A close-up of a logo

Description automatically generated

Modelling Of Software

Intensive Systems

Assignment 2: CBC

1st Master computer science

2024-2025

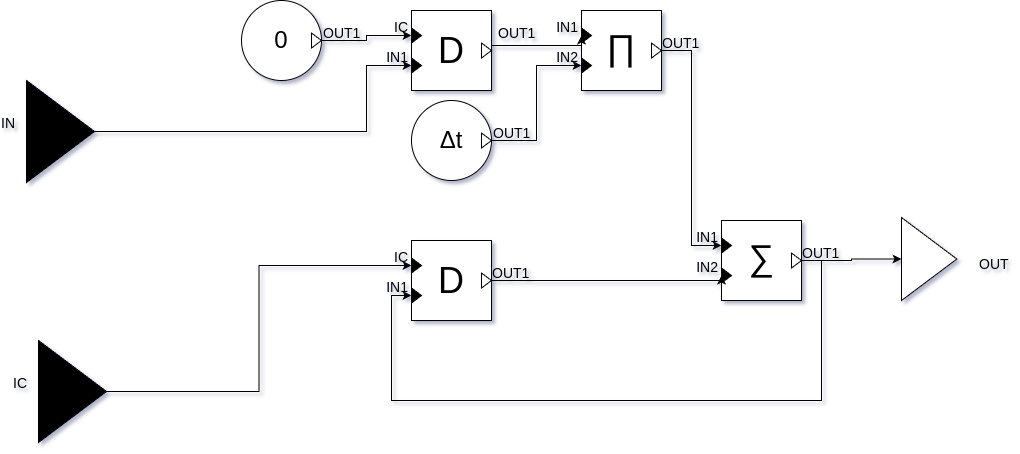
Liam Leirs

Robbe Teughels

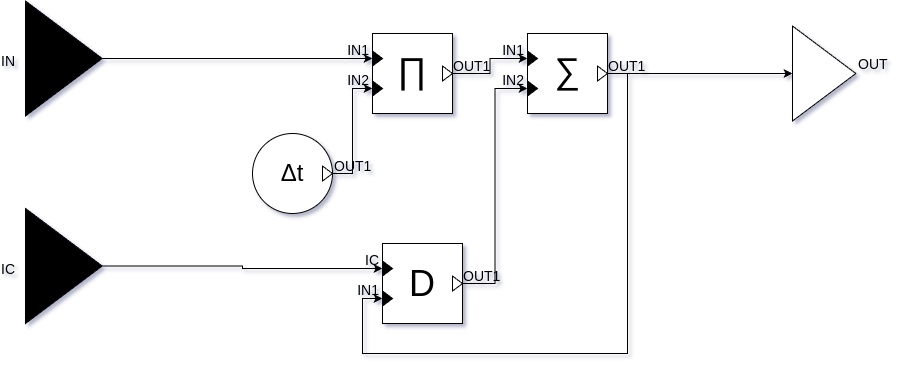
# Integration Methods

For each integrator, we constructed a custom CBD block with an initial condition (IC) input. This initialization ensures that each integrator block starts from a defined value.

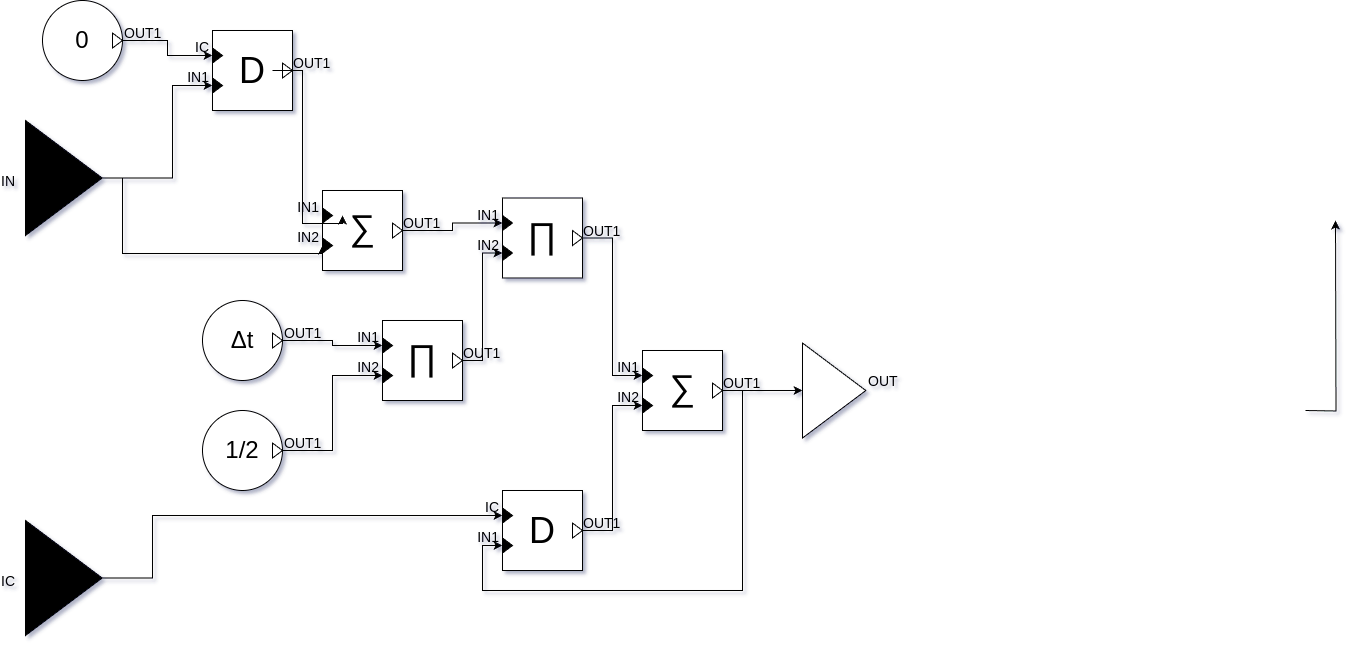
**Backwards euler**



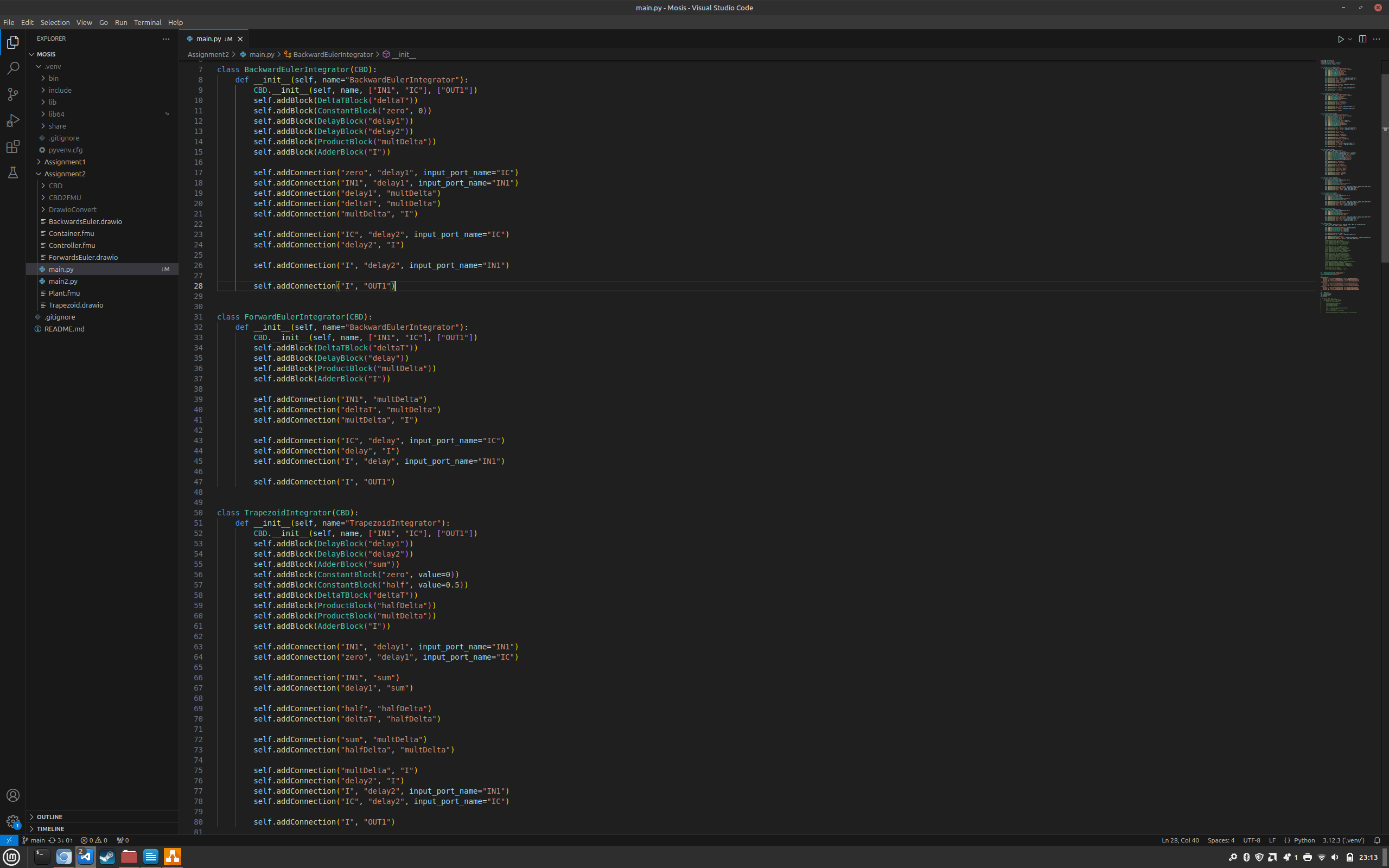
**Forwards euler**

****

**Trapezoid rule**

****

**Python code for blocks**

****

# **Comparison**

Computing the integral for g(t) we get the following values:

|  |  |  |  |
| --- | --- | --- | --- |
| Delta t | Backwards euler  value | Forwards euler  value | Trapezoid rule  value |
| 0.1 | 3.222190908877023 | 3.223153281886757 | 3.2226720953818915 |
| 0.01 | 3.213459296657502 | 3.213555450684053 | 3.213507373670786 |
| 0.001 | 3.212588796472303 | 3.212598411043006 | 3.21259360375765 |

Comparing with the analytical solution we get:

|  |  |  |  |
| --- | --- | --- | --- |
| Delta t | Backwards euler  error | Forwards euler error | Trapezoid rule error |
| 0.1 | 0.009698804877023015 | 0.010661177886757134 | 0.010179991381891629 |
| 0.01 | 0.0009671926575021139 | 0.001063346684053279 | 0.0010152696707863562 |
| 0.001 | 9.669247230315037e-05 | 0.0001063070430062929 | 0.00010149975765028074 |

As delta t increases, the approximation gets closer and closer to the actual value. Here we can also see that the backwards euler method gives us the best approximation in this case.

# Co-Simulation

Text