In the continuous limit, the 1-D Fourier Transform is

$$F(k) = \int f(x) \exp(-ikx) dx$$

. When we switch to a 2-d Fourier transform, x and k simply turn into vectors, and the kx turns into $\vec{k} \cdot \vec{x}$

$$F(\vec{k}) = \int \int f(\vec{x}) \exp(-i\vec{k} \cdot \vec{x}) d^2\vec{x}$$

For a 1-d discrete FT, recall that we want $k = 1$ to correspond to one wave across our domain (we're assuming it goes from $0 <= x < x_{max}$). So to get a 1-D DFT, the integral turns into a sum, and we have to scale the exponent properly:

$$F(k_{j'}) = \sum f(x_j) \exp(-2\pi i k_{j'} x_j / x_{max})$$

In 2-dimensions, we just do the same thing, doubling up the indices, and doing the dot product in the exponent. For simplicity of notation, let x be the variable along 1 dimension, and y along the other, with conjugate variables k and l. So, we have that the 2-D DFT=

$$F(k, l) = \sum \sum f(x, y) \exp[-2\pi i(kx/x_{max} + ly/y_{max})]$$

Now, we can break up the exponent, and pick the order in which we sum:

$$F(k, l) = \sum_y \exp(-2\pi i l y / y_{max}) \sum_x f(x, y) \exp(-2\pi i k x / x_{max})))$$

Now look at the second term - for a fixed value of y, that is just the DFT along the x-direction, which requires $n_x \log(n_x)$ work to calculate if we do an FFT. We have to do this for each value of y, so the total amount of work required is $n_y n_x \log(n_x)$. This leaves us with just the DFT along the y direction now. A single one would again just be $n_y \log n_y$ work, but we have to do one for *each* value of k. We have n_x values of k, so the total work for the y FFT is $n_x n_y \log n_y$. Since these two steps can happen independently, the total combined work is just the sum of them, or

$$n_x n_y \log(n_x) + n_x n_y \log(n_y) = n_x n_y \log(n_x n_y) = n_{tot} \log(n_{tot})$$

where $n_{tot} = n_x n_y$ is the total number of data points in our discretized function. So, the 2-D FFT takes as much work to carry out as a 1-D FFT with the same total number of data points would have taken.