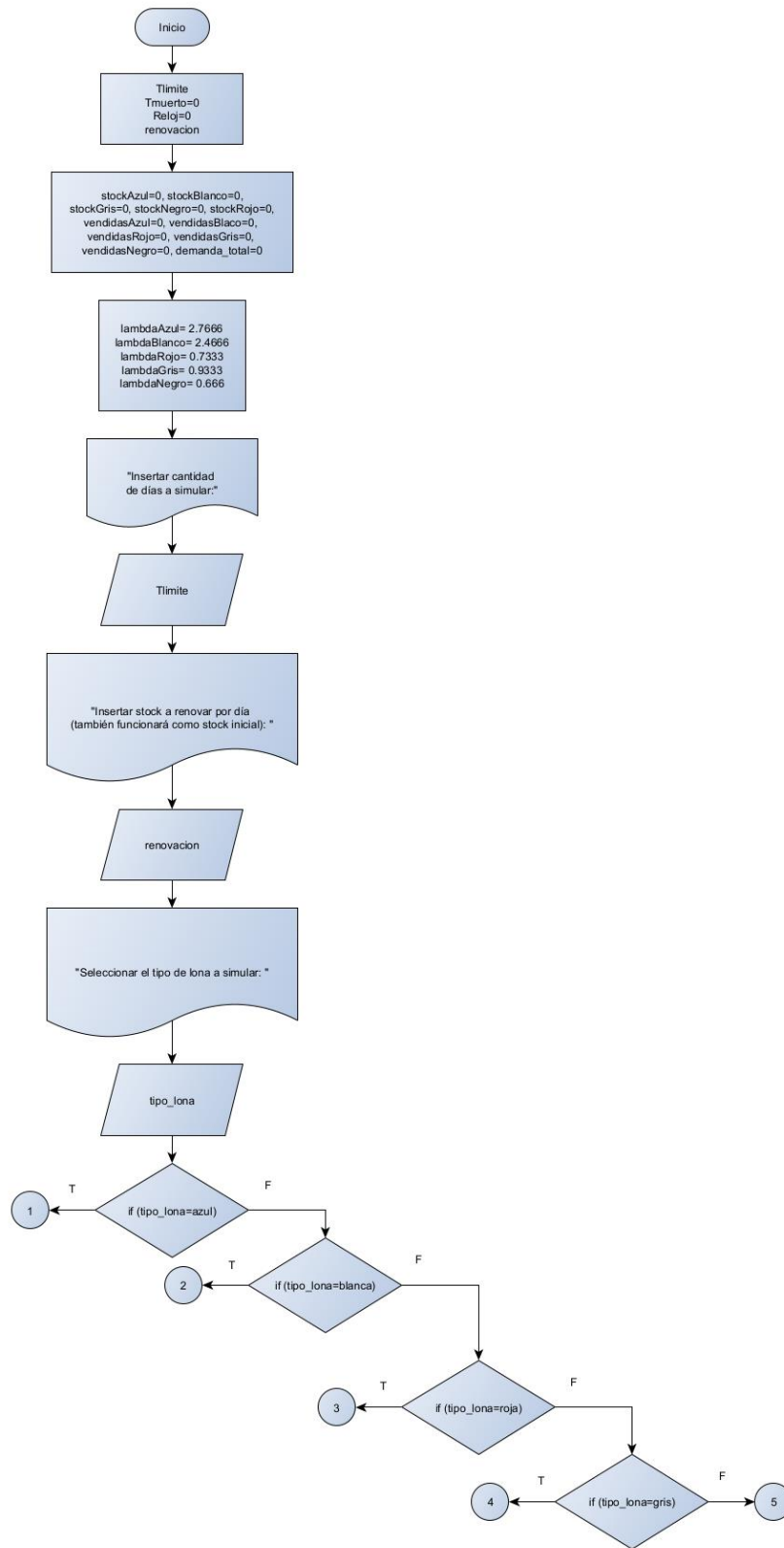
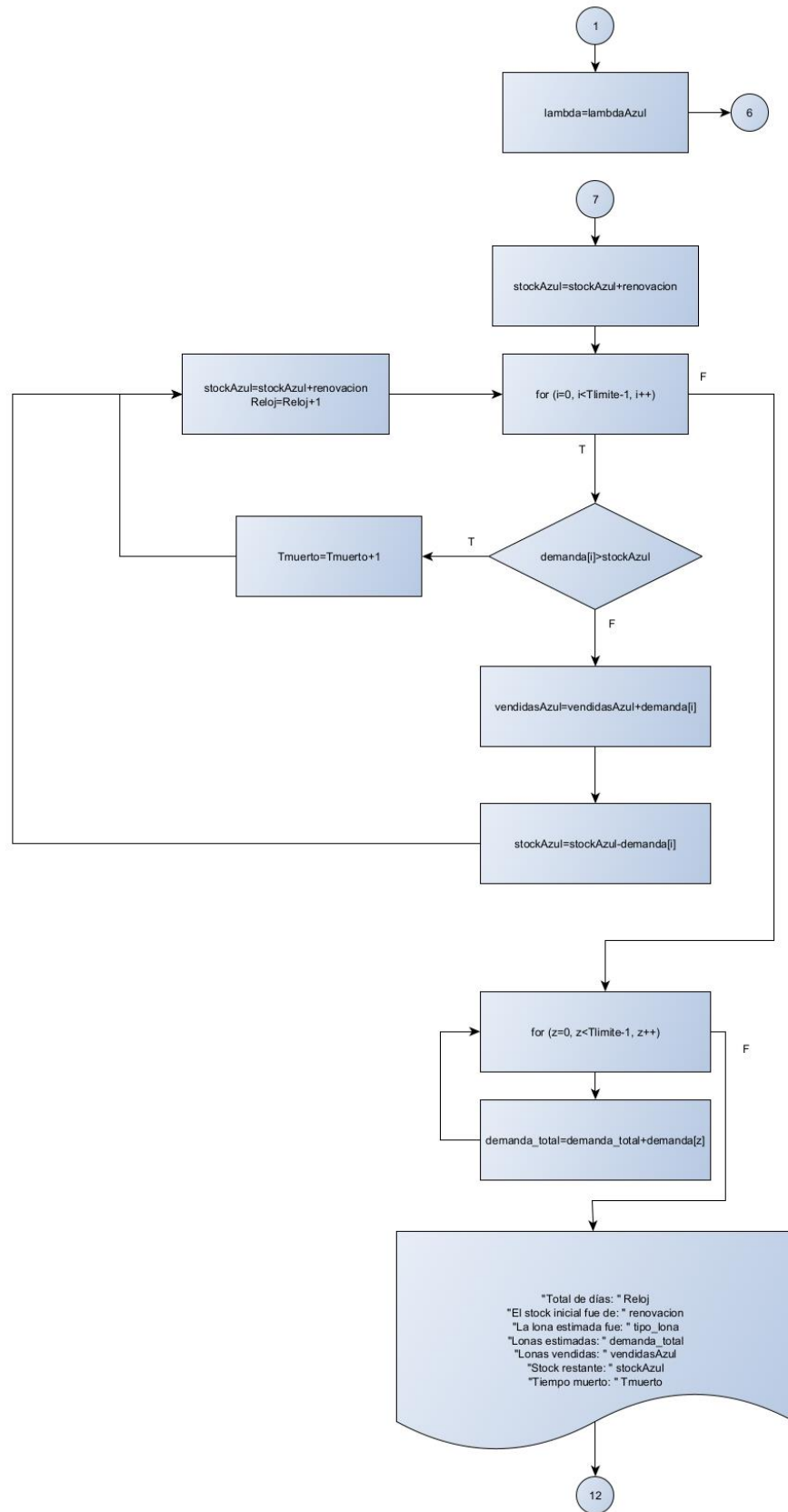
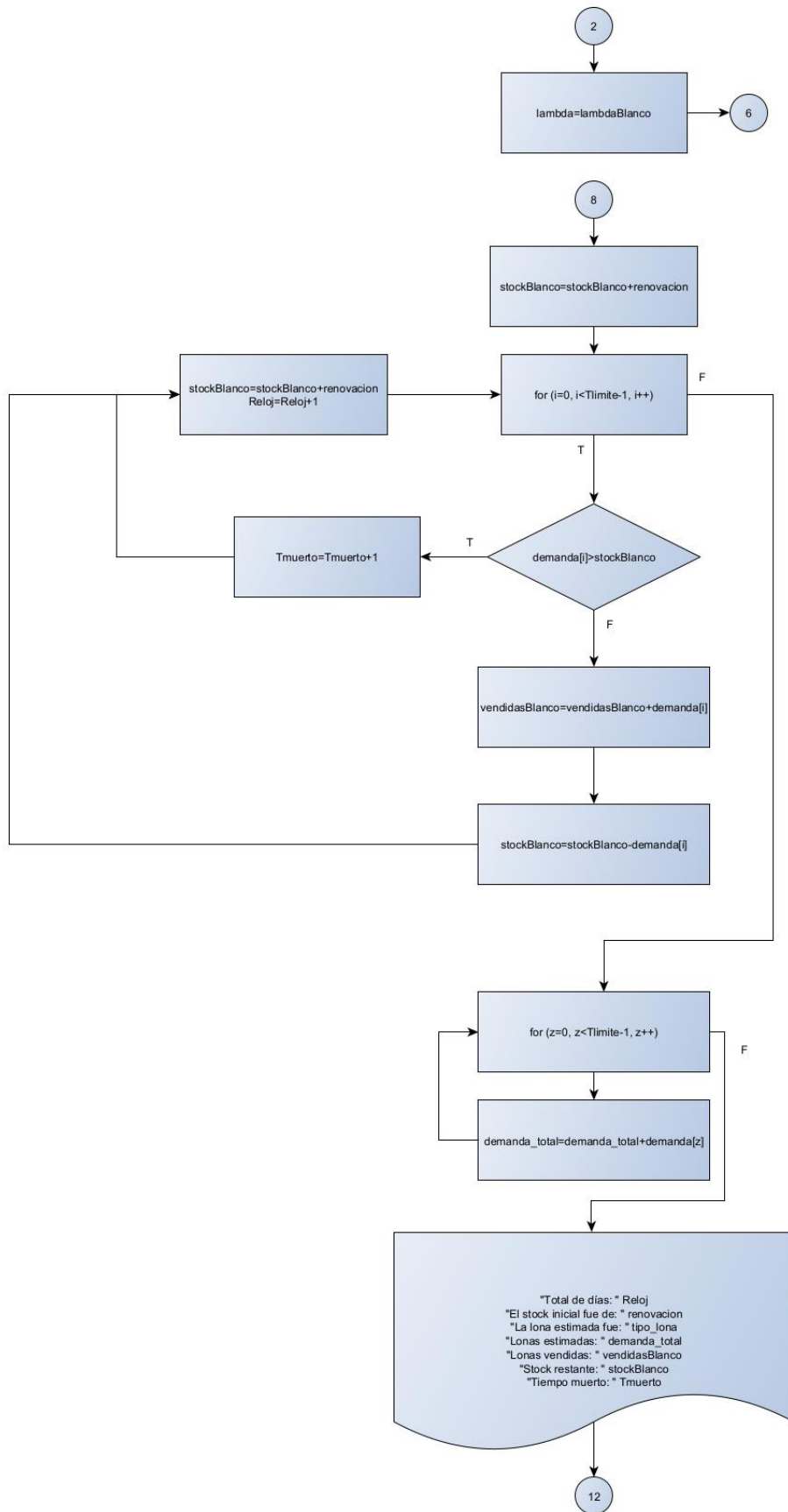


Figura 6.- Poisson de Negro

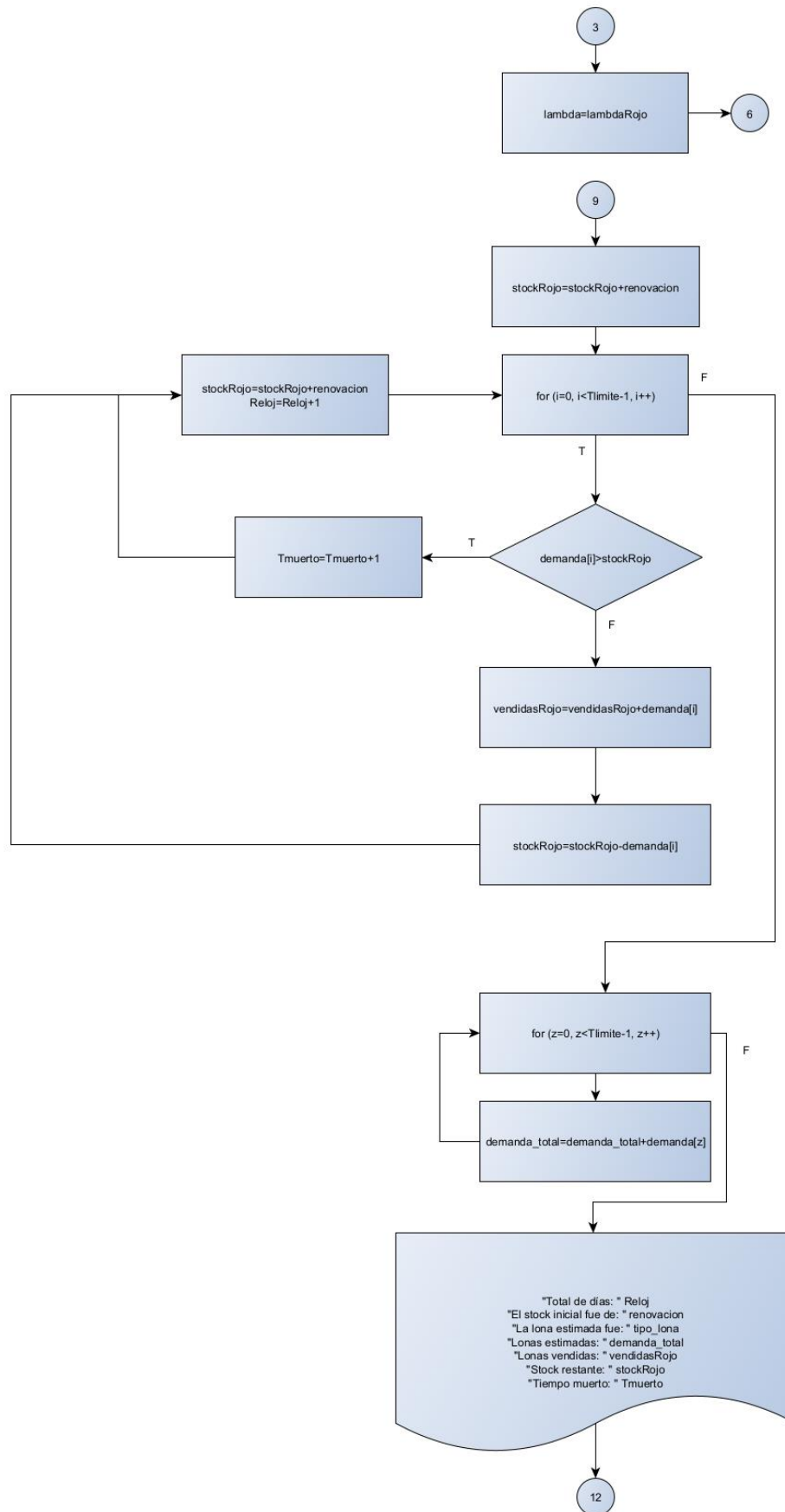
## MODELADO DE SIMULACIÓN

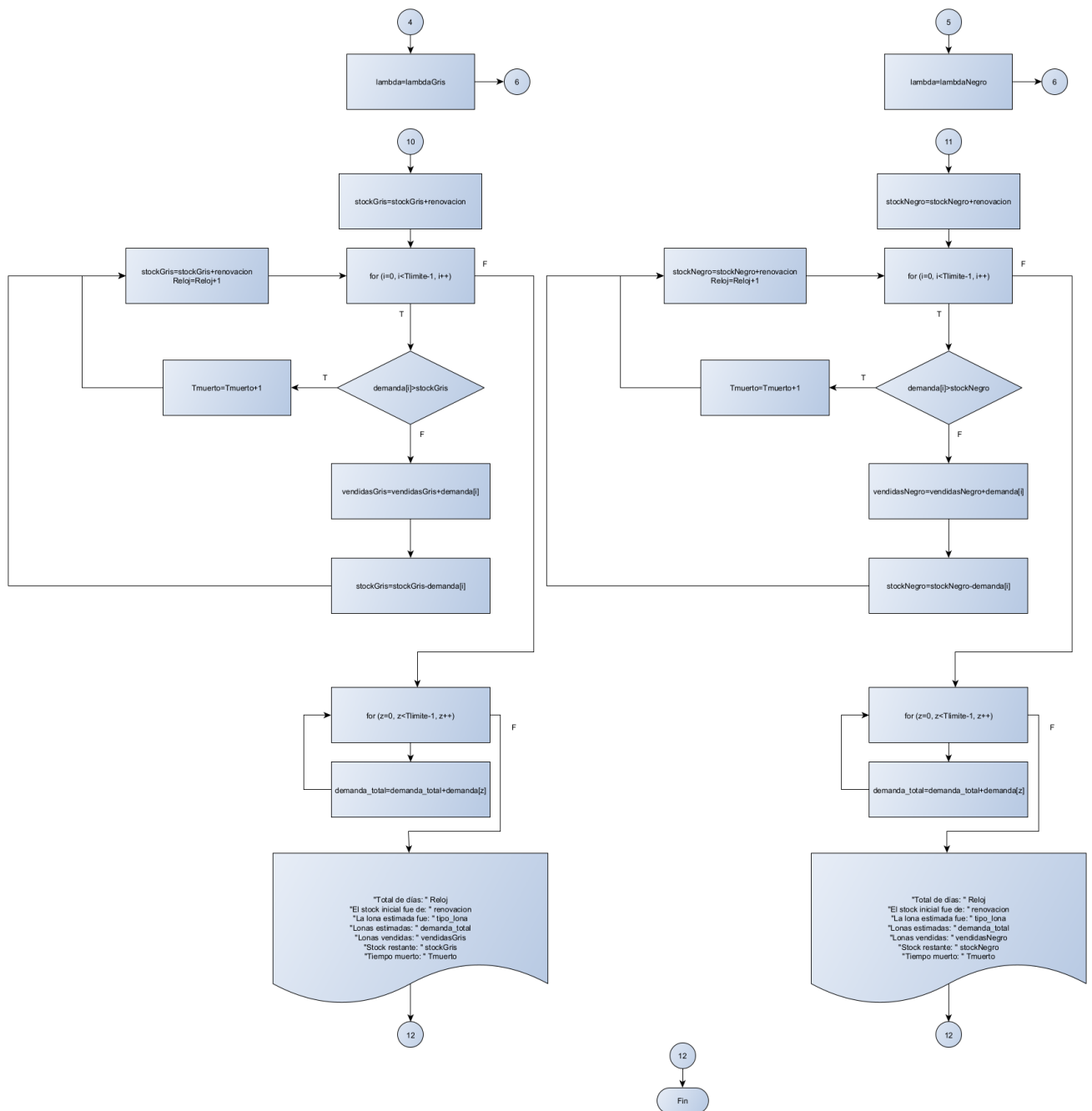


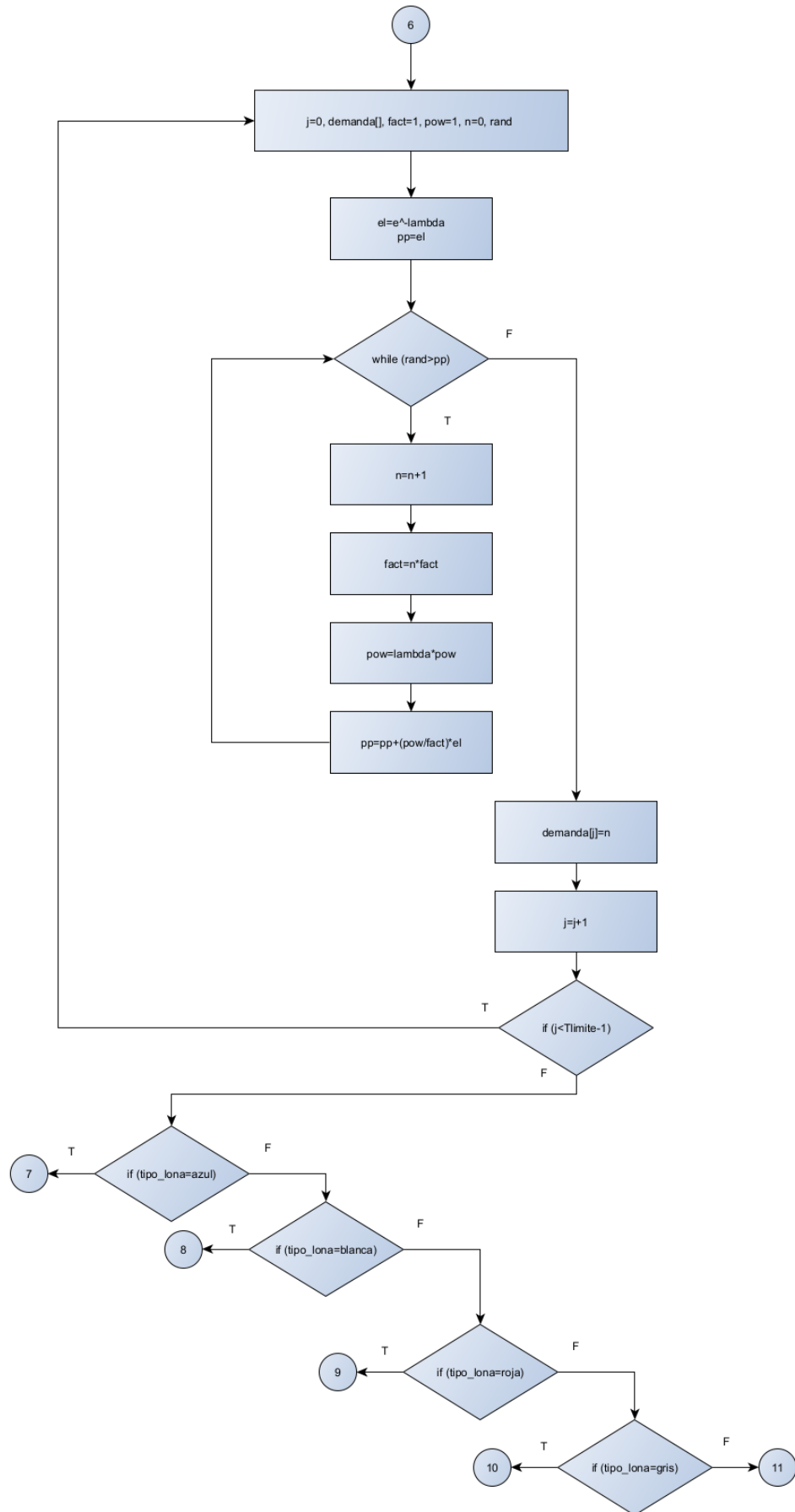












## TRASLACIÓN DEL MODELO

Para esta simulación utilizamos el lenguaje de programación Python, con una herramienta llamada Tkinter para su interfaz gráfica y un conjunto de librerías que nos ayudan para la toma de valores de nuestras variables aleatorias.

Primeramente, necesitamos importar nuestras librerías, junto a esto declaramos nuestras constantes para el programa:

```
1 from tkinter import *
2 from tkinter import ttk
3 from lib2to3.pgen2.pgen import generate_grammar
4 from matplotlib import pyplot as plt
5 from scipy import stats
6 import statistics
7 import numpy
8 import warnings
9 warnings.filterwarnings('ignore')
10 import random
11
12
13 from matplotlib.pyplot import text
14 #Constantes
15 OptionList = [
16     "Inserte dato",
17     "Lona Azul",
18     "Lona Blanca",
19     "Lona Gris",
20     "Lona Negra",
21     "Lona Roja"
22 ]
23 vrAzul = [7,3,2,4,0,7,2,3,1,1,1,1,3,2,3,1,1,1,3,4,3,4,4,1,2,5,4,4,3,3]
24 vrBlanco = [2,1,1,2,2,4,1,1,4,3,2,0,1,1,0,2,2,2,2,3,5,3,6,3,4,0,0,4,1,0]
25 vrGris = [0,0,1,0,0,0,0,0,1,3,0,2,3,0,0,1,3,1,0,1,1,1,0,2,2,2,1,1,1,1]
26 vrNegro = [2,0,0,0,0,0,1,1,0,0,0,0,1,0,0,0,1,0,1,1,2,0,0,0,2,4,1,2,0,1]
27 vrRojo = [0,0,1,1,0,0,0,0,0,1,0,1,1,4,0,0,0,0,0,0,0,0,1,1,2,2,2,1,2]
28 data = []
29
```

Figura 7.- Código 1

Posteriormente, con Tkinter iniciamos una ventana principal para insertar después nuestros inputs. También declaramos todas nuestras variables.

```
1 #Ventana principal
2 root = Tk()
3 root.geometry('350x230')
4 root.title('SIMULADOR MR. LONAS')
5 root.resizable(width=False, height=False)
6
7 #variables
8 datoLona = StringVar(root)
9 datoLona.set(OptionList[0])
10 datoDias = IntVar()
11 datoStock = IntVar()
```

Figura 8.- Código 2

```
1 frm = LabelFrame(root, text='inserta datos necesarios.', padx=30, pady=30)
2 frm.grid(row=0, column=0, columnspan=3, padx=20, pady=20)
3
4 ttk.Label(frm, text="Días a simular").grid(column=0, row=0)
5 ttk.Entry(frm, textvariable=datoDias).grid(column=2, row=0)
6
7 ttk.Label(frm, text="Tipo de lona").grid(column=0, row=1)
8 ttk.OptionMenu(frm, datoLona, *OptionList, ).grid(column=2, row=1)
9
10 ttk.Label(frm, text="Stock actual").grid(column=0, row=2)
11 ttk.Entry(frm, textvariable=datoStock).grid(column=2, row=2)
12
13 #Boton de aceptar
14 ttk.Button(frm, text=" ").grid(column=0, row=3)
15 ttk.Button(frm, text="Aceptar", command=ventanaNueva).grid(column=2, row=4, sticky= W + E)
16
17 root.mainloop()
```

Figura 9.- Código 3

Posteriormente, en esta ventana principal, llamada root, creamos un Label Frame, que tendrá dentro todos los inputs que necesitamos para la simulación. Entre esos inputs se encuentran: Los días a simular, el tipo de lona y el stock actual.

Después de esto se crea un botón que llama a una función que se encarga de hacer los procesos necesarios para la simulación.

```
1 def ventanaNueva():
2     lonasVendidas = 0
3     tiempoMuerto = 0
4     valorDias = datoDias.get()
5     valorStock = datoStock.get()
6     valorLona = datoLona.get()
7
8     stockRestante = valorStock
9
10    if(valorLona == "Lona Azul"):
11        for i in range(valorDias):
12            data.append(random.choice(vrAzul))
13
14        norm = stats.poisson.rvs(mu=statistics.mean(vrAzul), size=valorDias)
15        norm = numpy.ndarray.tolist(norm)
16
17        for i in range(valorDias):
18            if stockRestante < norm[i]:
19                tiempoMuerto += 1
20                stockRestante = stockRestante + valorStock
21            else:
22                stockRestante = stockRestante - norm[i]
23                stockRestante = stockRestante + valorStock
24                lonasVendidas = lonasVendidas + norm[i]
```

Figura 10.- Código 4

Dentro de esta función declaramos las lonas vendidas hasta ese momento (Siempre inicializa en 0) y el tiempo muerto, que hace referencia a los días en los que se tuvo que esperar al stock del día siguiente para cumplir con las necesidades de los clientes. Ambas variables son lonasVendidas y tiempoMuerto, respectivamente.

Posterior a eso, tomamos los valores de los inputs mediante la función get. Igualamos el valor restante al valor del stock que le asignamos en el input de la primera ventana o ventana principal.

Y entonces comparamos el valor obtenido en el input de “Tipo de lona” y dependiendo de la lona elegida (mediante un elif) se hace la generación de variables mediante su propia chi. En este primer caso podemos ver como el primer valor a poder elegir es “Lona azul” en donde en la variable norm se le asigna el chi mediante los datos obtenidos con anterioridad, y se crean una cantidad de variables de la misma longitud que la cantidad de días elegidos a simular mediante el input “Días a simular”.

Posteriormente se guardan todos estos valores en un vector para su manejo más sencillo.

Entonces, mediante ciclos for, se recorren los días simulados, con sus respectivas lonas simuladas mediante la variable norm, entonces, mediante un if, se da la condición en que si el stock en ese momento es menor a la cantidad de lonas pedidas simuladas mediante el array de norm se aumenta en 1 el tiempo muerto y se pase al siguiente día.

Pero si es que esta condición no se cumple (osease que si se tienen suficientes lonas para suplir el pedido de lonas del día) se reste la cantidad solicitada del array de norm de la variable stockRestante. Después se suma el stock del día siguiente y que se suma la cantidad de lonas venidas a la variable de lonasVendidas.

Figura 11.- Código 5

Posteriormente se crean dos ventanas. La primer Ventana es la ventana de los resultados de la simulación, en donde mediante un Label Frame imprimimos los siguientes datos:

- El total de días simulados.
- EL total de stock inicial.
- La lona que elegimos.
- Las lonas estimadas (que hace referencia a las lonas que fueron demandadas durante toda la simulación)
- Las lonas vendidas.
- El stock restante o sobrante.
- Tiempo muerto.

La segunda ventana despliega los siguientes datos:

- Los valores aleatorios generados (Osease la lista con todos los valores que se generaron durante la simulación)
- Cantidad de valores generados.
- Y el valor de lambda utilizado por el generador.

Después de esto se hace exactamente lo mismo con el resto de las variables de lonas disponibles (En este generador un

total de cinco contando con la ya explicada anteriormente).

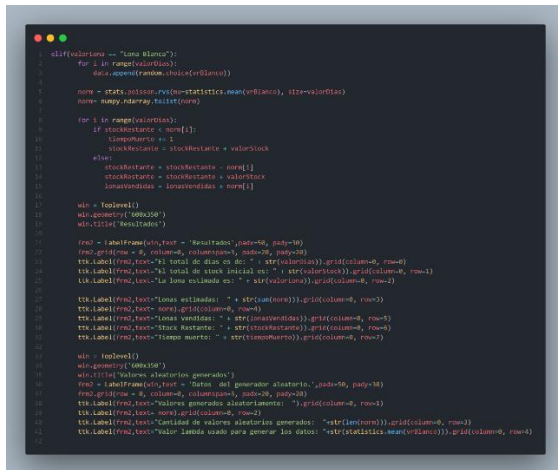


Figura 12.- Código 6

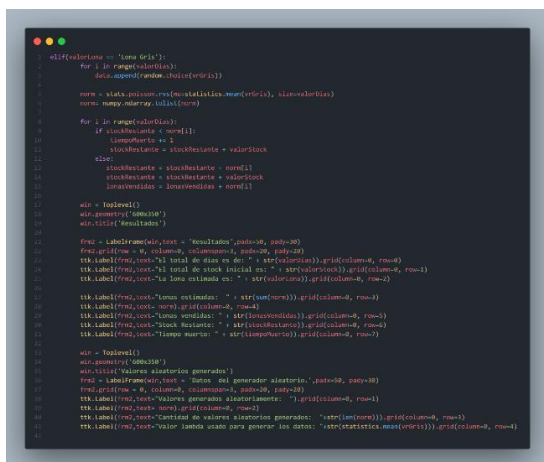


Figura 13.- Código 7

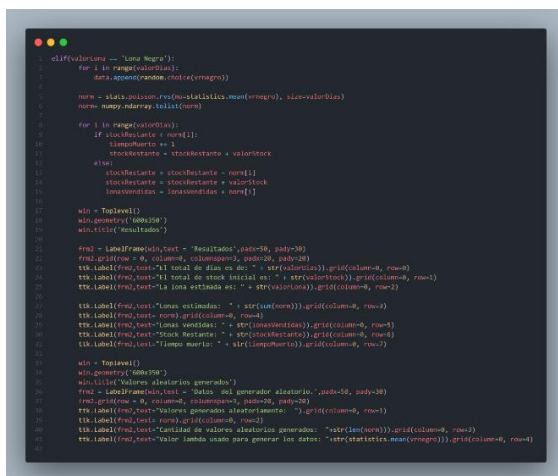


Figura 14.- Código 8

Por último, se agrega una condición en donde si no se elige ninguna lona simplemente despliegue una ventana con el aviso: ERROR, SELECCIONE UN TIPO DE LONA. Para así, en caso de que el usuario no haya seleccionado ninguna, no tenga problemas o confusiones a la hora de la simulación.



Figura 15.- Código 9

## SIMULACIÓN DEL MODELO

A continuación, se mostrará la simulación de nuestro modelo explicando las pantallas que se utilizan y se explicará la organización de estas.

### Pantalla de inicio

En la pantalla inicial, mostrada en la Figura 7 podemos encontrar una lista de elementos con los que el usuario puede interactuar para ingresar al código los valores de entrada.

Tenemos un cuadro de texto para ingresar la cantidad de días a simular, una lista desplegable para seleccionar el tipo de lona que se desea simular y un cuadro de texto para ingresar el stock actual que existe. Finalmente tenemos un botón de aceptar con el cual terminaremos de ingresar nuestros datos en el código para generar nuestros valores.

Inserta datos necesarios.

Dias a simular

Tipo de lona

Stock actual

Aceptar

Figura 16.- Ventana inicial

## Resultados

En esta ventana se mostrarán los distintos valores generados a partir de las entradas del usuario. Como podemos ver en la Figura 8, primero se muestran los valores que fueron ingresados por el usuario en la ventana anterior, el total de días usados, el total de stock inicial y el tipo de lona que fue seleccionada. Luego podemos encontrar los valores que fueron calculados por el generador, Lonas estimadas (El total de lonas que fueron pedidas en el intervalo de tiempo seleccionado), Lonas vendidas, Stock Restante y Tiempo muerto.

Resultados

El total de días es de: 30  
 El total de stock inicial es: 0  
 La lona estimada es: Lona Azul  
 Lonas estimadas: 92  
 1 4 2 6 4 7 5 1 4 2 2 0 2 2 6 2 3 2 3 4 2 2 4 4 2 1 5 2 5 3  
 Lonas vendidas: 0  
 Stock Restante: 0  
 Tiempo muerto: 29

Figura 17.- Ventana de resultados

## Valores aleatorios generados

Finalmente tenemos la ventana de valores aleatorios generados, mostrada en la Figura 9, donde podemos encontrar los valores que fueron generados de manera aleatoria en nuestro programa, primero tenemos los valores generados aleatoriamente, donde encontramos una lista de valores que fueron generados con la distribución del tipo de lona seleccionado, la cantidad de valores aleatorios generados y el valor de lambda usado para generar los datos, el cual también es obtenido del tipo de lona que fue seleccionado por el usuario en la pantalla inicial.

Valores aleatorios generados

Datos del generador aleatorio.

Valores generados aleatoriamente:  
 1 4 2 6 4 7 5 1 4 2 2 0 2 2 6 2 3 2 3 4 2 2 4 4 2 1 5 2 5 3  
 Cantidad de valores aleatorios generados: 30  
 Valor lambda usado para generar los datos: 2.7666666666666666

Figura 18.- Ventana de valores generados

## RESULTADOS

Para la experimentación se decidió establecer 30 días para simular y con un stock inicial de 2, el cual también serviría como nuestro valor de renovación diaria. Esto se repitió para todos los colores de lonas. Los resultados siguientes fueron los arrojados por la simulación desarrollada en Python.

Azul (stock inicial =2, 30 días)					
	Corrida 1	Corrida 2	Corrida 3	Corrida 4	Corrida 5
Lonas pedidas	86	94	80	84	89
Lonas vendidas	54	58	54	57	57
Stock restante	8	4	8	5	5
Tiempo muerto	7	8	4	5	6
Rojo (stock inicial =2, 30 días)					
	Corrida 1	Corrida 2	Corrida 3	Corrida 4	Corrida 5
Lonas pedidas	24	20	32	23	16
Lonas vendidas	24	20	32	23	16
Stock restante	38	42	30	39	46
Tiempo muerto	0	0	0	0	0
Gris (stock inicial =2, 30 días)					
	Corrida 1	Corrida 2	Corrida 3	Corrida 4	Corrida 5
Lonas pedidas	33	14	23	28	26
Lonas vendidas	33	14	23	28	26
Stock restante	29	48	39	34	36
Tiempo muerto	0	0	0	0	0
Blanco (stock inicial =2, 30 días)					
	Corrida 1	Corrida 2	Corrida 3	Corrida 4	Corrida 5
Lonas pedidas	83	72	82	76	67
Lonas vendidas	53	56	58	53	54
Stock restante	9	6	4	9	8
Tiempo muerto	6	4	6	5	3
Negro (stock inicial =2, 30 días)					
	Corrida 1	Corrida 2	Corrida 3	Corrida 4	Corrida 5
Lonas pedidas	17	23	21	13	17
Lonas vendidas	17	23	18	13	17
Stock restante	45	39	44	49	45
Tiempo muerto	0	0	1	0	0

Tabla 1. Resultados de 5 corridas.

## PRUEBAS DE LOS RESULTADOS

En los resultados obtenidos por el simulador podemos observar un gran parecido a los datos reales en el total de lonas que fueron pedidas. Conforma a esto podemos confirmar que las distribuciones son similares a los datos del generador con los datos reales. No fue necesario agregar tiempos de ejecución del programa, puesto que los resultados fueron datos de manera muy rápida.

Los tiempos muertos entre las distintas corridas no varían mucho, siendo los tiempos muertos más prominentes en las lonas blancas y azules, puesto a que son las más demandadas.

El primer error que tomamos en cuenta es que, en los resultados, al momento de que no se cumple con la demanda, pero hay stock disponible, la transacción no se lleva a cabo. Esto es un error, puesto que no se toma en cuenta que el negocio aún así puede vender lonas, pero no específicamente la demanda en caso de que el cliente así lo desee.

## ANÁLISIS

Tomando en cuenta las distintas corridas que fueron realizadas en el simulador del modelo, se logró observar que las lonas pedidas por los clientes en los lapsos de 30 días eran bastantes parecidos a los datos proporcionados por el sistema de ventas de los toldos. Esto confirma que la generación de las demandas del simulador produce resultados parecidos a la realidad. Tomando esto en cuenta concluimos que nuestra simulación es semejante a la realidad, puesto que los datos arrojados por el simulador son bastantes parecidos a los que podemos encontrar en el Anexo 1, por lo tanto, es funcional. Debido a que no se pudieron obtener los datos específicos de la renovación de los datos reales, entonces no podemos comparar lonas el stock restante, así como los tiempos muertos y pedidos no satisfechos, pero debido a que la generación de pedidos es parecida, entonces la simulación puede ser usada como herramienta para tener una aproximación a las demandas que serán



dadas por cada color de lona en una cantidad de días específica.

### **POSIBLES MODIFICACIONES DEL MODELO**

En base a las pruebas y el análisis de los resultados, consideramos que se puede mejorar el sistema haciendo lo siguiente:

- Obtener datos reales acerca de los tiempos muertos y pedidos de lonas no cumplidos.
- Tomar en cuenta que algunos clientes pueden comprar menos lonas si es que se tiene inventario de ellas, pero no se cumple con la demanda entera.
- Obtener más datos reales de ventas para poder así tener demandas específicas a una época del año y no solo específicamente a un mes. Esto debido a que la demanda de lonas puede ser distinta dependiendo de la época del año.

### **CONCLUSIONES**

Podemos concluir de la información presentada anteriormente y de los resultados obtenidos que hemos desarrollado la capacidad de analizar procesos de ensamble para no solo entender mejor un modelo de este tipo, sino también poder idear mejores maneras de hacerlo más eficiente mediante el uso de sistemas simulados, buscando la optimización y la mejora de los tiempos del sistema y haciendo una mejor toma de decisiones. También pudimos mejorar nuestro dominio de métodos estadísticos descriptivos aplicándolos a diferentes conjuntos de datos para observar su comportamiento e implementarlo en nuestro sistema para asegurar el desarrollo de un modelo de alta calidad.

### **BIBLIOGRAFÍA**

- Informes del sistema de venta en línea de “Mr. Toldos”
  - Venta de toldos por día.

### Anexo 1.- Datos de los días 30 de marzo al 28 de abril

Cantidad de lonas vendidas por color en cada día del mes.					
	Azul	Rojo	Gris	Blanco	Negro
Día 1	7	0	0	2	2
Día 2	3	0	0	1	0
Día 3	2	1	1	1	0
Día 4	4	1	0	2	0
Día 5	0	0	0	2	0
Día 6	7	0	0	4	0
Día 7	2	0	0	1	1
Día 8	3	0	0	1	1
Día 9	1	0	1	4	0
Día 10	1	1	3	3	0
Día 11	1	0	0	2	0
Día 12	1	1	2	0	0
Día 13	3	1	3	1	1
Día 14	2	4	0	1	0
Día 15	3	0	0	0	0
Día 16	1	0	1	2	0
Día 17	1	0	3	2	1
Día 18	1	0	1	2	0
Día 19	3	0	0	2	1
Día 20	4	0	1	3	1
Día 21	3	0	1	5	2
Día 22	4	0	1	3	0
Día 23	4	1	0	6	0
Día 24	1	1	2	3	0
Día 25	2	2	2	4	2
Día 26	5	2	2	6	4
Día 27	4	2	1	0	1
Día 28	4	2	1	4	2
Día 29	3	1	1	1	0
Día 30	3	2	1	6	1