Documentazione progetto: spirografo

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 3

Analisi 4

1.4 Analisi del dominio 4

1.5 Analisi e specifica dei requisiti 4

1.6 Use case 6

1.7 Pianificazione 6

1.8 Analisi dei mezzi 8

1.8.1 Software 8

1.8.2 Hardware 8

2 Progettazione 8

2.1 Design dell’architettura del sistema 8

2.2 Design dei dati e database 8

2.3 Design delle interfacce 8

2.4 Design procedurale 8

3 Implementazione 9

4 Test 9

4.1 Protocollo di test 9

4.2 Risultati test 13

4.3 Mancanze/limitazioni conosciute 13

5 Consuntivo 13

6 Conclusioni 13

6.1 Sviluppi futuri 13

6.2 Considerazioni personali 13

7 Bibliografia 13

7.1 Bibliografia per articoli di riviste: 13

7.2 Bibliografia per libri 13

7.3 Sitografia 14

8 Allegati 14

# Introduzione

## Informazioni sul progetto

Quello che verrà descritto in questo documento è il primo lavoro a lungo termine, del terzo anno, che mi è stato assegnato. La data di inizio del progetto è il 9 settembre, avrò 16 lezioni disponibili compresa la prima per consegnare la documentazione e terminare lo sviluppo del programma. La data di fine è quindi fissata al 23 dicembre. Sarò il solo a lavorare a questo progetto, per cui punto a potermi organizzare meglio, non dovendo coordinare altre persone, ma terrò anche in conto che la mole di lavoro sarà maggiore. L’obiettivo di questo progetto è di creare un’applicazione gestita con GUI che permetta di simulare i disegni di uno spirografo. L’applicazione dispone di diverse funzioni, oltre che al semplice disegno, in modo da migliorare l’esperienza generale col prodotto. Il linguaggio che ho scelto è Java, per via delle conoscenze acquisite sui JFrame nello scorso anno scolastico.

## Abstract

### Situazione iniziale

La funzione principale di questo progetto è quella di simulare uno spirografo. Uno spirografo è uno strumento, spesso venduto come giocattolo, usato per tracciare delle curve tramite un anello e un disco che gira al suo interno. Mettendo una matita o una penna nel disco, e facendo girare il disco dentro l’anello, la matita sul foglio traccerà delle particolari curve chiamate ipotrocoide, delle curiose linee che attraggono l’attenzione per via della loro particolarità. In rete circolano già dei software che simulano degli spirografi e ho persino trovato qualche versione da browser fatta piuttosto bene.

### Attuazione

Lo sviluppo di questo progetto è stato lungo e complesso: pur andando spedito alle prime lezioni, e pur avendo un risultato seppur minimo nella seconda lezione.

### Risultato

## Scopo

Questo progetto è volto a creare un’applicazione per computer che simuli uno spirografo. Il prodotto finale dovrà essere un’applicazione in Java con GUI, che permetta di simulare i disegni di uno spirografo, con aggiunta di diverse funzioni volte a semplificare e migliorare l’esperienza generale.

## Analisi

## Analisi del dominio

Questo capitolo dovrebbe descrivere il contesto in cui il prodotto verrà utilizzato, da questa analisi dovrebbero scaturire le risposte a quesiti quali ad esempio:

* Background/Situazione iniziale
* Quale è e come è organizzato il contesto in cui il prodotto dovrà funzionare?
* Come viene risolto attualmente il problema? Esiste già un prodotto simile?
* Chi sono gli utenti? Che bisogni hanno? Come e dove lavorano?
* Che competenze/conoscenze/cultura posseggono gli utenti in relazione con il problema?
* Esistono convenzioni/standard applicati nel dominio?
* Che conoscenze teoriche bisogna avere/acquisire per poter operare efficacemente nel dominio?
* …

## Analisi e specifica dei requisiti

|  |  |  |
| --- | --- | --- |
| **ID** | **Requisito** | **Priorità** |
| 01 | Finestra ridimensionabile | Bassa |
| 02 | Dischi e anelli ridimensionabili | Media |
| 03 | Sempre al centro della tela abbiamo il disco sulla quale ruota il secondo disco o un anello che ospita un disco che gira | Alta |
| 04 | Il disco e l’anello non devono essere più grandi della pagina | Alta |
| 05 | Il disco mobile ha più punti dove puoi attaccare la matita | Media |
| 06 | Il colore del percorso è personalizzabile | Alta |
| 07 | È possibile disegnare il percorso in layer gestiti separatamente | Media |
| 08 | La pagina avrà un menù per gestire i layer | Media |
| 09 | I layer possono venir nascosti o cancellati | Bassa |
| 10 | Quando il colore del percorso viene cambiato, viene chiesto anche se cambiare layer | Bassa |
| 11 | È possibile tracciare più percorsi con misure degli anelli diversi | Media |
| 12 | Per ogni tracciato è possibile interrompere l’operazione | Alta |
| 13 | L’immagine creata può venir esporta come png | Media |
| 14 | Salvataggio e caricamento dei parametri | Bassa |
| 15 | Premendo il tasto spazio è possibile inserire una modalità automatica | Media |

**Spiegazione elementi tabella dei requisiti:**

**ID**: identificativo univoco del requisito

**Nome**: breve descrizione del requisito

**Priorità**: indica l’importanza di un requisito nell’insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

**Versione**: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l’ultima versione, mentre le vecchie dovranno essere inserite nei diari.

**Note**: eventuali osservazioni importanti o riferimenti ad altri requisiti.

**Sotto requisiti**: elementi che compongono il requisito.

## Use case

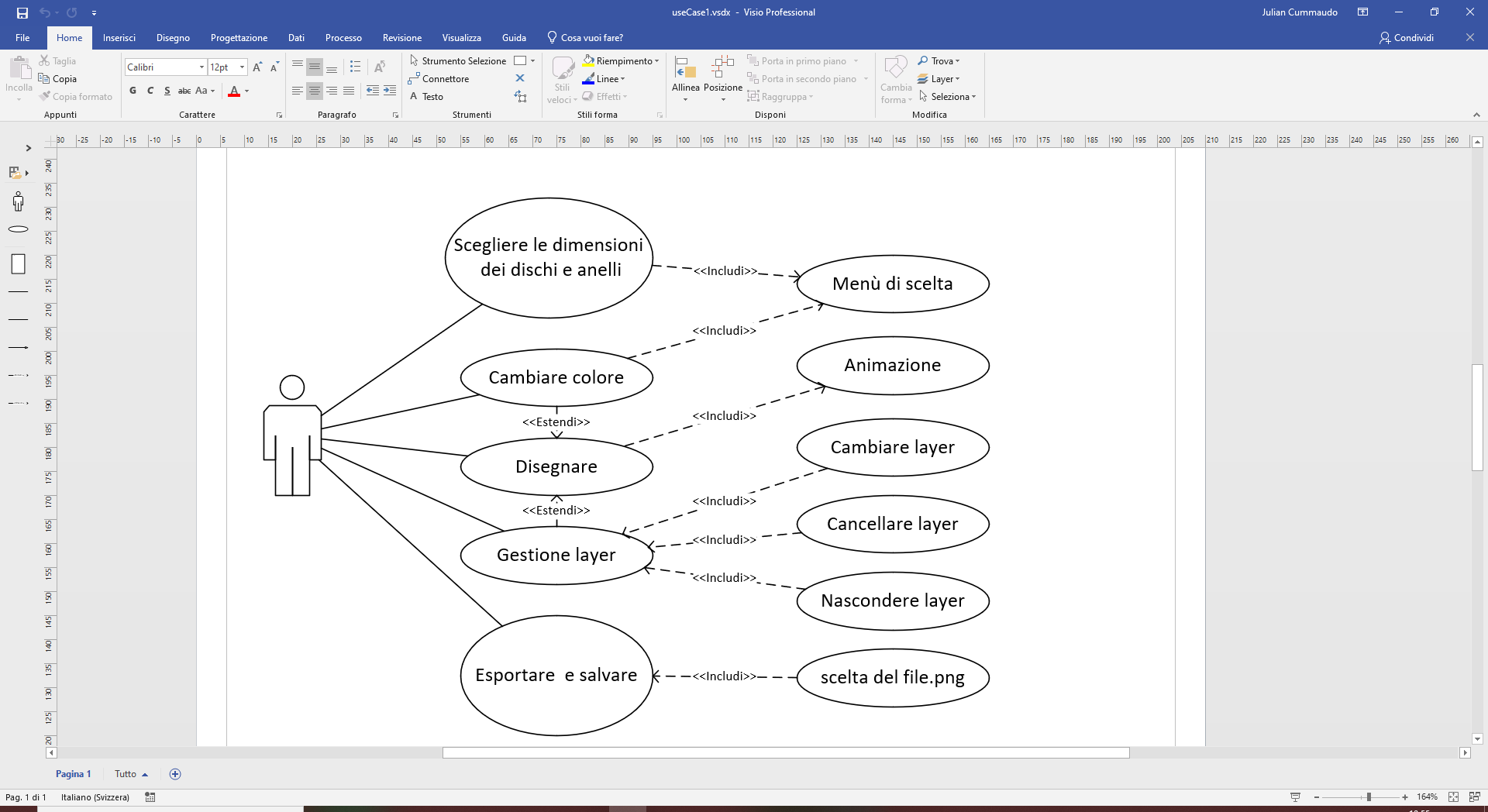


Figura 1: Use Case

## Pianificazione

Per la realizzazione di questo progetto ho utilizzato un modello di pianificazione waterfall.

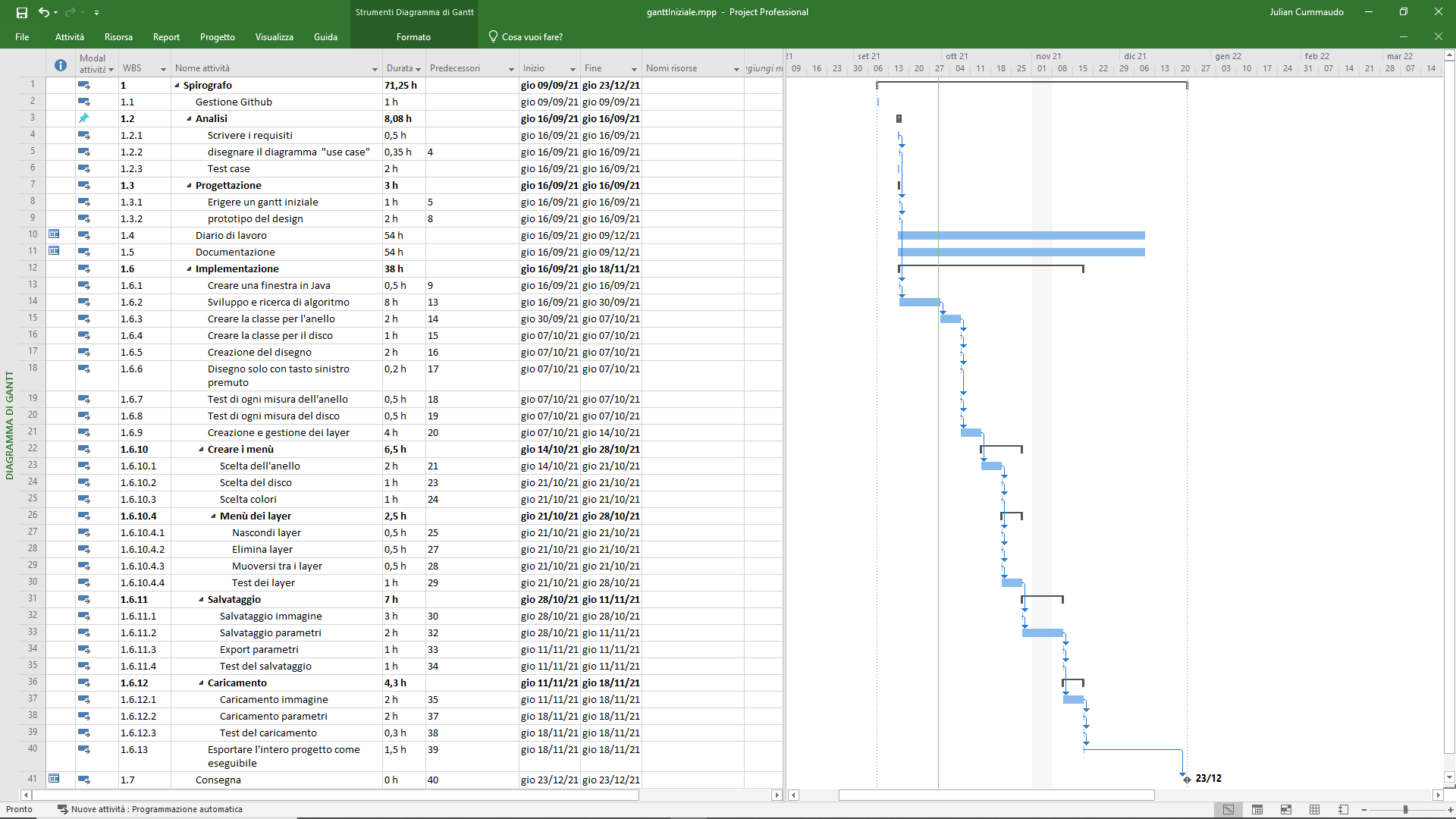


Figura 2: Gantt iniziale

## Analisi dei mezzi

Elencare e descrivere i mezzi disponibili per la realizzazione del progetto. Ricordarsi di sempre descrivere nel dettaglio le versioni e il modello di riferimento.

### Software

Per scrivere il programma in Java è stato usato Apache NetBeans IDE 12.4.

Per gestire i file e le cartelle è stato usato Git.

Per scrivere la documentazione ed il diario è stato usato Word da Microsoft Office Professional Plus 2019.

Per creare il Gantt iniziale ed il Gantt consuntivo, è stato usato Microsoft Project Professional 2019.

Per creare il diagramma use case, è stato usato Microsoft Visio Professional 2019.

Per progettare il documento di design è stato usato l’applicativo web WireFramePro dal sito: <https://mockflow.com/apps/wireframepro>.

### Hardware

Il programma può essere usato su qualsiasi computer che abbia installato almeno l la JDK 16.

## Design delle interfacce

Il primo documento di design delle interfacce è stato concluso il 30 settembre del 2021, i prototipi delle interfacce si presentano come segue:

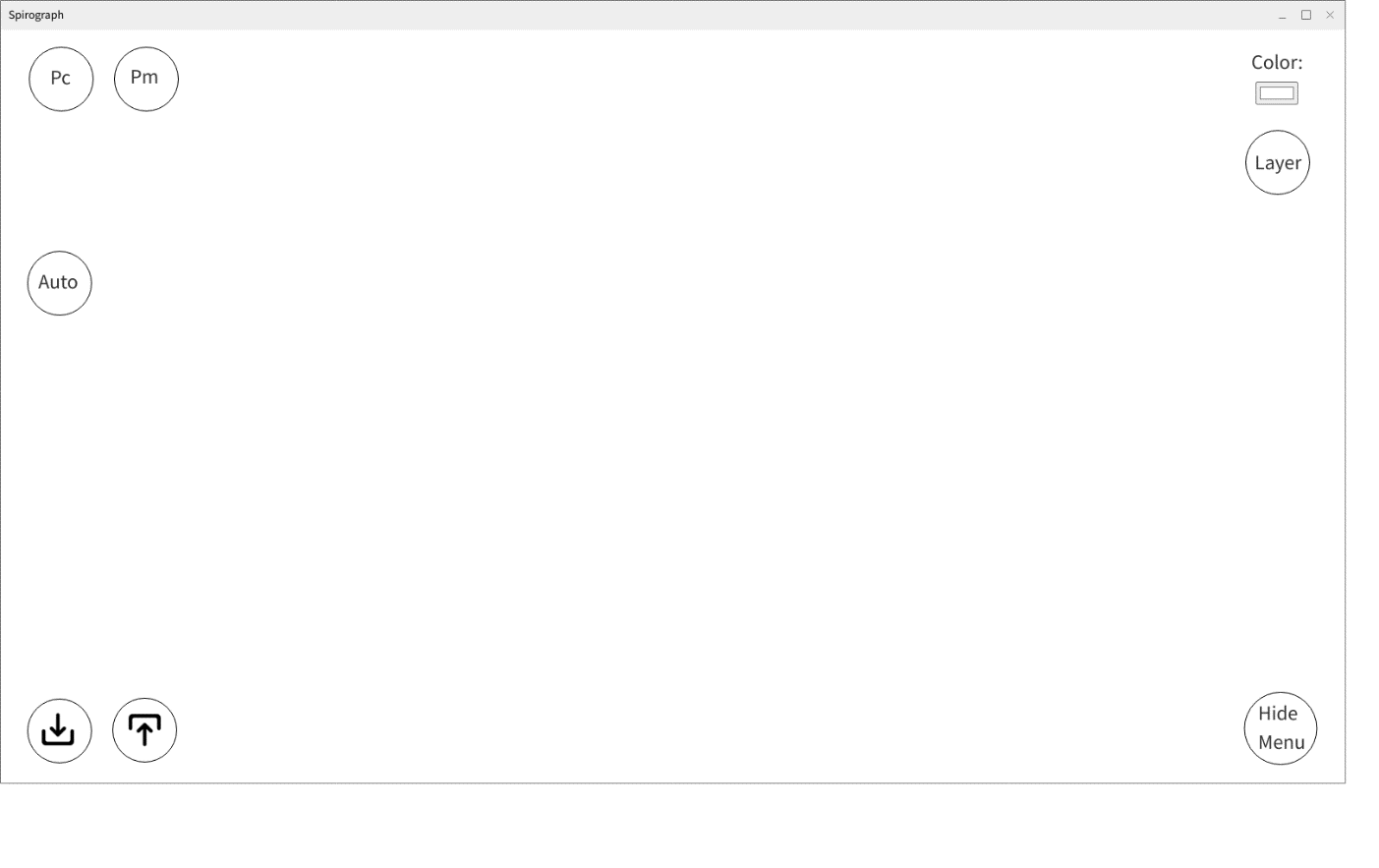


Figura 3: Interfaccia iniziale

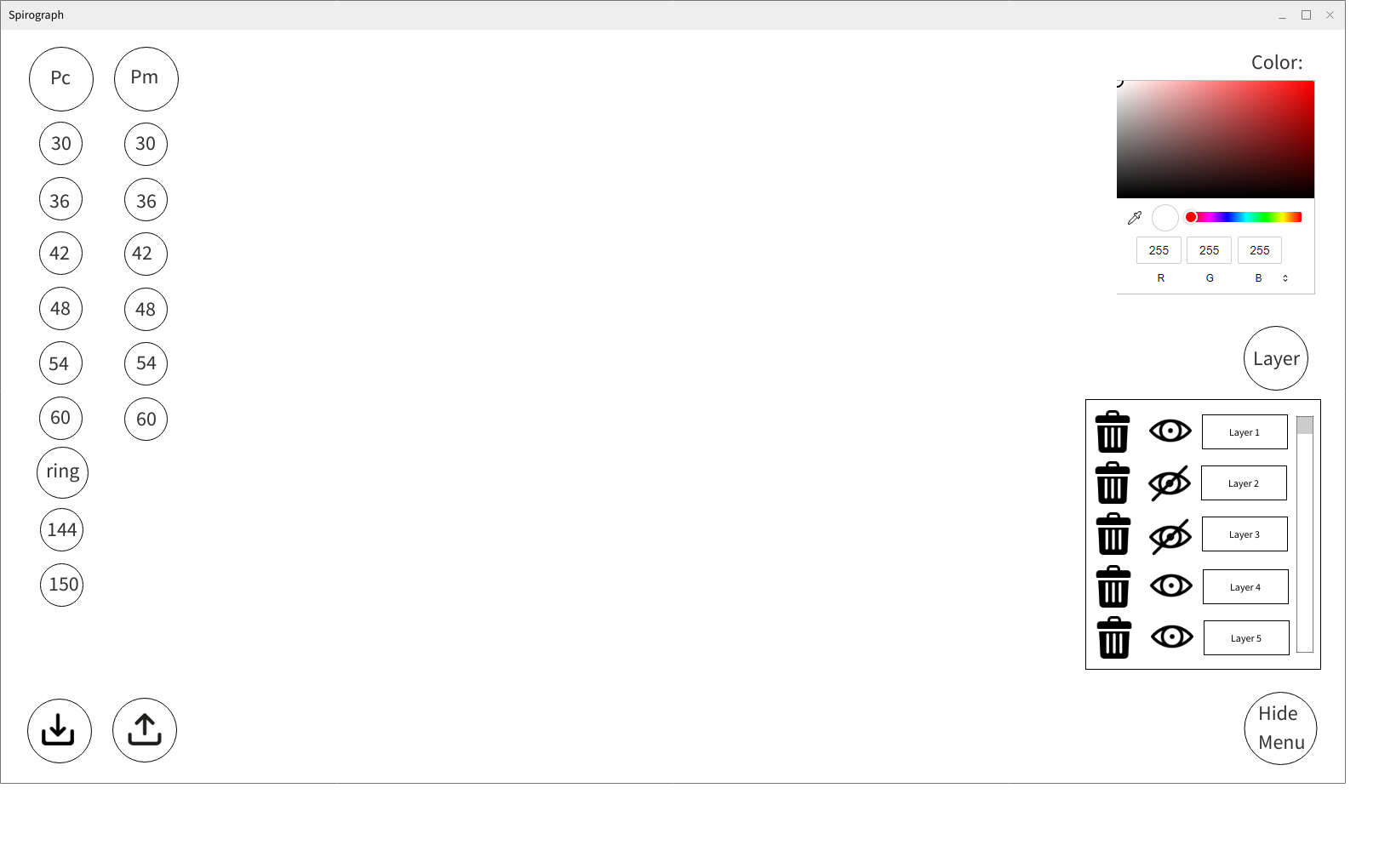


Figura 4: Menù dell'interfaccia iniziale

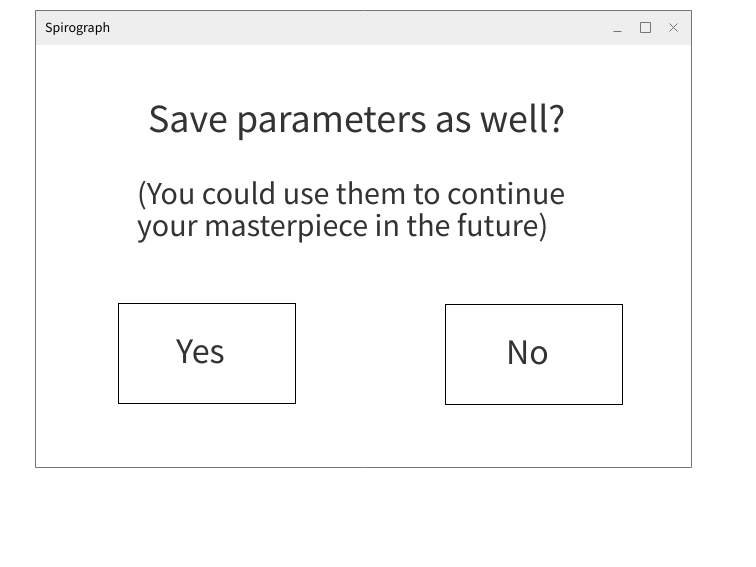


Figura 5: Richiesta di salvare i parametri oltre all'immagine

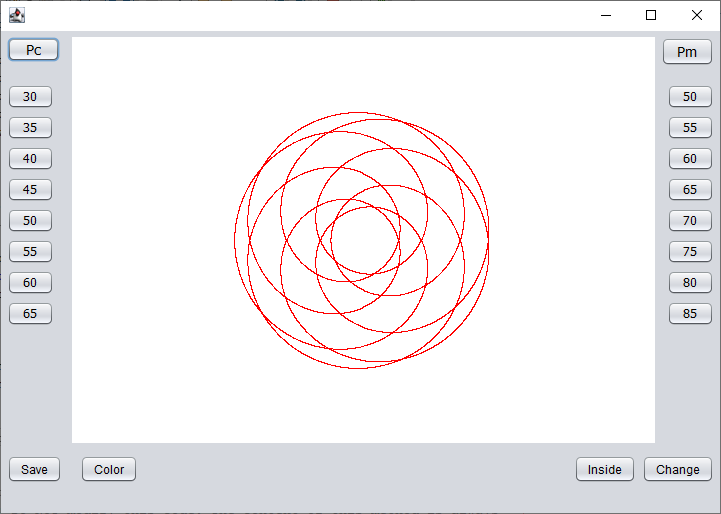


Figura 6: Interfaccia finale

# Implementazione

## Classi:

### Samt.Spiro.Graphics.Canvas.java

Questa è la classe principale del progetto, dove vengono eseguiti tutti i calcoli per disegnare effettivamente lo spirografo. In questa classe sono presenti diversi metodi con il compito di gestire lo spirografo, qui di seguito spiegherò gli attributi che ha questa classe:

#### Attributi

##### private Double radiusMobile;

##### private Double radiusCenter;

Questi due attributi sono il cuore del disegno: radiusMobile rappresenta il raggio del cerchio in movimento, che ruota attorno al disco o anello fisso al centro della tela, di conseguenza, radiusCenter rappresenta il raggio del cerchio o dell’anello fisso al centro. L’attributo fromButtonRadiusMobile, si differenzia da radiusMobile per via dei calcoli che vengono eseguiti su quest’ultimo. Questi primi attributi, infatti, sono quelli che determinano le dimensioni effettive, in pixel, dei raggi, ma questo verrà spiegato nel dettaglio nella sezione dei metodi. Per entrambi gli attributi, sono presenti i metodi setter e getter, il metodo setter ha come particolarità quella impostare gli attributi “fromButton” con il valore passato al metodo, inoltre, ricaricando il raggio mobile o centrale, a dipendenza di quale setter si sta usando. I metodi di “reload” verranno spiegati nella sezione dei metodi.

##### private Double fromButtonRadiusMobile;

##### private Double fromButtonRadiusCenter;

Questi due attributi, come spiegato sopra, rappresentano i raggi dei dischi fissi ed in movimento nel disegno. La differenza tra questi attributi “fromButton” è che qui dentro vengono salvati i valori presi direttamente dai bottoni sull’interfaccia grafica. Questo vuol dire che se il bottone del raggio mobile 60 viene premuto, la variabile fromButtonRadiusMobile conterrà il valore “80”, mentre la variabile radiusMobile conterrà invece il valore, calcolato con le dimensioni della finestra, usato per disegnare lo spirografo. Questi due attributi non posseggono alcun metodo setter o getter, ma vengono impostati dai metodi setter di radiusMobile e radiusCenter.

##### private Double offset = 40.0;

L’attributo offset contiene il valore sulla quale si poserà l’ipotetica matita sul cerchio a partire dal bordo del disco. In breve: se l’offset è a 40, come di default, mentre il raggio del disco mobile è a 80, allora, in una situazione con uno spirografo fisico, corrisponderebbe a piazzare la matita a metà del raggio del disco, per poi ruotarlo attorno al disco centrale. Questo attributo possiede un metodo getter e un metodo setter standard.

##### private boolean insideMode = true;

Questo attributo determina se il disco all’esterno dovrà girare intorno al disco centrale, oppure se dovrà farlo all’interno. In una situazione reale con uno spirografo fisico, questo corrisponderebbe a decidere se il disco mobile girerà attorno ad un disco centrale, oppure se dovrà girare all’interno di un anello più grande. In questo caso, se l’attributo è true, vuol dire che il disco centrale sarà interno, quindi il disco mobile ci girerà attorno. Se invece l’attributo dovesse essere false, allora il disco mobile girerà all’interno del disco centrale. l’attributo possiede un metodo getter e per venire cambiato viene usato un metodo chiamato changeInsideMode(), che verrà spiegato meglio nella sezione dei metodi.

##### private Point2D.Double current = new Point2D.Double(0,0);

Questo attributo current contiene le coordinate del punto corrente che deve venir disegnato sul canvas per arrivare a rappresentare il disegno di uno spirografo. Questo punto viene calcolato in base alle formule trovate sul sito: <http://www.mathematische-basteleien.de/spirographs.htm> grazie ai metodi getOutsidePoint() e getInsidePoint(). Questo attributo non possiede alcun setter o getter, in quanto viene usato solo da questa classe.

##### private ArrayList<Drawing> drawings = new ArrayList<Drawing>();

La lista drawings è una lista contenente tutti i disegni che l’utente ha deciso di salvare in un layer. Nel momento in cui l’utente decide di salvare un disegno in un layer, tutti i punti che compongono il disegno corrente vengono salvati in questa lista. Questo attributo non possiede metodi setter o getter.

##### private int indexOfDrawing = 0

Questo attributo d’appoggio serve solo a tenere in memoria l’indice del disegno corrente, memorizzato nella lista drawings. L’attributo possiede un metodo setter.

##### private Color pointColor = Color.RED;

L’ultimo attributo è pointColor, che determina il colore con la quale disegnare i punti, e di conseguenza, lo spirografo. Questo attributo possiede un metodo setter.

#### Metodi

Questa classe è composta da molti attributi, di conseguenza non andrò ad elencare tutti i metodi setter e getter in questa sezione, se non per delle eccezioni. Questi sono i metodi presenti nella classe Canvas.java:

##### public Canvas()

Il metodo costruttore istanzia un oggetto di tipo Canvas, inoltre, imposta dei valori di default ai raggi mobili e centrali, infine inizializza tutti i componenti

##### public void setRadiusCenter(Double radius)

##### public void setRadiusMobile(Double radius)

Questi due metodi setter impostano il valore radius agli attributi “fromButton”, in modo da tenere in memoria il valore selezionato dall’utente, così da poter fare i calcoli del disegno usando questo valore. Questi due metodi invocano anche due metodi di reload, la funzione di questi viene spiegata qui sotto.

##### public void reloadRadiuses()

Questo metodo invoca entrambi i metodi seguenti:

##### public void reloadRadiusMobile()

##### public void reloadRadiusCenter()

Questi due metodi, richiamati dai setter dei raggi, servono ad aggiornare il raggio usato per disegnare lo spirografo, in modo da calcolarlo usando le dimensioni della finestra.

##### public void changeInsideMode()

Un semplice metodo che inverte la insideMode, richiamato direttamente dal bottone nella GUI che inverte la modalità.

##### public void changeDrawing(Drawing next)

Questo metodo cambia il disegno corrente, aumentando di uno indexOfDrawing e aggiungendo il nuovo disegno next alla lista drawings.

##### public void saveImage()

##### public Point2D.Double getOutsidePoint(Double pc, Double pm, Double offset, Double angolo)

##### public Point2D.Double getInsidePoint(Double pc, Double pm, Double offset, Double angolo)

##### protected void paintComponent(Graphics g)

I primi due metodi servono a calcolare il punto attuale, in base ad un angolo. Tutto ciò che questi metodi fanno è utilizzare le formule matematiche per la creazione delle curve di uno spirografo trovate in rete. Vengono richiamati nel paintComponent(), che controlla se insideMode è true, o false, e utilizza di conseguenza il metodo corretto. Il metodo paintComponent(), in breve, si preoccupa di ciclare 60000 volte la scelta e la stampa di un punto, a partire dal centro della tela. Ho deciso di ciclare più volte del necessario per avere abbastanza angoli diversi sulla quale disegnare lo spirografo, avendo meno cicli, e quindi meno punti, il disegno sarebbe sembrato normale per dimensioni dei raggi ridotte, ma aumentando i raggi e quindi le dimensioni effettive del disegno, quest’ultimo non avrebbe avuto abbastanza punti per dare l’illusione di una curva.

### Samt.Spiro.Helpers.Drawing.java

#### Attributi

##### private ArrayList<Point2D> points = new ArrayList<Point2D>();

##### private ArrayList<Color> pointColor = new ArrayList<Color>();

Questi due attributi si occupano del salvataggio totale di un disegno. Per ogni punto di un disegno salvato in points, viene salvato il colore con cui rappresentarlo nella lista pointColor.

##### private boolean visible = true;

Questo attributo serve a capire se un disegno è visibile o meno, cioè se dovrà venir disegnato o meno.

#### Metodi

##### public void addPointAndColor(Point2D p, Color c)

Questo metodo aggiunge il punto ed il colore passato alle liste points e pointColor.

##### public void hideOrShowDrawing()

Come il metodo changeInsideMode(), nella classe Canvas, questo metodo serve solo ad invertire l’attributo visible.

# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-01  REQ-03  REQ-04 | **Nome:** | Ridimensionare la finestra |
| **Descrizione:** | Verificare che la finestra sia ridimensionabile da una dimensione di 500-400 px fino alla grandezza massima dello schermo, verificando che i menù cambino grandezza in base alla dimensione in modo da restare sempre visibili. | | |
| **Procedura:** | 1. Avviare il programma 2. Trascinando l’angolo in basso a destra, verificare che i menù restino nella giusta posizione 3. Stringere la pagina in larghezza ed in altezza fino alle dimensioni minime richieste 4. Ridimensionare la pagina verificando che gli elementi addetti al disegno restino al centro 5. Ridimensionare la pagina e verificare che il disco e l’anello non escano dalla finestra | | |
| **Risultati attesi:** | La finestra non diventerà più piccola di 500-400 px e non diventerà più grande della grandezza massima dello schermo. I menù e i dischi devono restare nelle posizioni richieste, al centro o ai lati, con le dimensioni corrette. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-02  REQ-05 | **Nome:** | Ridimensionare gli anelli e i dischi. Attaccare la matita a più punti del disco |
| **Descrizione:** | Verificare che siano presenti e funzionanti i menù di ridimensionamento degli anelli e dei dischi, inoltre, verificare che che il disco mobile abbia dei punti dove attaccare la matita. | | |
| **Procedura:** | 1. Impostare il raggio dell’anello a 85px 2. Impostare il raggio del disco a 55px 3. Impostare 3 punti dove la matita potrà attaccarsi al disco 4. Avviare il programma 5. Provare a cambiare punto d’attaccamento della matita 6. Per ogni punto, girare il disco almeno un paio di volte | | |
| **Risultati attesi:** | Alla rotazione del disco, avere un disegno diverso per ogni punto. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-06 | **Nome:** | Personalizzare il colore del percorso |
| **Descrizione:** | Cambiare il colore del percorso durante l’esecuzione del programma. | | |
| **Procedura:** | 1. Avviare il programma 2. Cominciare a disegnare con il colore di base 3. Cambiare tramite un color picker il colore 4. Tornare a disegnare | | |
| **Risultati attesi:** | Il colore dovrebbe cambiare, ma ciò che è stato disegnato prima deve restare del colore base. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-004  REQ-07  REQ-08  REQ-09 | **Nome:** | Disegnare su diversi layer, nasconderli o cancellarli dal menù |
| **Descrizione:** | Verificare che il percorso siagestito in diversi layer e che essi possano venir gestiti da un menù, dove sarà possibile nasconderli o cancellarli. | | |
| **Procedura:** | 1. Avviare l’applicazione 2. Tracciare un percorso 3. Aggiungere dei Layer dal menù 4. Cambiare Layer 5. Tracciare un altro percorso 6. Nascondere il Layer precedente 7. Rivelare il Layer precedente 8. Cancellare il Layer precedente | | |
| **Risultati attesi:** | I Layer creati devono venir visualizzati nel menù, nascondendo un Layer, il percorso tracciato deve sparire dalla tela ma rimanere nel menù, rivelandolo, deve ricomparire invariato. Cancellando un Layer, esso deve sparire dal menù e deve sparire dalla tela. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-005  REQ-10 | **Nome:** | Notificare il cambio del colore con una richiesta di cambiare Layer |
| **Descrizione:** | Cambiando il colore del percorso, deve venir chiesto all’utente, in maniera poco evasiva, se creare un nuovo Layer e se spostarsi in questo nuovo Layer. | | |
| **Procedura:** | 1. Avviare il programma 2. Disegnare con il colore di base 3. Cambiare tramite un color picker il colore 4. Verificare la comparsa del messaggio a destra | | |
| **Risultati attesi:** | Il colore deve cambiare. Al cambiamento del colore deve apparire un messaggio a destra. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-006  REQ-11  REQ-12 | **Nome:** | Tracciare più percorsi con diverse misure ed interrompere l’operazione |
| **Descrizione:** | Verificare che sia possibile interrompere il percorso che si sta tracciando, e che sia possibile tracciare più percorsi con misure degli anelli diverse. | | |
| **Procedura:** | 1. Avviare l’applicazione 2. Tracciare un percorso 3. Interrompere l’operazione smettendo di premere il tasto sinistro 4. Interrompere l’operazione, nel caso sia stato premuto il tasto spazio, con un apposito bottone nel menù 5. Cambiare la dimensione degli anelli 6. Senza cambiare Layer, ricominciare a disegnare | | |
| **Risultati attesi:** | Mollando il tasto destro o schiacciando il bottone di interruzione del percorso, il percorso deve smettere di venir disegnato, rendendo possibile navigare tranquillamente nel menù. Dopo aver cambiato la dimensione degli anelli deve essere possibile ricominciare a disegnare senza problemi, mantenendo il vecchio percorso. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-007  REQ-15 | **Nome:** | Premendo il tasto spazio, far partire la modalità automatica |
| **Descrizione:** | Se l’utente preme il tasto spazio, il percorso si disegnerà da solo. | | |
| **Procedura:** | 1. Avviare il programma 2. Selezionare una misura degli anelli, o lasciare quella di default 3. Premere il tasto spazio | | |
| **Risultati attesi:** | Dopo aver premuto il tasto, il percorso si deve disegnare da solo, seguendo la traiettoria che avrebbe seguito anche col mouse. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-008  REQ-13  REQ-14 | **Nome:** | Salvare l’immagine prodotta e salvare e caricare dei parametri |
| **Descrizione:** | Verificare che sia possibile salvare l’immagine prodotta, inoltre, verificare che sia possibile salvare i parametri dell’immagine prodotta, in modo da poterli caricare in futuro e continuare il disegno. | | |
| **Procedura:** | 1. Avviare l’applicazione 2. Tracciare un percorso 3. Cliccare sul bottone addetto al salvataggio 4. Salvare il file 5. Verificare che il file sia stato creato 6. Verificare che corrisponda al disegno sulla tela 7. Tracciare un altro percorso 8. Salvare i parametri con il bottone adeguato 9. Premere il bottone per caricare i parametri 10. Caricare i parametri e verificare che sia possibile     1. navigare tra i Layer     2. nascondere o cancellare dei Layer     3. disegnare sul percorso senza problemi | | |
| **Risultati attesi:** | Cliccando il pulsante addetto al salvataggio dev’essere possibile salvare l’immagine, aprendo il file creato, dovrà venir visualizzata l’immagine prodotta sulla tela in maniera pulita (senza i vari menù). Premendo il bottone apposito dev’essere possibile salvare i parametri con l’estensione X-X. Premendo il bottone addetto al caricamento dei parametri deve apparire il menù del sistema operativo per caricare un file, dopo aver scelto il file dei parametri, dev’essere possibile eseguire tutti i punti del passaggio 10. | | |

## Risultati test

|  |  |  |
| --- | --- | --- |
| ID | Esito | Eventuali commenti |
| TC001 | Riuscito | Il ridimensionamento è andato a buon fine, la finestra non diventa più piccola di 500X400px |
| TC002 | Riuscito in parte |  |
| TC003 | Fallito | Non ho fatto in tempo a implementare il cambio del colore |
| TC004 | Fallito |  |
| TC005 | Fallito |  |
| TC006 | Fallito |  |
| TC007 | Fallito |  |
| TC008 | Fallito |  |

## Mancanze/limitazioni conosciute

Il progetto ha molte mancanze rispetto a ciò che avrebbe dovuto essere il prodotto finale. Il motivo principale per cui non sono riuscito a implementare tutte queste funzioni è stata la mancanza di organizzazione e l’eccessiva distrazione durante le ore assegnate al progetto in classe. Questi fattori hanno avuto, come diretta conseguenza, un ritardo che non ha fatto altro che incrementare ad ogni lezione. Per cominciare: non è possibile salvare i file o caricare dei parametri per ricreare un’immagine da una vecchia sessione. La mancanza più grande nel progetto è sicuramente l’assenza della possibilità di lavorare con dei layer.

# Consuntivo

# 

Figura 7:Gantt consuntivo

Concluso il progetto, posso affermare che la progettazione iniziale è stata oltremodo inutile e sbagliata. Nonostante non credo che sarei riuscito a fare di meglio, questo mi ha insegnato che dovrei imparare a progettare meglio.

# Conclusioni

Ritengo che la mia soluzione non sia completa, pertanto non avrà sicuramente alcun impatto.

## Sviluppi futuri

Il progetto potrebbe venir concluso perlomeno completando i requisiti minimi.

## Considerazioni personali

Questo progetto è cominciato con il piede giusto, nelle prime lezioni si vedevano possibili miglioramenti, il primo disegno è apparso dopo poche lezioni, portando quindi le aspettative alle stelle. Purtroppo, dopo le lezioni di metà novembre, è andato tutto storto. Da questo progetto ho imparato poco in materia, ma anche che dovrei avere aspettative più basse

# Bibliografia

## Sitografia

09.09.2021

<https://it.wikipedia.org/wiki/Spirograph>  
 <https://nathanfriend.io/inspiral-web/>

30.09.2021

<https://mockflow.com/#Wireframe>

10.07.2021

<http://www.mathematische-basteleien.de/spirographs.htm>