Requirements:

* FIR Filtering Module:
  + Functional:
    - Apply Filter to original image and calculate resulting image
    - Do not save resulting image to disk (?)
    - Pass filter coefficients as std::vector<float>
    - Filter coefficients must not be changed during calculation
    - Must use n Threads, n >= 1, each thread executes own calculation
  + Structural:
    - Filter API to setup filter
    - Workerthread class with n instantiations
    - Structure to save filter coefficients
    - Check for valid coefficients, valid image, valid number of threads
* Testapplication:
  + Functional:
    - Pass filter coefficients to filtering module
    - Set number of execution threads of filtering module
    - Open / pass image file to library
    - Measure “filtered images per second”
    - Start / stop filter library?
  + Structural:
    - User input coefficients?
    - User input image file?
    - User input number of threads?
* General:
  + Only use open source / free available software
  + Document instructions to build the tools
  + Test the program with different inputs
  + Use Windows, Linux, or both
  + Target x86 or x86\_64
  + If Windows, use native compilation, not clr