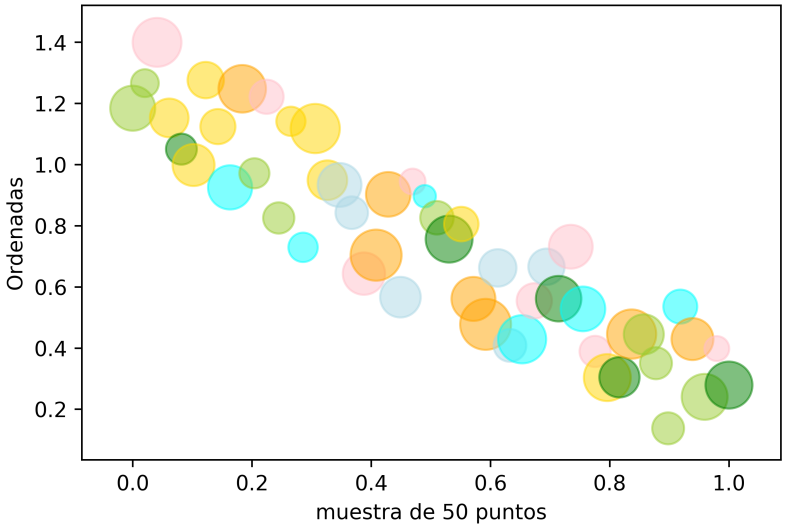
Librería matplotlib. Ejercicios resueltos

1. Nube de puntos en torno a una recta

Se pide la generación de una nube de N = 50 puntos aleatorios, en torno a la recta y = 1 - x. Digamos por ejemplo que se crean con una dispersión vertical extraída de [-0.5, 0.5] uniformemente. Los puntos deben generarse con tamaños y colores aleatorios.



Nube de puntos en torno a una recta. Solución

Las abcisas de los puntos se han obtenido dividiendo el intervalo $[-1, 1]$ en N intervalos fijos. las ordenadas se han generado aleatoriamente, añadiendo a caad punto de la recta una cantidad verticalmente aleatoria de $[-1, 1]$, que se ha generado uniformemente.

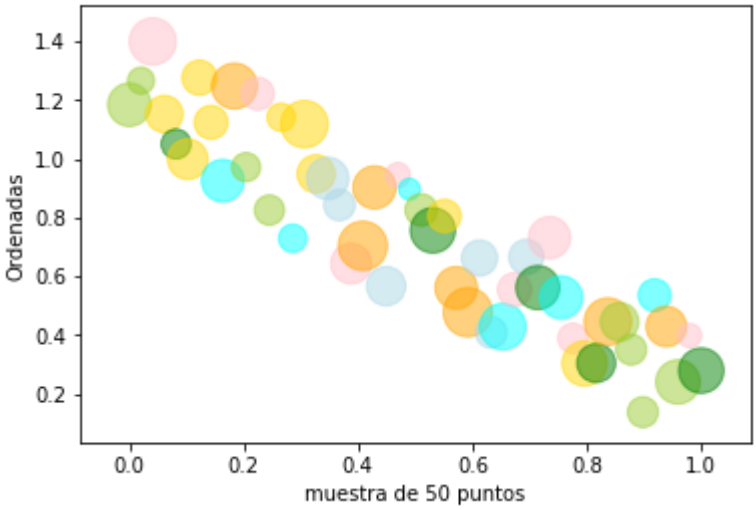
Aunque la estética (colores y tamaños) es lo de menos, se ha procurado atender con un grado de transparencia del 50% y tamaños aleatorios, ajustados a gusto del diseñador, yo en este caso.

El resto es el uso de la instrucción `matplotlib.pyplot.scatter`.

Nota: La instrucción `matplotlib.pyplot.savefig` se ha de emplear antes de `matplotlib.pyplot.show` necesariamente.

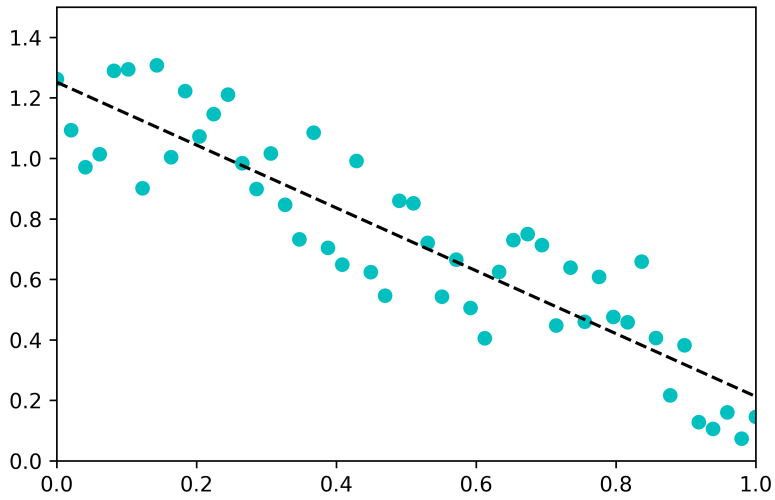
In [1]:

```
import numpy as np
import matplotlib.pyplot as plt

N = 50                                  # Tamanno muestral
x = np.linspace(0, 1, N)                # abcisas
y = 1 - x  + 0.5 * np.random.rand(N)    # ordenadas
# Colores, entre una lista dada:
colores  = np.random.choice(["pink", "cyan", "yellowgreen", "lightblue", "gold", "orange",
tamannos = 100 + 500 * np.random.rand(N)  # tamaños

plt.scatter(x, y, c=colores, s=tamannos, alpha = 0.5)
plt.xlabel("muestra de " + str(N) + ' puntos')
plt.ylabel('Ordenadas')
plt.savefig('./figuras/fig-ejercicio-1.png', dpi=600)
plt.show()
```



2. Recta de regresión

Deseamos ahora ver la recta de regresión que aproxima la nube de puntos generados en el ejercicio anterior. Esta vez cambiamos la apariencia, porque los colores y tamaños realmente no afectan a la recoa de regresión.



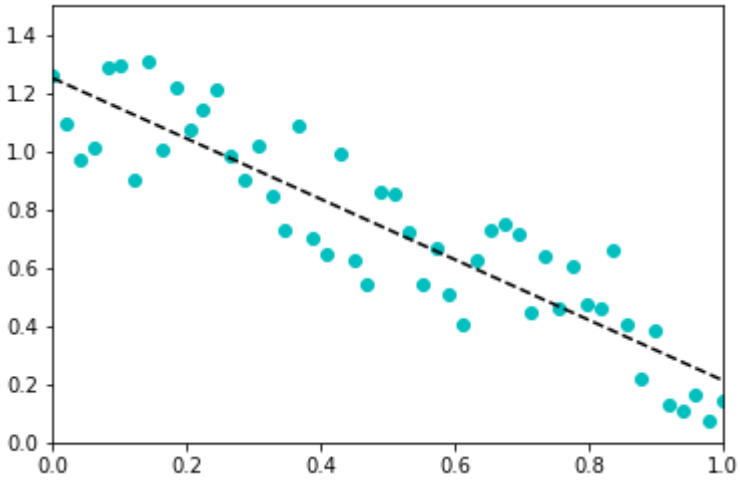
2. Recta de regresión. Solución

La generación de puntos es igual que en el ejercicio anterior. Se ha repetido únicamente para mantener la solución independiente del código de la anterior.

La estética se simplifica, omitiéndose colores y tamaños.

In [2]:

```
import numpy as np
import matplotlib.pyplot as plt

N = 50                                  # Tamanno muestral
x = np.linspace(0, 1, N)                # abcisas
y = 1 - x  + 0.5 * np.random.rand(N)    # ordenadas

fit = np.polyfit(x, y, 1)
fit_fn = np.poly1d(fit)
plt.plot(x, y, 'co', x, fit_fn(x), '--k')
plt.xlim(0, 1)
plt.ylim(0, 1.5)
plt.savefig('./figuras/fig-ejercicio-2.png', dpi=600)
plt.show()
```

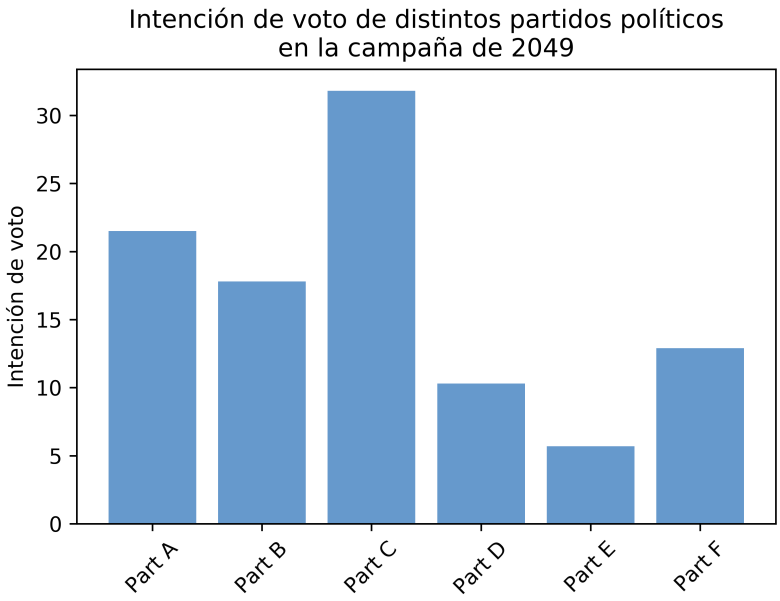


3. Diagrama de barras

Tras un sondeo a pocos días de unas elecciones, se ha registrado la intención de voto que sigue, donde se han camuflado los nombres de los partidos políticos para no influir en tu opinión personal:

```
partidos = ['Part A', 'Part B', 'Part C', 'Part D', 'Part E', 'Part F']
intencion_voto = [21.5, 17.8, 31.8, 10.3, 5.7, 12.9]
```

Se pide mostrar estos datos en un diagrama de barras como el que sigue:



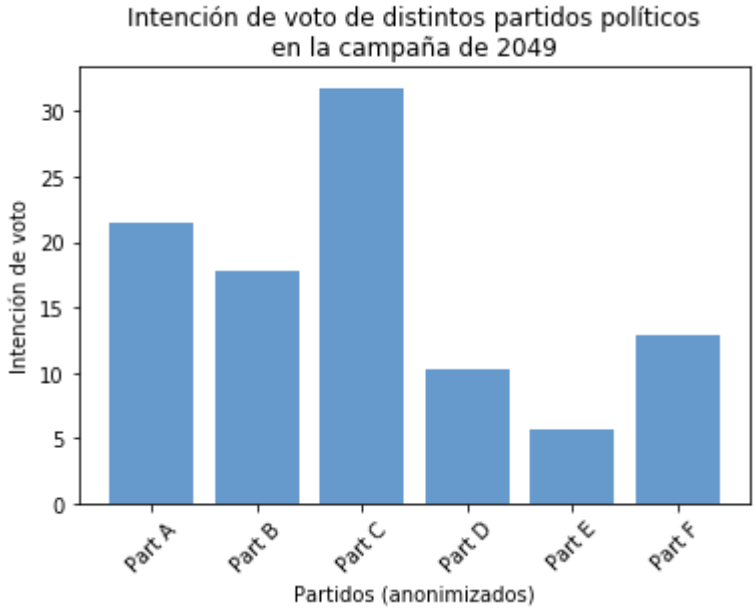
3. Diagrama de barras. Solución

Los datos para este programa son los datos en el enunciado. La generación de puntos es igual que en el ejercicio anterior. Se ha repetido únicamente para mantener la solución independiente del código de la anterior.

La estética se simplifica, omitiéndose colores y tamaños. Se han inclinado los nombres de los partidos políticos 45º, para separarlos entre sí, y facilitar su lectura.

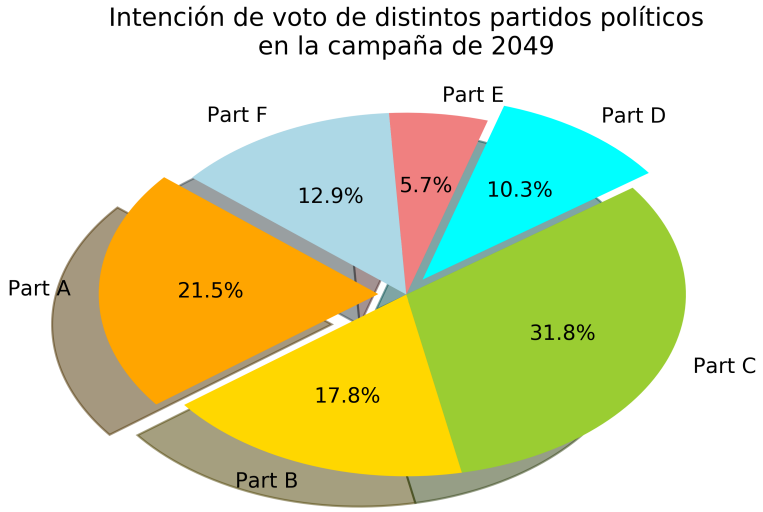
In [3]:

```
import matplotlib.pyplot as plt

partidos = ['Part A', 'Part B', 'Part C', 'Part D', 'Part E', 'Part F']
intencion_voto = [21.5, 17.8, 31.8, 10.3, 5.7, 12.9]
x_pos = [i for i, _ in enumerate(partidos)]
plt.bar(x_pos, intencion_voto, color=(0.4, 0.6, 0.8, 1.0))
plt.xlabel("Partidos (anonimizados)")
plt.ylabel("Intención de voto")
plt.title("Intención de voto de distintos partidos políticos\n" + "en la campaña de 2049")
# Rotación de los nombres de las abcisas:
plt.xticks(x_pos, partidos, rotation=45)
plt.savefig('./figuras/fig-ejercicio-3.png', dpi=600)
plt.show()
```



4. Diagrama de sectores

Para el sondeo anterior, deseamos ahora ver los datos recogidos en un diagrama de sectores, con los datos de los partidos A y D detacados.



3. Diagrama de sectores. Solución

Los datos para este programa son los datos en el enunciado anterior. Se han repetido en la solución para que ésta sea completa, independiente del código de la anterior.

In [4]:

```
import matplotlib.pyplot as plt

partidos = ['Part A', 'Part B', 'Part C', 'Part D', 'Part E', 'Part F']
intencion_voto = [21.5, 17.8, 31.8, 10.3, 5.7, 12.9]
x_pos = [i for i, _ in enumerate(partidos)]
colores = ['orange', 'gold', 'yellowgreen', 'cyan', 'lightcoral', 'lightblue']

# Sacar un poco el primer y el cuarto partido:
explode = (0.1, 0, 0, 0.1, 0, 0)

# Plot
plt.pie(intencion_voto, explode=explode, labels=partidos, colors=colores, autopct='%1.1f%%'
plt.title("Intención de voto de distintos partidos políticos\n" + "en la campaña de 2049")
plt.savefig('./figuras/fig-ejercicio-4.png', dpi=600)
plt.show()
```