# ASCII Art Studio Project Report

Developer: Toros Kutlu

Stockholm University

DA2005 Programming techniques VT24

## Program and Task(s) Implemented

This project implements an ASCII Art Studio, a Python-based application that allows users to convert images into ASCII art. The program supports loading JPG and PNG images, converting them to grayscale, and rendering them as ASCII art with adjustable dimensions while preserving the aspect ratio. The ASCII art output width is set to 50 characters by default, with height adjusted accordingly.

## Starting and Using the Program

To start the program, navigate to the project directory and run the User\_Interface.py script using a Python 3 interpreter. The program provides a command-line interface where users can interact through commands like 'load <filename>', 'render', 'info', and 'quit'. After loading an image with the 'load' command, users can use the 'render' command to display the ASCII art in the console. The 'info' command displays details about the currently loaded image.

## Libraries/Modules Used

The project utilizes the Pillow library (PIL Fork) for image processing tasks such as loading, converting images to grayscale, and resizing. Pillow is not included in Python's standard distribution and needs to be installed using pip, Python's package installer. To install Pillow, run the following command in your terminal or command prompt: 'pip install Pillow'.

## Program Structure

The program is structured into two main files: ASCII\_Art\_Studio.py and User\_Interface.py. ASCII\_Art\_Studio.py contains the ASCII\_Art\_Studio class, responsible for the core functionality like loading images, converting them to grayscale, resizing, and generating ASCII art. User\_Interface.py defines the User\_Interface class, which handles user interactions through a command-line interface. The separation of concerns is maintained, with image processing logic encapsulated in the ASCII\_Art\_Studio class and user interaction handled by the User\_Interface class.

## Testing Strategy and Implementation

The project includes a comprehensive testing suite in the file Test\_ASCII\_Art\_Studio.py, which uses Python's ‘unittest’ framework. The tests cover all major functionalities of the ASCII Art Studio, including loading images, rendering ASCII art, and user interface commands. Mocking is used to simulate user inputs and file operations, ensuring that the tests are isolated from the external environment. The custom test runner and result classes enhance the output of test results, providing clear and detailed feedback on test execution. This thorough testing approach ensures the reliability and robustness of the ASCII Art Studio application.

## Additional Reflections

The code design emphasizes modularity, readability, and ease of use. Object-oriented programming principles are applied to separate concerns and enhance maintainability. The choice of data structures and algorithms is driven by the need for efficiency and simplicity, with PIL's image processing capabilities leveraging optimized underlying libraries for performance. Future enhancements could include more advanced image processing features, user-configurable ASCII character sets, and support for additional image formats.