In [5]:
```
print("Hello world on jupyter");
```

```
Hello world on jupyter
```

In [7]:
```
print(9*3)
```

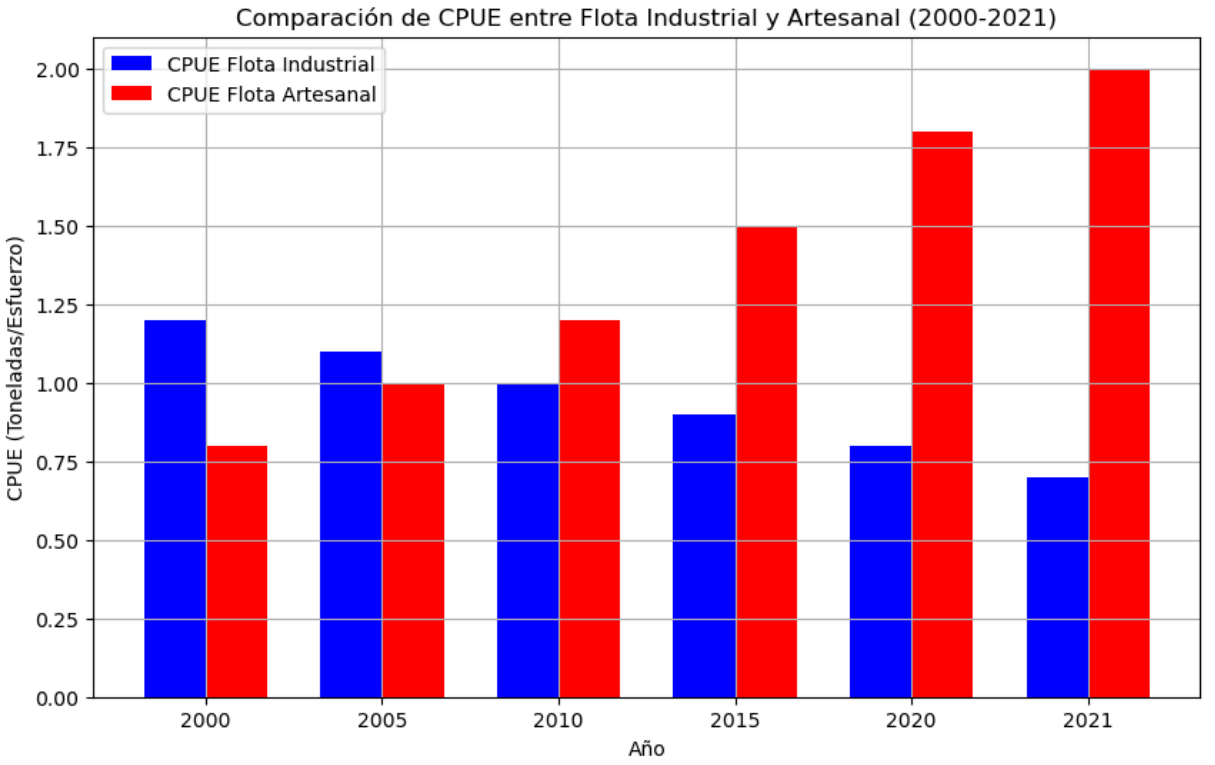
```
27
```

In [9]:
```
import math


def areaCirculo(r):
    area = math.pi * r **2
    return area



radioUsuario = 0

while radioUsuario <=0:
 radioUsuario = float(input("Por favor ingrese el radio del circulo: "))


resultadoUsuario = (areaCirculo(radioUsuario))


print("\n")
print("*"*80)

print(f"El area del circulo es = {resultadoUsuario:.2f}")
```

```
********************************************************************************
****
El area del circulo es = 201.06
```

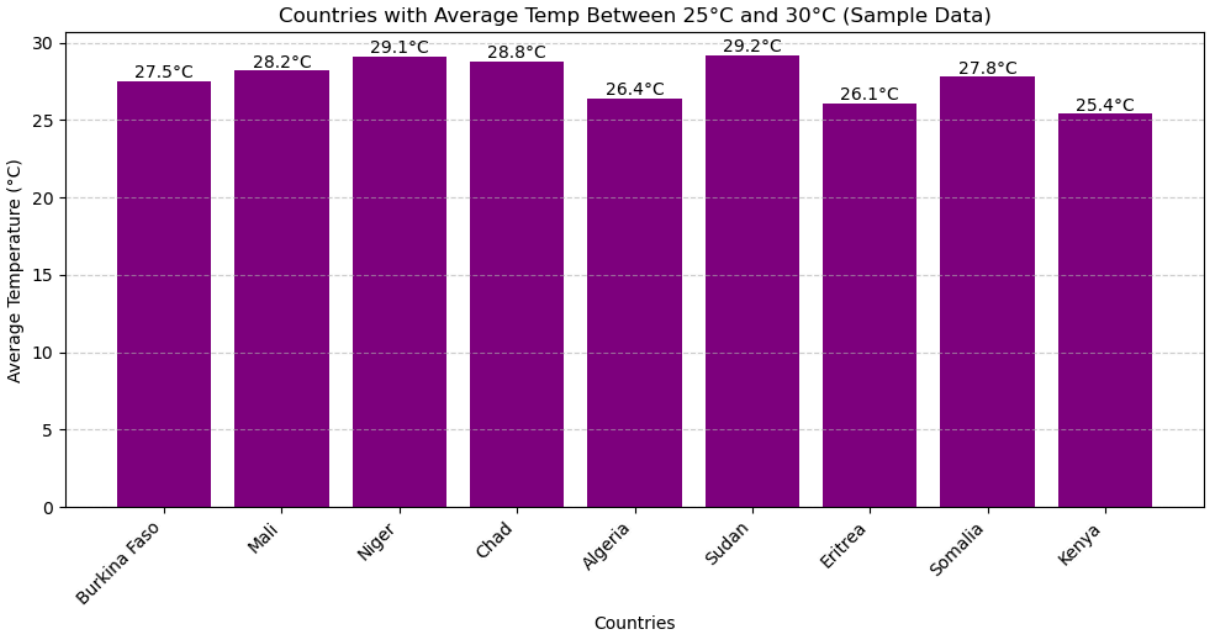
In [11]:
```
import matplotlib.pyplot as plt
import pandas as pd

# Datos
data = {
    'Año': [2000, 2005, 2010, 2015, 2020, 2021],
    'CPUE Flota Industrial': [1.2, 1.1, 1.0, 0.9, 0.8, 0.7],
    'CPUE Flota Artesanal': [0.8, 1.0, 1.2, 1.5, 1.8, 2.0]
}

# Convertir los datos en un DataFrame
df = pd.DataFrame(data)

# Crear el gráfico de barras agrupadas
fig, ax = plt.subplots(figsize=(10, 6))
bar_width = 0.35

# Posiciones de las barras
bar1 = range(len(df['Año']))
bar2 = [x + bar_width for x in bar1]
```

```
# Crear las barras
plt.bar(bar1, df['CPUE Flota Industrial'], color='b', width=bar_width, label
plt.bar(bar2, df['CPUE Flota Artesanal'], color='r', width=bar_width, label=

# Añadir títulos y etiquetas
plt.xlabel('Año')
plt.ylabel('CPUE (Toneladas/Esfuerzo)')
plt.title('Comparación de CPUE entre Flota Industrial y Artesanal (2000-2021
plt.xticks([r + bar_width/2 for r in range(len(df['Año']))], df['Año'])
plt.legend()
plt.grid(True)

# Mostrar el gráfico
plt.show()
```

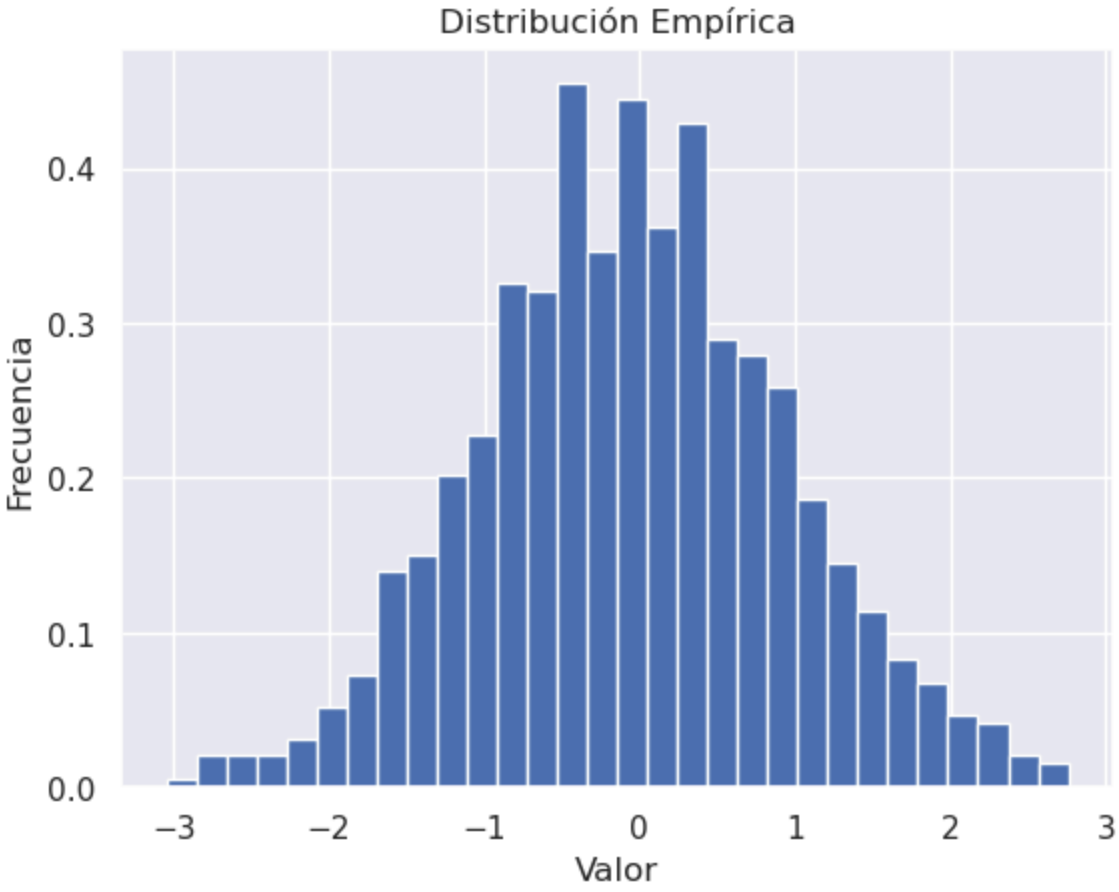


In [13]:
```
import matplotlib.pyplot as plt

def get_country_temps(data_source="sample"):  # Optional: Specify data sourc
  if data_source == "sample":
    countries = [
        "Burkina Faso", "Mali", "Niger", "Chad", "Algeria", "Ethiopia",
        "Sudan", "Eritrea", "Djibouti", "Somalia", "Kenya", "Uganda",
        # ... (more countries)
    ]
    # Sample average temperatures (replace with actual data)
    average_temps = [27.5, 28.2, 29.1, 28.8, 26.4, 23.4, 29.2, 26.1, 31.2, 2
                     # ... (more temperatures)
    ]
  else:
    # Implement code to fetch data from your chosen source
    # ...
```

```
    pass
  return countries, average_temps

try:
  # Get country and temperature data
  countries, average_temps = get_country_temps()

  # Filter countries and temperatures between 25 and 30°C
  filtered_countries = [country for country, temp in zip(countries, average_
  filtered_temps = [temp for temp in average_temps if 25 <= temp < 30]

  # Check if there are any countries within the temperature range
  if not filtered_countries:
    print("No countries found with average temperatures between 25°C and 30°
    exit()

  # Create the plot
  plt.figure(figsize=(10, 6))  # Adjust figure size for potentially fewer co

  # Create bars and add labels with loop
  bars = plt.bar(filtered_countries, filtered_temps, color='purple')  # Purp
  for bar, temp in zip(bars, filtered_temps):
    plt.text(bar.get_x() + bar.get_width() / 2, temp + 0.2, f"{temp:.1f}°C",

  plt.xlabel("Countries")
  plt.ylabel("Average Temperature (°C)")
  plt.title("Countries with Average Temp Between 25°C and 30°C (Sample Data)

  # Customize the plot (optional)
  plt.xticks(rotation=45, ha="right")  # Rotate x-axis labels for readabilit
  plt.grid(axis='y', linestyle='--', alpha=0.6)

  # Display the plot
  plt.tight_layout()
  plt.subplots_adjust(bottom=0.3)  # Adjust space for x-axis labels

  plt.show()
except Exception as e:
  print(f"An error occurred: {e}")
```

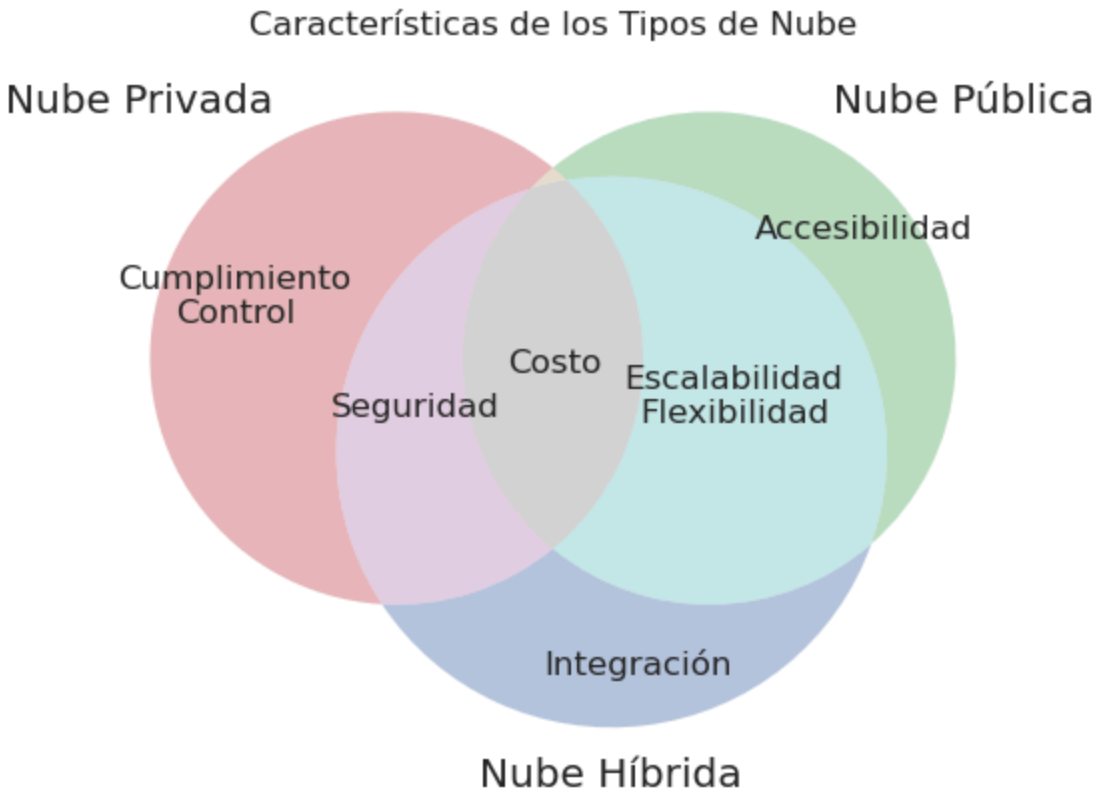


In [15]:
```python
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Generamos una muestra de datos aleatorios
np.random.seed(0)
datos = np.random.normal(loc=0, scale=1, size=1000)

# Creamos el histograma de la distribución empírica
sns.set()
plt.hist(datos, bins=30, density=True)

# Agregamos título y etiquetas
plt.title('Distribución Empírica')
plt.xlabel('Valor')
plt.ylabel('Frecuencia')

# Mostramos el gráfico
plt.show()
```



In [17]:

```
import matplotlib.pyplot as plt
from matplotlib_venn import venn3

# Define las características de cada tipo de nube
nube_privada = {'Seguridad', 'Control', 'Costo', 'Cumplimiento'}
nube_publica = {'Escalabilidad', 'Flexibilidad', 'Costo', 'Accesibilidad'}
nube_hibrida = {'Seguridad', 'Escalabilidad', 'Flexibilidad', 'Costo', 'Inte

# Crea el diagrama de Venn
venn = venn3([nube_privada, nube_publica, nube_hibrida],
             ('Nube Privada', 'Nube Pública', 'Nube Híbrida'))

# Añade etiquetas a cada subconjunto
venn.get_label_by_id('100').set_text('\n'.join(nube_privada - nube_publica -
venn.get_label_by_id('010').set_text('\n'.join(nube_publica - nube_privada -
venn.get_label_by_id('001').set_text('\n'.join(nube_hibrida - nube_privada -
venn.get_label_by_id('110').set_text('\n'.join((nube_privada & nube_publica)
venn.get_label_by_id('101').set_text('\n'.join((nube_privada & nube_hibrida)
venn.get_label_by_id('011').set_text('\n'.join((nube_publica & nube_hibrida)
venn.get_label_by_id('111').set_text('\n'.join(nube_privada & nube_publica &

# Añade un título
plt.title('Características de los Tipos de Nube')

# Muestra el diagrama
plt.show()
```


Características de los Tipos de Nube
Nube Privada
Nube Pública
Accesibilidad
Cumplimiento
Control
Costo
Escalabilidad
Flexibilidad
Seguridad
Integración
Nube Híbrida

In [ ]: