**Scientific calculator project**

**Membri del gruppo 6:**

* Luca Taddeo [l.taddeo5@studenti.unisa.it](mailto:l.taddeo5@studenti.unisa,it) (referente)
* Andrea Landi [a.landi88@studenti.unisa.it](mailto:a.landi88@studenti.unisa.it)
* Antonio Scaldaferri [a.scaldaferri4@studenti.unisa.it](mailto:a.scaldaferri4@studenti.unisa.it)
* Filippo Somma [f.somma13@studenti.unisa.it](mailto:f.somma13@studenti.unisa.it)

**Fase 1: Pre-game:**

**Link bacheca Trello:** https://trello.com/b/SAiN1zvs/calcolatrice

**Definition of Formatting:**

* Blank Space prima e dopo gli operandi;
* Blank Space dopo la virgola e non prima;
* Compact control readability style;
* Convenzione *ClassNameTest* per i nomi delle class test (e.g. la class test della classe *Adder* si chiamerà *AdderTest*);
* Convenzione *testMethodName* per i nomi dei test methods (e.g. il test method del metodo *add* si chiamerà *testAdd*).

**Definition of Done:**

* Acceptance criteria rispettati;
* Superamento degli unit tests;
* Superamento dei test funzionali (di integrazione);
* Codice formattato correttamente;
* Documentazione delle classi completa.

**Linguaggio /ambiente di sviluppo:** Java/ Apache Netbeans.

**Design architecture of the program:**

|  |  |
| --- | --- |
| Come pattern architetturale è stato scelto il Model-View-Controller in modo da separare la logica di presentazione dei dati (view) dalla logica di business (model). Viene utilizzato il framework JavaFX, per cui:   * La classe **StackPerGliperandi** corrisponde al **Model**; * La classe **FXMLDocumentController** corrisponde al **Controller**; * La **View** è rappresentata dal file **FXML.fxml**. |  |

**Sprint Planning:**

**Stima iniziale della Project Velocity:** 11

**Tasks scelti per questo Sprint:**

* *#1 Costruzione stack*:   
  Creazione classe per lo stack che implementi la tecnologia LIFO.
* *#2 Notazione operandi*:  
  Implementare notazione cartesiana per i numeri complessi.

Implementare notazione per i numeri reali

* *#3Scelta operandi*:  
  Utilizzare gli ultimi operandi per eseguire un'operazione.
* *#4 Allocazione del risultato*:  
  Allocare il risultato nello stack.
* #7 Somma:

Implementare la somma degli ultimi due operandi.

**Tasks assegnati ai membri del Team:**

* Luca Taddeo: *#4 Allocazione del risultato, #3 Scelta operandi.*
* Andrea Landi: *#2 Notazione operandi.*
* Antonio Scaldaferri: *#1 Costruzione stack.*
* Filippo Somma: *#7 Somma.*

**Goal 1° sprint:**

Visualizzazione del risultato di una somma tra due operandi inseriti mediante un’interfaccia a riga di comando.