# FC723 Programming Theory

**Assessment Title:** Portfolio Project 1

**Module Code:** FC723

**Class/Group:** Group B

**Tutor Name:** Callum Birkett

**Student GUID Number:** 3075459T

**Date of Submission:** 2025-1-25

I confirm that this assignment is my own work. Where I have referred to academic sources, I have provided in-text citations and included the sources in the final reference list.

## Table of Contents

* Pseudocode for Euclidean Algorithm
* Python Implementation
* Code Refactoring and Comments
* Git Workflow and Version Control
* Proposed Algorithm Extension

## Pseudocode for Euclidean Algorithm

Explanation:

* The algorithm calculates the GCD by iteratively replacing the larger number with the remainder of the division until one of the numbers becomes zero.

文本

AI 生成的内容可能不正确。

* Step 1: Input two numbers A and B.
* Step 2: While both numbers are not zero:
  + If A > B, set remainder R = A % B, then swap values A = B, B = R.
  + Otherwise, swap A and B.
* Step 3: Return the GCD, which is stored in A.

## Python Implementation

文本

AI 生成的内容可能不正确。

The principle and the steps are the same as the Pseudocode part. Thus, no more explanation here.

## Code Refactoring and Comments

* Changes made to improve the code:
* Used clear and simple variable names.
* Used a,b=b,a instead of c=a, a=b, b=c.
* Added the validate\_input() function to ensure correct user input.
* Steps taken to make the code easier to read:
* Checked and fixed spaces and indentation.
* Included structured comments to explain each function's purpose and logic.
* Divided the code into separate functions to keep it organized.
* Why Certain Refactoring Decisions Were Made:
* Improved readability: Separating them into functions like validate\_input() Euclidean() and is\_coprime() makes the code easier to understand and maintain.
* Better reusability: All functions help avoid code repetition.
* Error handling: The try-except block prevents crashes by handling invalid inputs properly.
* How Readability Improvements Affect Maintainability:
* Easier to update and debug because of well-structured code and clear comments.
* Modular functions make adding new features simpler.
* Organized code reduces mistakes and makes future changes easier.

## Git Workflow and Version Control

图形用户界面, 文本

AI 生成的内容可能不正确。

[GitHub Repository Link](https://github.com/xb0616/FC723-Portfolio-Assessment-1-P449777): https://github.com/xb0616/FC723-Portfolio-Assessment-1-P449777

I used github desktop to achieve version control, so I will list the steps about it rather than the terminal steps

* Upload Project to GitHub Desktop:
* Open GitHub Desktop and click “Add Local Repository”.
* Select your project folder and add it to GitHub Desktop.
* Commit Changes:
* In the “Changes” tab, review the modified files.
* Enter a short description (e.g., "Initial commit") in the Summary field.
* Click “Commit to main” to save changes locally.
* Publish to GitHub:
* Click “Publish repository” to upload your project to GitHub.
* Once published, future changes can be uploaded by clicking “Push origin”.

## Proposed Algorithm Extension

* To extend the functionality of the Euclidean algorithm, a new function is\_coprime () which utilizes the Euclidean algorithm to determine whether two numbers are coprime has been introduced. Two numbers are considered coprime if their GCD = 1.
* Other new function is validate\_input() which checks the user’s input.
* In the main function the algorithm first takes user input for two numbers, computes their GCD using the Euclidean function, and then checks if the result is 1 to check if they are coprime.
* Pseudocode for Algorithm Extension

文本

AI 生成的内容可能不正确。

* Python Implementation for Algorithm Extension

文本

AI 生成的内容可能不正确。