**1er trimestre**

Pensamiento computacional

Desarrollo de aplicaciones teniendo en cuenta la estructura y funcionamiento de los dispositivos donde se tienen que ejecutar.

En cualquier lenguaje de programación existen tres estructuras básicas para crear apps:

1. Secuencia → tiene que ser ordenada
2. Selección (if) → T/F
3. Repetición (bucles) → debe tener salida

Una sentencia son instrucciones del lenguaje de programación.

En el condicional simple las sentencias solo se ejecutan si la condición es verdadera en caso contrario no hace nada.

Crear func 🡪 argumentos, llamarla 🡪 parámetros.

Una función es un conjunto de lineas de código que realizan una tarea determinada, a las que se le asigna un nombre

La resolución de un problema en un ordenador mediante un programa, requiere como paso previo, una especificación detallada del proceso que hay que realizar para alcanzar la solución.

Se denomina algoritmo a una descripción no ambigua y ordenada de las operaciones que hay que realizar para obtener la solución de un determinado problema en un tiempo finito.

Está demostrado matemáticamente que la solución de un problema se resuelve descomponiendo el problema en partes más pequeñas. La solución de cada una de las partes conduce a la solución del problema original.

Para depurar nuestro código, cuando no funciona, podemos añadir comentarios a las líneas sospechosas.

Validar el código significa comprobar que las entradas por pantalla son correctas y corresponden el tipo de dato que se pide.

**2do trimestre**

El aprendizaje automático (machine learning) consiste en que el ordenador aprenda a partir de un conjunto de datos y estadísticas, para poder predecir comportamientos futuros.

* Media: el valor medio de una serie de datos
* Mediana: el valor central de una serie de datos
* Moda: valor que más se repite

Los datos numéricos son números que se pueden dividir en 2 grupos: discretos, son números enteros; y continuos, son números que tienen valores infinitos, como por ejemplo el precio de un artículo o su tamaño.

Los datos categóricos son valores que no se pueden medir unos con otros, por ejemplo, el valor de un color.

Los datos ordinales son como los categóricos, pero se pueden comparar entre sí.

Conociendo el tipo de datos obtenidos de la fuente sabremos qué técnicas hay que utilizar para analizarlos.

La **desviación estándar** es un número que describe cómo de dispersa está la serie, una desviación estándar pequeña significa que todos los números están cerca de la media; sin embargo, una desviación estándar alta implica que la serie ocupa un alto rango de valores.

La **varianza** es otro número que nos indica la dispersión de la muestra. La desviación estándar es la raíz cuadrada de la varianza, es decir, el cuadrado de la desviación estándar es la varianza.

22 dic 2022

Histograma

Para visualizar el conjunto de datos dibujaremos un histograma, los métodos se encuentran en la librería “matplotlib.pyplot”. Después de crear el histograma (está guardado en memoria), hay que mostrarlo.

En un histograma, en el eje de abscisas o eje X, nos muestra la diversidad de valores de la lista (el rango); y en el eje de ordenadas o eje Y, nos muestra la cantidad que hay de cada valor de la lista.

plt.hist(x, 5)

plt.show()

La librería numpy está pensada para trabajar con grandes cantidades de números de forma rápida (las listas de numpy son más eficientes que las listas de python)

plt.plot(xCoordinates, yCoordinates)

**Scatter** es un gráfico de dispersión (una nube de puntos), para crearlo necesitas el mismo número de coordenadas x que de y.

plt.scatter(xCoordinate, yCoordinate)

**Etiquetas**

Para añadir etiquetas a los ejes, utilizaremos de la librería matplotlib, el método xlabel(“”), ylabel(“”). Primero se traza el gráfico, después se añaden las etiquetas y por último se muestra.

**Título**

Para añadir un título en la parte superior del gráfico utilizaremos el método plt.title(“”)

**Rejilla (grid)**

Utilizaremos el método grid()

Nota: un histograma es un gráfico que muestra la frecuencia de una distribución

**Gráficos de tarta (Pie chart)**

Para crear un gráfico de tarta solamente se necesita una lista y utilizar el método pie()

Nota: es importante utilizar el gráfico adecuado según los datos que queramos representar.

También podemos desplazar una porción para destacarla. Para desplazar las porciones tenemos que utilizar una lista de valores numéricos indicando el desplazamiento de cada porción. Para cambiar los colores de las porciones vamos a necesitar otra lista que indique el color de cada porción. Las cadenas se pueden expresar por el nombre del color, su abreviatura o su código en hexadecimal.

Los colores en hexadecimal se expresan a pares #RRGGBB

La leyenda es una indicación visual de qué representa cada color, suele aparecer a la derecha del gráfico. Para poder utilizar la función legend(), primero hay que haber definido las etiquetas.

**Regresión lineal**

El término regresión hace referencia a la relación que existe entre 2 variables. En machine learning y en los modelos estadísticos, esta relación se utiliza para predecir futuros acontecimientos (eventos). Es útil para saber la relación que existe entre dos variables.

El método que me genera la línea de regresión pertenece a la librería scipy. La ecuación general de una recta es y = ax + b, los valores de x se obtienen de la toma de datos, y la a y la b son dos constantes. La a es la pendiente y la b es la ordenada en el origen (intercept). Se trata de encontrar la ecuación de una receta que pase lo más cerca posible de todos  los puntos del gráfico de dispersión (lo ideal sería un espeto)

Los métodos en Python pueden devolver más de un valor

**Función map()**

Si queremos pasar como parámetro todos los elementos de una lista a una función, aplicando la función a cada uno de estos elementos utilizaremos: map(función, \*args). Los parámetros que van después del nombre de la función tienen que ser objetos iterables.

El resultado de map es un objeto almacenado en una posición de memoria en formato hexadecimal. “0x“ == hexadecimal

Nota: el nombre de las variables es un alias de una posición de memoria RAM.

El operador suma está sobrecargado, esto quiere decir que opera sobre distintos tipos de datos.

20 ene 2023

Los valores que obtenemos del método linregress(), los vamos a utilizar en otra función que será la ecuación de la recta.

def myfunc(x): …

La x son los valores que se obtienen en la recogida de datos.

A partir de los valores obtenidos por observación me proporciona la coordenada que me falta. La lista de valores por observación se corresponde con la coordenada x de cada punto.

Mi modelo va a ser una lista que tenga todas las coordenadas y.

La r, la p y el std\_error son valores que nos indican la corrección y la fiabilidad del modelo. La regresión lineal representa la relación que hay entre 2 variables. Se necesitan dos listas de datos (coordenadas del eje x e y) para representar un gráfico de dispersión. La función myfunc() se utiliza para calcular puntos de la recta, mymodel es una lista. La lista x son los valores que conocemos o de los que queremos conocer su valor y. El objeto que devuelve map() es una lista.

GUI (Graphic User Interface, interfaz gráfica de usuario) significa que podemos utilizar una aplicación fuera del entorno, del mismo estilo que un editor, un navegador, etc. Hay distintas librerías para programación gráfica con Python pero nosotros utilizaremos **tkinter** por su facilidad de manejo.

Una ventana es un objeto. Dentro de las ventanas situaremos objetos, a estos objetos se les denomina widgets. Cuando utilizamos el constructor de una clase a parte de las propiedades predefinidas, podemos añadir los valores que queramos. Para situar los widgets en la ventana se utiliza un “gestor de geometría”.

Las ventanas también son un widget y se dice que son el padre de los widget que están dentro. Para que el botón reaccione al evento de hacer click, debemos añadir código.

**Eventos**

 Un evento es cualquier acción desencadenada o por el usuario o por el SO. El código asociado al evento normalmente suele ser una función.

**Temas**

En programación gráfica se denomina tema al aspecto visual de los widgets. Para que los widgets tengan un aspecto más moderno utilizaremos Ttk (Themed tk)

Los widgets con tema deben estar dentro de un widget contenedor con tema. A la ventana principal le añadiremos un marco.

Un widget es tan grande como su contenido.

label[“text”] = “Hello”

label.config(text = “Hello”)

label.configure(text = “Hello”)

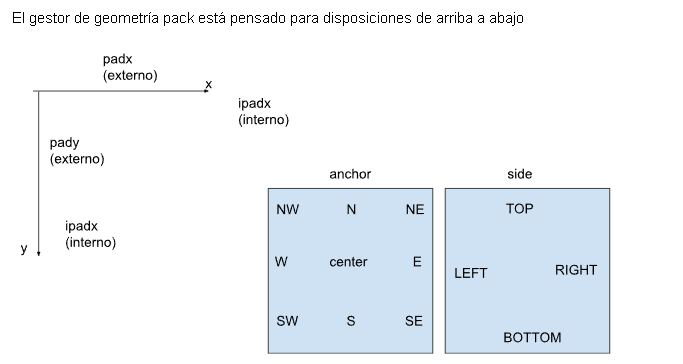
Para que una función sea callback su nombre debe ser el valor del parámetro “command” en la creación del objeto. El nombre de la función callback nunca se escribe entre comillas.

**Gestor de geometría pack**

Con el parámetro side indicamos en qué parte de la ventana se posiciona el widget. Los valores de fill son siempre cadenas.

Se puede cambiar el aspecto del widget (las etiquetas) para que tengan relieve, utilizando el parámetro relief.

Si ya hay un widget se posiciona a su derecha. Con este gestor de geometría hay que tener presente el orden en el que situamos los widgets.



Gestor de geometría **Grid**

Cada fila y columna en el grid se identifica por un índice (columna, fila). Las filas no tienen porqué empezar por 0. Las columnas pueden tener huecos (gaps), no tienen porqué ir seguidas. En la intersección de una fila y una columna tenemos una celda, que es donde se van a situar los widgets.

El ancho de una columna depende del ancho del widget que contenga, de igual forma, el alto de una fila depende del alto del widget que contenga. Las filas y las columnas se pueden expandir (span).

Antes de posicionar los widgets, hay que configurar el grid, siempre va en un widget contenedor.

container.columnconfigure(índice, peso) | container.rowconfigure(índice, peso)

El peso indica el valor relativo del tamaño de una columna o fila con respecto de las demás.

Sticky es una propiedad de cómo se sitúa el widget en la celda (parámetros iguales a anchor).

Frame widget

Es un widget que se muestra como un rectángulo simple, su uso típico es organizar otros widgets. Utilizaremos la clase ttk.Frame(). Para mostrar un borde utilizaremos la propiedad borderwidth (por defecto es 0). Otras propiedades son width y height.

Las propiedades de los widgets se pueden cambiar de la siguiente forma: frame[“padding”] = (left, top, right, bottom)

StringVar() es una clase que crea objetos de texto

Cuando creamos una ventana nueva, tenemos que decidir si vamos a utilizar estilos o no. Si vamos a utilizar estilos hay que añadir siempre un frame, la ventana principal puede tener varios frames.

Los objetos StringVar() comunican siempre que haya un cambio en su valor. Esta característica es útil si queremos actualizar automáticamente otro widget basándonos en el valor del objeto stringVar()

string\_var = tk.StringVar()

string\_var.trace(mode, callback) - mode {“w”,  “r”, “u”}

mode:

“w” : se dispara cuando se cambia el contenido.

“r”: se dispara cuando se lee el contenido

“u”: se dispara cuando se elimina la variable

**Estilos**

Un tema es una colección de estilos que determina la apariencia de un widget. Un estilo es la descripción de la apariencia de una clase de widget. Un tema ya viene configurado con una serie de estilos predeterminados. Los estilos hay que definirlos antes de utilizarlos

Para poder aplicar estilos, se escribe la T mayúscula delante del nombre de la clase y podemos preceder el estilo con su nombre

Con el operador \* en los argumentos de una función indicamos que se le van a pasar un número de parámetros variable