Setting up X11

Additionally, A4 Q2 requires you to set up a graphical display. To test this on the student server, you will need to set up X-Forwarding. Below are some instructions for setting up X-Forwarding. We suggest you get the `graphicsDemo` provided to run as early as possible so if you are unable to get X-Forwarding set up you have time to seek help in office hours. If you are using Linux you should be fine. If making an SSH connection to a campus machine, be sure to pass the **-Y** option and turn on **X-forwarding**. Windows Users:

- If you are using Windows and PuTTY, you need to download and run an X server such as **XMing** (https://sourceforge.net/projects/xming/), and be sure that PuTTY is configured to forward X connections.
- Alert course staff immediately if you are unable to set up your X connection (e.g. if you can't run `xeyes`).
- Run "export DISPLAY=localhost:0.0" if the terminal prompts "Cannot open the display"
- If working on your own machine, make sure you have the necessary libraries to compile graphics. Try executing the following from within the `q4` directory:
  - `g++20 window.cc graphicsdemo.cc -o graphicsdemo -lX11` (Note: this is a dash followed by lower case L followed by X and then one one.)
  - Run the program: `./graphicsdemo`
- Some students on Windows have had issues in the past with getting the information to display properly. https://stackoverflow.com/questions/24361267/x11-window-isnt-updating-drawing-automatically talks about this--do you get a display update if you resize the window? Is https://github.com/microsoft/wslg/issues/558 possibly relevant? (updating the $DISPLAY variable in particular)

Mac OS users:

- On machines running Mac OS, you will need to install **XQuartz.** (See https://www.xquartz.org/)
- Once installed, you must restart your machine.
- You will also need to explicitly tell `g++` where the `x11` library is located.
- If the above compilation command does not work, browse through your Mac's file system looking for a directory `X11` that contains directories `lib` and `include`.
- You must then specify the `lib` directory using the `-L` option and the `include` directory using the `-I` (uppercase i) option.
- For most installations, the following should work (from within the `q4` directory): `g++20 window.cc graphicsdemo.cc -o graphicsdemo -lX11 -L/usr/X11/lib -I/usr/X11/include`

Enabling X11 and VS Code on Windows:

- see if the instructions at https://x410.dev/cookbook/enabling-ssh-x11-forwarding-in-visual-studio-code-for-remote-development/ help. Some students have reported success this way.
- Another alternative: "I got VcXsrv working with a SSH config file on VsCode. Had to add an Environment Variable to my windows machine and as well as change add a JSON Property for vscode in settings.json"