**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:**

**Tool utilizzati:** Trello, GitHub, Tableau (per i grafici).

**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**. |  |

**Primo Sprint Planning:**

**Stima iniziale della Project Velocity:** 11

**User stories scelte 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*.*

Implementazione della classe ComplexNumber.

* *#3 Scelta operandi*:

Utilizzare gli ultimi operandi per eseguire un'operazione.

* *#7 Allocazione del risultato:*

Allocare il risultato nello stack.

* *#7 Somma:*

Implementare la somma degli ultimi due operandi.

Implementazione mediante il pattern Factory Method.

**User stories assegnate 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.

**Primo Sprint - Sviluppo**

**Tasks completati dai membri del Team:**

* Luca Taddeo: *#7 Somma (Implementazione mediante il pattern Factory Method),*

*#8 Differenza (Implementazione mediante il pattern Factory Method),*

*#9 Prodotto (Implementazione mediante il pattern Factory Method),*

*#10 Rapporto (Implementazione mediante il pattern Factory Method),*

*#11 Radice quadrata (Implementazione mediante il pattern Factory Method),*

*#12 Inversione di segno (Implementazione mediante il pattern Factory Method);*

* Andrea Landi: *#2 Notazione operandi (Implementazione della classe ComplexNumber),*

*#7 Somma (Implementazione mediante il pattern Factory Method),*

*#8 Differenza (Implementazione mediante il pattern Factory Method),*

*#9 Prodotto (Implementazione mediante il pattern Factory Method),*

*#10 Rapporto (Implementazione mediante il pattern Factory Method),*

*#11 Radice quadrata (Implementazione mediante il pattern Factory Method),*

*#12 Inversione di segno (Implementazione mediante il pattern Factory Method);*

* Antonio Scaldaferri: *#1 Costruzione stack,*

*#5 Visualizzazione elementi,*

*#6 Interfaccia grafica,*

*#7 Somma (Implementazione mediante il pattern Factory Method),*

*#8 Differenza (Implementazione mediante il pattern Factory Method),*

*#9 Prodotto (Implementazione mediante il pattern Factory Method),*

*#10 Rapporto (Implementazione mediante il pattern Factory Method),*

*#11 Radice quadrata (Implementazione mediante il pattern Factory*

*Method),*

*#12 Inversione di segno (Implementazione mediante il pattern Factory Method);*

* Filippo Somma: *#2 Notazione operandi (Implementazione della notazione cartesiana per i*

*numeri complessi e della notazione per i numeri reali),*

*#3 Scelta operandi,*

*#4 Allocazione del risultato,*

*#5 Visualizzazione elementi,*

*#6 Interfaccia grafica,*

*#7 Somma (Implementare la somma degli ultimi due operandi),*

*#8 Differenza Implementare la differenza degli ultimi due operandi),*

*#9 Prodotto (Implementare il prodotto degli ultimi due operandi),*

*#10 Rapporto (Implementare il rapporto degli ultimi due operandi),*

*#11 Radice quadrata (Implementazione mediante il pattern Factory Method),*

*#12 Inversione di segno (Implementare l'inversione di segno dell'ultimo elemento inserito nello stack);*

**Sprint review:**

* **User stories complete alla fine di questo sprint:** *#1 Costruzione stack,*

*#2 Notazione operandi,*

*#3 Scelta operandi,*

*#4 Allocazione del risultato,*

*#5 Visualizzazione elementi,*

*#6 Interfaccia grafica,*

*#7 Somma,*

*#8 Differenza,*

*#9 Prodotto,*

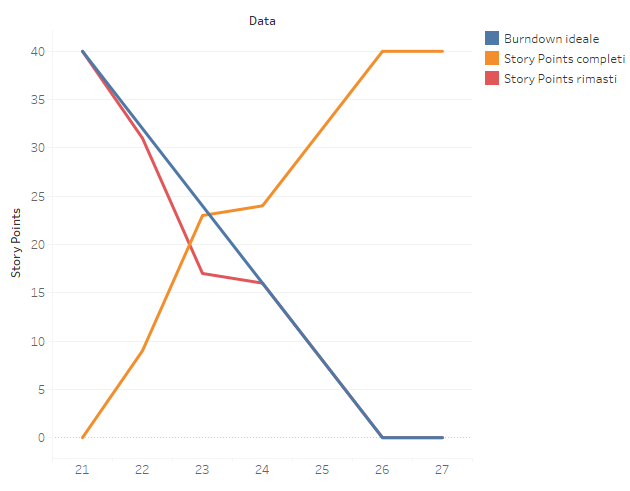
*#10 Rapporto,*

*#11 Radice quadrata,*

*#12 Inversione di segno.*

* **Project Velocity misurata:** 40

Il primo giorno di lavoro il team si accorge di una stima errata sia della velocity che degli story points attribuiti alle singole user stories. Ne consegue che il gruppo ha contestualmente rieffettuato la stima degli story points e il calcolo della nuova velocity.



**Retrospective:**

* **Keep doing:**

1. Continuare ad utilizzare Trello sfruttando i suoi Power Up in modo da esplicitare tutte le informazioni associate ad ogni user story per organizzare meglio il lavoro;
2. Continuare ad utilizzare GitHub in modo da dover effettuare poche revisioni in seguito ai vari merge.

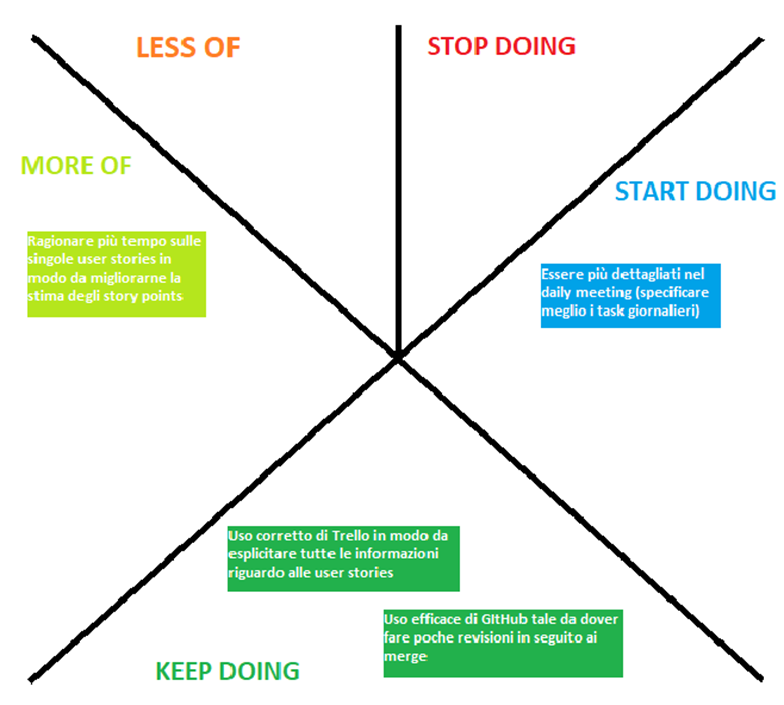
* **More of**: impiegare più tempo a ragionare sulle singole user stories poiché la stima effettuata in fase di pre-game non è stata accurata.

A parità di story points, infatti, alcuni task han richiesto più lavoro rispetto ad altri.

Segue l’elenco delle stime modificate.

1. Notazione operandi (8 SP);
2. Scelta operandi (1 SP);
3. Radice quadrata (5 SP).

* **Start doing:** è necessario essere più dettagliati nei daily meeting nello specificare i task giornalieri.



**Secondo Sprint Planning:**

**Stima iniziale della Project Velocity:** 36

**User stories scelte per questo Sprint:**

* *#13 CLEAR stack*:   
  Eliminare tutti gli elementi presenti nello stack.

Fare in modo che la stringa "clear" venga letta correttamente.

* *#14 DROP stack*:  
  Implementare la funzione di rimozione dell'ultimo elemento inserito nello stack.

Fare in modo che la stringa "drop" venga letta correttamente.

* *#15 DUP stack:*

Implementare la funzione di duplicazione dell'ultimo elemento nello stack.

Fare in modo che la stringa "dup" venga letta correttamente.

* *#16 SWAP stack*:  
  Implementare lo scambio di posizione degli ultimi due elementi dello stack.

Fare in modo che la stringa "swap" venga letta correttamente.

* *#17 OVER stack:*  
  Implementare la funzione che permette di inserire una copia del penultimo elemento dello stack.

Fare in modo che la stringa "over" venga letta correttamente.

* *#18 Notazione variabili:*

Verificare il corretto inserimento dei valori all'interno delle variabili.

* *#19 Set varabile*:  
  Implementare la funzione che rimuove l'ultimo elemento inserito nello stack e lo salva nella

variabile scelta.

* *#20 Get variabile*:  
  Implementare la funzione che prende il valore della variabile selezionata e lo inserisce nello

stack.

* *#21 Somma variabile*:  
  Implementare la somma tra il top dello stack e una variabile in input salvando in risultato in

quest'ultima.

* *#22 Differenza variabile:*

Implementare la differenza tra una variabile in input e il top dello stack salvando in risultato

nella variabile.

* *#23 Operazione personalizzata*:  
  Implementare la funzione che permette di definire una nuova operazione concatenando operazioni di default e/o operazioni precedentemente definite.

**User stories assegnate ai membri del Team:**

* Luca Taddeo: *#18 Notazione variabili, #20 Get variabile, #21 Somma variabile, #22 Differenza variabile.*
* Andrea Landi: *#18 Notazione variabili, #19 Set variabile, #23 Operazione personalizzata.*
* Antonio Scaldaferri: *#13 CLEAR stack, #14 DROP stack.*
* Filippo Somma: *#15 DUP stack, #16 SWAP stack, #17 OVER stack.*

**Goal 2° sprint:**

Aggiungere all’applicazione le funzionalità avanzate di:

* CLEAR, DROP, DUP, SWAP e OVER sullo stack;
* memorizzazione di un valore all’interno di una variabile;
* somma e sottrazione utilizzando le variabili come operandi.

**Primo Sprint - Sviluppo**

**Tasks completati dai membri del Team:**

* Luca Taddeo:
* Andera Landi:
* Antonio Scaldaferri:
* Filippo Somma:

**Sprint review:**

* **User stories complete alla fine di questo sprint:** *#13 CLEAR stack*;

*#14 DROP stack*;*#15 DUP stack; #16 SWAP stack*;*#17 OVER stack;#18 Notazione variabili;#19 Set varabile*;*#20 Get variabile*;*#21 Somma variabile*;*#22 Differenza variabile;#23 Operazione personalizzata*.

* **Project velocity misurata:**

**Retrospective:**

* **Keep doing:**

1. Continuare ad utilizzare Trello sfruttando i suoi Power Up in modo da esplicitare tutte le informazioni associate ad ogni user story per organizzare meglio il lavoro;
2. Continuare ad utilizzare GitHub in modo da dover effettuare poche revisioni in seguito ai vari merge.

* **More of:**

1. Definire in modo più preciso tutti i task relativi alle user stories;

* **Start doing:**
* **Less of:**
* **Stop doing:**