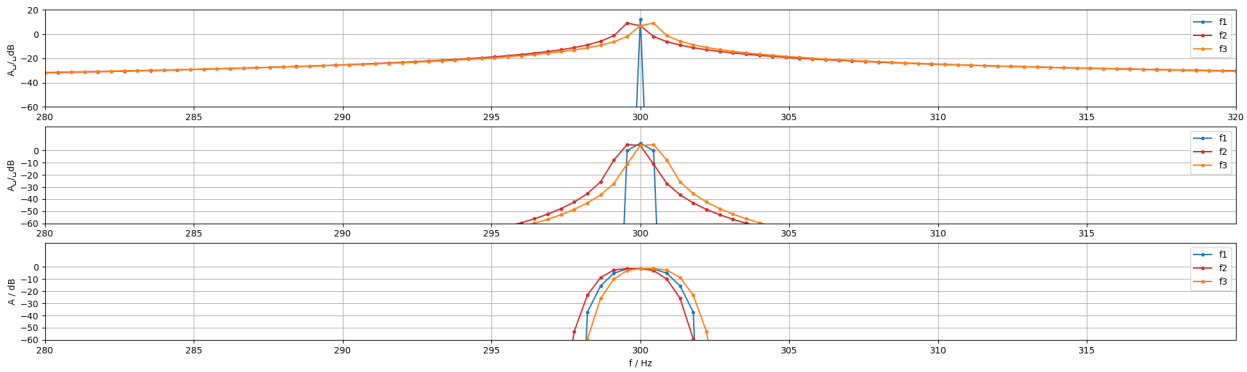
```
import numpy as np
import matplotlib.pyplot as plt
from numpy.fft import fft, ifft, fftshift
from scipy.signal.windows import hann, flattop


f1, f2, f3 = 500, 500.25, 499.75
N = 1800
k = np.arange(N)
fs = 800
A = 4 # Amplitude
x1 = A * np.sin(2 * np.pi * f1/fs * k)
x2 = A * np.sin(2 * np.pi * f2/fs * k)
x3 = A * np.sin(2 * np.pi * f3/fs * k)

wrect = np.ones(N)
whann = hann(N, sym=False)
wflattop = flattop(N, sym=False)

X1wrect = fft(x1)
X2wrect = fft(x2)
X3wrect = fft(x3)

X1whann = fft(x1 * whann)
X2whann = fft(x2 * whann)
X3whann = fft(x3 * whann)

X1wflattop = fft(x1 * wflattop)
X2wflattop = fft(x2 * wflattop)
X3wflattop = fft(x3 * wflattop)

def fft2db(X):
    N = X.size
    Xtmp = 2/N * X
    Xtmp[0] = Xtmp[0] / 2
    if N % 2 == 0:
        Xtmp[N//2] = Xtmp[N//2] / 2
    return 20 * np.log10(np.abs(Xtmp))

df = fs/N
f = np.arange(N) * df

plt.figure(figsize=(16*1.5, 10/1.5))
plt.subplot(3, 1, 1)
plt.plot( f , fft2db( X1wrect ) , 'C0o-' , ms=3, label='f1')
plt.plot(f , fft2db( X2wrect ) , 'C3o-' , ms=3, label='f2' )
plt.plot(f , fft2db( X3wrect ) , 'C1o-' , ms=3, label='f3' )
plt.ylim (-60 , 20 )
plt.xlim (280 , 320 )
plt.legend()
```

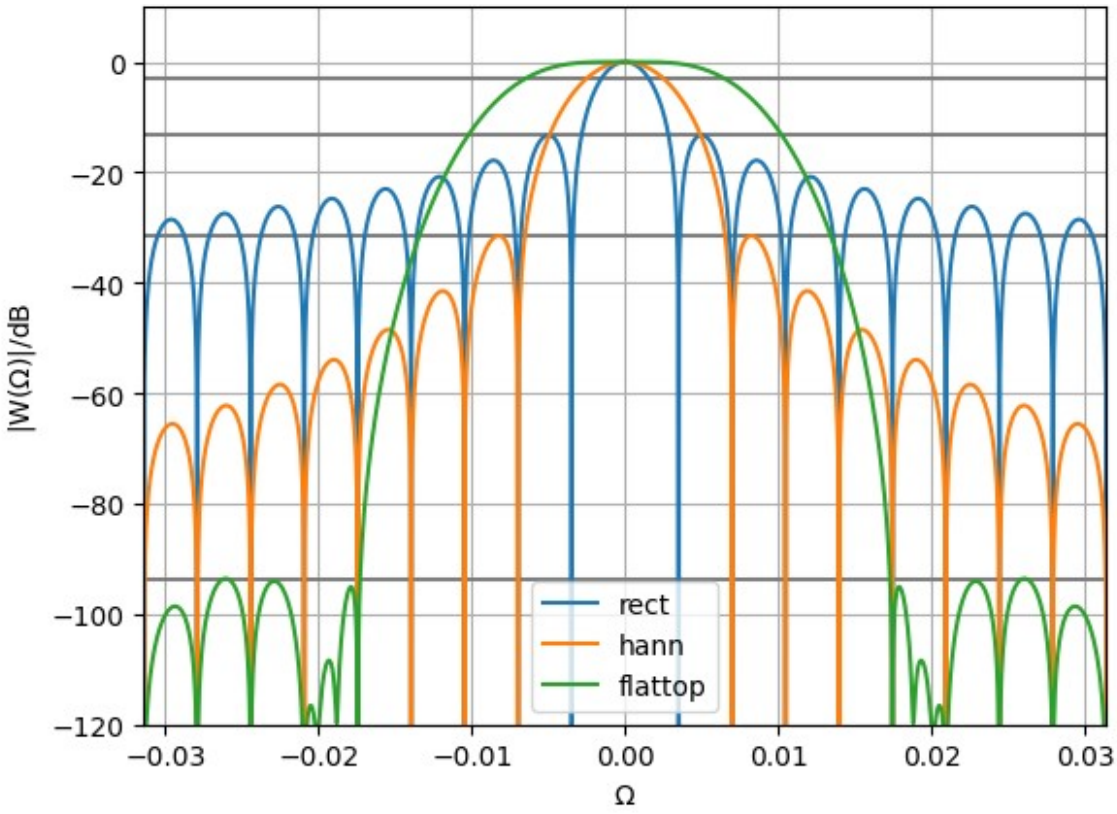
```
plt.ylabel( 'A␣/␣dB ' )
plt.grid( True )

plt.subplot( 3 , 1 , 2 )
plt.plot( f , fft2db(X1whann ) , 'C0o-' , ms=3, label='f1' )
plt.plot( f , fft2db(X2whann ) , 'C3o-' , ms=3, label='f2' )
plt.plot( f , fft2db(X3whann ) , 'C1o-' , ms=3, label='f3' )
plt.ylim (-60 , 20 )
plt.xlim(280, 320)
plt.xticks(np.arange(280, 320, 5))
plt.yticks(np.arange(-60, 10, 10))
plt.legend( )
plt.ylabel( 'A␣/␣dB' )
plt.grid( True )

plt.subplot(3, 1, 3)
plt.plot(f, fft2db(X1wflattop), 'C0o-', ms=3, label='f1')
plt.plot(f, fft2db(X2wflattop), 'C3o-', ms=3, label='f2')
plt.plot(f, fft2db(X3wflattop), 'C1o-', ms=3, label='f3')
plt.ylim(-60, 20)
plt.xlim(280, 320)
plt.xticks(np.arange(280, 320, 5))
plt.yticks(np.arange(-60, 10, 10))
plt.legend()
plt.xlabel('f / Hz')
plt.ylabel('A / dB')
plt.grid(True)
```




```
def winDTFTdB(w):
    N = w.size
    Nz = 100*N
    W = np.zeros(Nz, dtype=complex)
    W[0:N] = w
    W = np.abs(fftshift(fft(W)))
    W /= np.max(W)
    W = 20 * np.log10(W)
    Omega = 2*np.pi/Nz * np.arange(Nz) - np.pi
```

```
    return Omega, W

plt.plot([-np.pi, +np.pi], [-3.01, -3.01], 'gray')  # mainlobe
bandwidth
plt.plot([-np.pi, +np.pi], [-13.3, -13.3], 'gray')  # rect max
sidelobe
plt.plot([-np.pi, +np.pi], [-31.5, -31.5], 'gray')  # hann max
sidelobe
plt.plot([-np.pi, +np.pi], [-93.6, -93.6], 'gray')  # flattop max
sidelobe

Omega, W = winDTFTdB(wrect)
plt.plot(Omega, W, label='rect')
Omega, W = winDTFTdB(whann)
plt.plot(Omega, W, label='hann')
Omega, W = winDTFTdB(wflattop)
plt.plot(Omega, W, label='flattop')

plt.xlim(-np.pi, np.pi)
plt.ylim(-120, 10)
plt.xlim(-np.pi/100, np.pi/100)
plt.xlabel(r'$\Omega$')
plt.ylabel(r'|W($\Omega$)|/dB')
plt.legend()
plt.grid(True)
```

```
C:\Users\rafal\AppData\Local\Temp\ipykernel_13740\2564652716.py:8:
RuntimeWarning: divide by zero encountered in log10
  W = 20 * np.log10(W)
```


0
−20
−40
−60
−80
−100
−120
|W(Ω)|/dB
−0.03
−0.02
−0.01
0.00
0.01
0.02
0.03
Ω
rect
hann
flattop