**Active Redundancy Implementation**

# Introduction

This document discusses the technologies used to implement the Active Redundancy algorithm, as well as how the work was divided among the team members. The assignment builds upon the previous Heartbeat assignment, introducing Active Redundancy into the preexisting Heartbeat implementation.

# Tools and Technologies

The following technologies were used to implement the algorithm:

* **Programming Language:** Python 3.7
* **IDE:** PyCharm 2019.2.3 (Community Edition)
* **VCS:** Git (Github)
  + <https://github.com/dh7445/HeartBeat.git>

# Source Code

The source code uses the same structure as the source code for the Heartbeat assignment. The major changes consist of:

* Duplicating the Sender and Receiver class to create the backup process.
* Modifying the backup Sender to remove the nondeterministic fault.
* Modifying the Monitor to print the values from the primary processes and using the values from the backup processes in case the primary one encounter an error.

# Tasks Distribution

For the purpose of completing the assignment, the tasks were distributed and performed as follow:

* The team met to discuss the assignment and work on the implementation.
* The implementation was done by using pair programming: Diego was in charge of writing the actual code while José and Murtaza provided live feedback and support during the process.
* José typed the documentation and Diego and Murtaza verified it.

# Challenges

To implement the Active Redundancy logic, we duplicated the logic of the Sender and Receiver from the primary processes into the backup processes, modifying them to suit the roles of the backup processes. However, we encountered a problem where the backup processes weren't behaving correctly. A lot of time was spent deciphering what the problem was to no avail. In the end, we scratched the code made and copied the logic from the primary process to the backup processes again. This resolved the issue, but we are not sure why since, as far we could tell, the logic remained the same.

Other than that, the development proceeded smoothly. From a business logic perspective, the most challenging decision we encountered was deciding how to present the logic in such a way that made demonstration of the Active Redundancy simple.