# Keegan Sanchez

Vancouver, WA | (360) 909-1769 | keegan.sanchez@wsu.edu | *https://github.com/Lurgypai*

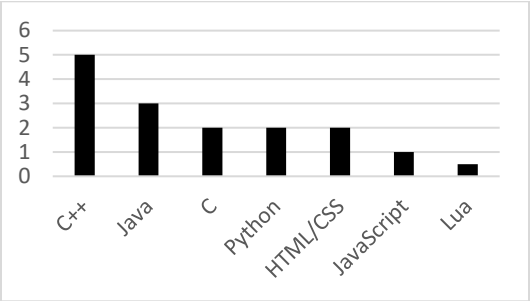
## Education

**Washington State University** **B.S. Computer Science**
Class of May 2022

**GPA:** *3.98*

## Languages

I began programming as a hobbyist in 2013. Depicted to the right are my years of experience with all the languages I've used. See my GitHub for samples of my C++ work. (*https://github.com/Lurgypai*)




**Tools I'm Comfortable Working With:** Visual Studio, Eclipse, IntelliJ, PyCharm, Vim, CMake, Git

## Experience

**WSU Vancouver / Research Assistant** (May 2021 – Current)
*Researching solid state firmware using VSSIM and the OpenSSD FPGA platform. Implemented custom FTL and buffer management algorithms.*

**WSU IT / Student Employee** (November 2020 – May 2021)
*Work assisting students and faculty troubleshoot software problems, and update/install software. Provide individuals with troubleshooting and support via phone and video calls.*

**ArcaneMC / Independent Developer** (June 2017 – February 2018)
*Commission based work building server plugins for Minecraft. Collaborated with a team of developers using Slack and BitBucket in conjunction with Git to manage development.*

## Projects

*I have worked on a variety of personal projects, written mainly in C++. I primarily enjoy programming games. C++ projects are in my github repository,* https://github.com/Lurgypai.

**rummagesale.net** (2020 – Present, *http://rummagesale.net*) Developing a custom website from scratch working with HTML and CSS. Learning to host on an Ubuntu server using and Apache HTTP Server. Designing interesting layouts with cohesive formatting.

**Suqua** (2018 – Present, *https://github.com/Lurgypai/Suqua*) Writing 2d online multiplayer networking libraries using C++, SDL, OpenGL, and ENET. Learned to use sockets, serialize/unserialize data for network transfer. Wrote an authoritative server network model, including client side prediction and rollback. Wrote the graphical engine, which interfaces directly with OpenGL. Learned to effectively buffer data to the GPU, as well as how to write vertex and fragment shaders. Learned to use compute shaders to handle a simple particle system. Cross platform on Windows and Linux using CMake to manage build systems.