Software Design Pattern: Prototype Design

We have coded our calculator with the *prototype design pattern*.

For this pattern, we implemented multiple prototypes as described in the below macro-architectural UML design. With the prototype design pattern, we were able to make a copy of the original objects in the calculator rather than creating the new instance of the calculator from scratch. This pattern allowed us to save time and resources as many of the objects had complex processes.

p = prototype -> Clone()

Return copy of self

Return copy of self

Return copy of self

**Macro Architecture Findings**:

* In CalculatorPrototype1, we created a calculator based on the interviews done in D[1] with a rough draft of the math functions but the calculator did not show any sign of “ON” or “OFF” with a ‘0’ upon entering in a number and/calculation.
* In CalculatorPrototype2, we revised the calculator to tailor to the needs of our user’s responses from the interview. We revised the functions so that they no longer gave a “double” number solution if it was supposed to be an “int” number solution.
* In CalculatorPrototype3, we finalized the functions and added in *exception handling* to ensure that the calculator met the needs of both the interviewees and our potential users.