William Cromar

wcromar@knights.ucf.edu
904-814-9428

Skills and Technologies

Languages: Java, Python, C, C++, Octave, SQL, HTML/CSS, LaTeX
Operating Systems: Linux (including Bash command line), Windows 7/8/10
Development Tools: iPython, Git, Subversion (SVN), MS Visual Studio, CMake
Libraries: Scikit-Learn, TensorFlow, NumPy, NLTK, Seaborn

Education

University of Central Florida *Fall 2015 - Spring 2019 (expected)*
Bachelor of Science, Computer Science *Current GPA 3.88*
Minor: Statistics
Burnett Honors College

St. Johns River State College *Fall 2013 - Spring 2015*
Computer Programming and Analysis *Final GPA: 3.9*

Experience

Member, JV Programming Team (UCF) *September 2016 - Present*

- Attend weekly practice competitions to improve problem solving skills and programming ability. Implement efficient software solutions in **Java** within a short time-frame.
- Collaborate and communicate with other team members to design and implement creative solutions to challenging problems. Test solutions to ensure correctness.

Software Engineer Intern (DiSTI) *October 2015 - September 2016*

- Developed and maintained leading-edge virtual environment software written in **C++**. Identified, reported, and resolved bugs in a large code-base.
- Tested software solutions and reviewed code written by team members to ensure consistent software quality. Created and improved automated tests using in-house tools, as well as custom **Python** scripts.

Projects

`github.com/will-cromar`

Sentiment Classifier: Developed a natural language sentiment classifier in **Python** using **sklearn**, capable of identifying positive or negative tone in bodies of text. Trained using real-world natural language data, preprocessed with **NLTK**. Incorporated sentiment classifier into larger group project providing stock market news analysis, price prediction, and report generation.

Connect-4 AI: Implemented an automated *Connect-4* player from scratch in **C**. Incorporates probabalistic methods such as Monte-Carlo Search Trees, as well as heuristic techniques from game theory to efficiently achieve near-optimal play.

PL/0 Compiler: Collaborated with a team of two other people to implement, in **C**, a compiler for the PL/0 programming language. Capable of lexing, parsing, and generating bytecode for PL/0 source files. Implemented virtual machine to test generated bytecode.

Text Editor: Developed a basic graphical plain-text editor in **Java** using the **Swing** framework, capable of reading and writing to text files. Constructed UML class diagram to document object relationships.