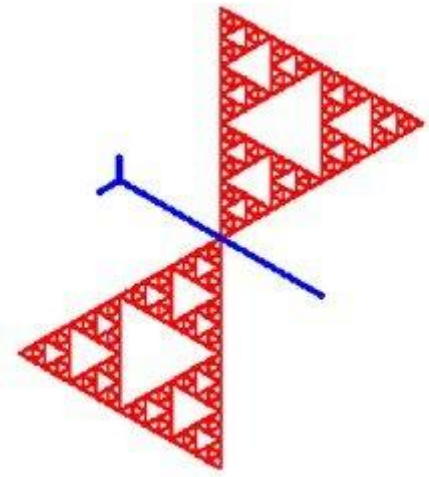
PROGETTO

LINGUAGGIO LOGO

Università degli studi di Camerino

TANIA CARTECHINI

MATRICOLA 115422



Indice

PROGETTO LOGO (link github)

Descrizione delle classi e delle responsabilità

CURSORE

La classe cursore è un oggetto che estende la classe Point (java.awt.Point) e implementa l'interfaccia ICursore, essa viene istanziata all'interno della classe Area.
Ha la responsabilità di gestire i movimenti del cursore sull'area di disegno.
Il cursore si muove grazie a dei comandi (forward e backward) e può disegnare una linea o meno a seconda se l'attributo plot è settato (true o false).
La linea viene tracciata con angolo definito dall'attributo direzione (default: 0°).
La linea tracciata ha uno spessore (default: 1) e un colore (di default: nero).
All'inizio del programma è posizionato sulla home (base/2; altezza/2) che corrisponde al punto centrale dell'area di disegno, la cui dimensione di default: è 640 *480.

Attributi privati

- Point home
- int direzione
- boolean plot
- Color coloreLinea
- Color colore Area
- int penSize
- int base
- int altezza

Costruttore
La classe ha un costruttore parametrico al quale vengono passate le dimensioni dell'area di disegno (int base, int altezza).

Metodi specifici (pubblici)

- void forward (int distanza): sposta il cursore in avanti data "direzione" per una distanza "dist" .
- void backward (int distanza): utilizza il metodo forward con distanza negativa.
- void left (int direzione): ruota il cursore in senso antiorario di gradi "direzione".
- void right (int direzione): ruoto il cursore in senso orario di gradi "direzione".
- void penup: stacca il pennino dall'area di disegno (plot= false).
- void pendown: attacca il pennino dall'area di disegno.
- void home: posiziona il cursore al centro dell'area disegno.

Questi metodi sono delle funzionalità fondamentali per l'oggetto cursore e sono quindi dei metodi esposti dall'interfaccia ICursore.

Getters/Setters

Tutti gli attributi possiedono i getters e i setters tranne base ed altezza (solo set) e home (get).

LINEA

La classe Linea rappresenta un segmento sull'area di disegno.
Essa può essere tracciata direttamente con il comando logo LINE oppure con movimento forward o backward del cursore quando il pennino è attaccato al foglio (isPlot()==true).
La classe Linea implementa l'interfaccia Comparable<Linea> che ci permette di controllare se due linee sono sovrapposte e della stessa lunghezza.

ATTRIBUTI PRIVATI

- Point estremo1 (java.awt.Point)
- Point estremo2
- int spessore
- Color colore (java.awt.Color)

COSTRUTTORI

- un costruttore senza parametri in cui vengono definiti i valori di default
- un costruttore parametrico al quale vengono passati tutti gli attributi.

La classe linea non presenta metodi specifici, però implementa il metodo compareTo(Linea linea) esposto dall'interfaccia Comparable<Linea> ed esegue l'override del metodo toString() per facilitare la scrittura del file di output.

GETTERS/SETTERS

Gli attributi presentano solo getters, in quanto la creazione della linea è una responsabilità della classe Area.

SPEZZATA

La classe Spezzata è formata da una o più oggetti di tipo linea.
Essa inizialmente ha cardinalità uguale a 1, in quanto una singola linea viene gestita come una spezzata dalla classe Area. Ogni qualvolta che viene tracciata una linea vengono controllati se i suoi estremi vanno a concatenarsi ad una spezzata già esistente.
La classe Spezzata è la superclasse della classe poligono, quindi nel caso che la linea aggiunta renda la spezzata chiusa viene creato un oggetto di tipo Poligono.

Attributi protetti

- ArrayList<Linea> linee

Per renderlo visibile alla sottoclasse Poligono.

Attributi privati

- Point estremo1
- Point estremo2

Costruttori

- un costruttore di default
- un costruttore parametrico (costruisce una spezzata di cardinalità uguale 1).

Metodi specifici

- boolean addLinea (Linea linea): aggiunge o meno una linea alla spezzata.
- Poligono creaPoligono(Linea linea): crea o meno un poligono.
- Linea getLinea (int index): restituisce la linea di indice “index”.

Getters/setters
Tutti gli attributi presentano i getters e i setters tranne l’ArrayList di linee che non possiede un setter poiché è creato tramite il metodo addLinea.

POLIGONO

La classe Poligono è una sottoclasse della classe Spezzata.
Essa aggiunge l'attributo colore che corrisponde al colore di riempimento.
Essa può essere creata tramite il comando logo POLYGON oppure tramite il metodo creaPoligono(Linea linea) di Spezzata.
Il colore del poligono è stabilito dall'attributo colore Area di cursore se creato tramite il metodo creaPoligono(Linea linea), altrimenti se creato tramite il comando logo POLYGON allora viene preso dal parametro del comando stesso.

Attributi privati

- Color colore (colore dell'area)
- Attributi ereditati da spezzata

Costruttori

- due costruttori parametrici (il primo costruisce il poligono da una spezzata aperta e una linea, mentre il secondo da un insieme di vertici, il colore dell'area e il numero di lati).

Metodi specifici

La classe non presenta metodi specifici, però esegue l'override del metodo toString() per facilitare la scrittura del file di output

Getters/Setters

L'attributo colore possiede il getter e il setter.

PARSER

La classe Parser ha la responsabilità di leggere il file sorgente scritto nel linguaggio LOGO caricandolo in una lista di stringhe che verrà poi utilizzata dalla classe Interprete.
Durante la lettura del file, vengono saltate le righe vuote e tutti i comandi letti vengono trasformati in maiuscolo (case insensitive).
La classe Parser implementa l'interfaccia IParser.

Attributi privati

- List <String> righe: le righe del file
- String filePath: il percorso del file sorgente

Costruttori

- un costruttore di default,
- un costruttore parametrico (con parametro il filePath)

Metodi specifici pubblici

- parse (): lettura del file sorgente.
- toString(): override del toString che restituisce il file sorgente.

Getters/Setters

L'attributo righe possiede un getter.

AREA

La classe Area ha la responsabilità di gestire il cursore, le figure geometriche (linee, spezzate e poligoni) e di produrre in output il file logo elaborato costituito solo da linee e poligoni.
La classe Area implementa l'interfaccia IArea.
Essa prende le dimensioni dell'area di disegno e il colore di sfondo dal comando LOGO "SIZE" se presente, altrimenti imposta i valori di default e successivamente passa le dimensioni al costruttore del cursore.

Attributi privati

- Color coloreSfondo
- ArrayList<Spezzata> spezzate
- ArrayList<Poligono> poligoni
- int base
- int altezza
- Cursore cursore

Costruttore

- uno di default

Metodi specifici pubblici

- boolean isLinea (): controllo se la linea è già esistente.
- void addSpezzata (Spezzata spezzataUnaLinea): aggiunge la spezzata di cardinalità 1 come parametro ad una spezzata già esistente eventualmente creando un poligono oppure crea una spezzata.
- void addPoligono (Poligono poligono): aggiunge un poligono all'area di disegno.
- void forward (int dist): muove il cursore in avanti e se il pennino è attaccato al foglio allora traccia la linea.
- void backward (int dist): muove il cursore indietro, richiama il metodo forward con distanza negativa.
- void clearScreen(): cancella il contenuto delle liste di spezzate e poligoni.
- void size(int base, int altezza, Color coloreSfondo): imposta le dimensioni dell'area di disegno e il suo colore di sfondo.
- void output (String path): genera il file in output del percorso stabilito dal "Path".

Getters/Setters

Tutti gli attributi possiedono i getters e i setters tranne le liste di poligoni e spezzate.

COMANDI

La classe Comandi è un HashMap<String, Integer.> il quale contiene tutti i comandi di logo.
Essa viene utilizzata da esecutore per l'esecuzione dei comandi passati al metodo esegui(String[] riga).
Ad ogni chiave (comando) è associato un valore intero numerico.
La classe Comandi implementa IComandi.

Attributi ereditati da HashMap<String, Integer>

Costruttore

- un costruttore di default

Metodi specifici pubblici

- void addComandi(String comando): permette l'aggiunta di nuovi comandi per future implementazioni.
- Integer ricerca Comandi(String comando): ricerca il comando interno dell'HashMap se trovato restituisce il valore associato alla chiave altrimenti restituisce -1.

Getters/Setters
Non presenta né getter né setter.

ESECUTORE

La classe Esecutore ha la responsabilità di eseguire i comandi passati dall'interprete tramite il metodo esegui(String [] riga).
Il comando ricevuto dall'interprete viene ricercato nell'hashmap dei comandi "mappa" e successivamente viene restituito il valore corrispondente alla chiave, ad ogni valore dell'hashmap corrisponde un case dello switch del metodo esegui (String[] riga).
Ogni case richiama una funzionalità diversa in base al comando passato.

Attributi privati

- int numLatiPoligono
- Color colore
- Area area
- Comandi mappa

Costruttore

- Un solo costruttore parametrico (Area area)

Metodi specifici pubblici

- void esegui (String[] riga) throws Exception: esegue il comando corrispondente al valore della chiave restituito da ricerca comando (riga[0]).
- void ripeti (int n, String comandi) throws Exception: ripete una lista di comandi per n volte.
- void Poligono (List <String[]> params): crea un poligono da una lista di vertici.

Getters/Setters

Non presenta né getter né setter.

INTERPRETE

La classe Interprete ha la responsabilità di analizzare le righe restituite dal Parser e inviarle alla classe Esecutore.

Attributi privati

- Parser parser
- Area area
- Esecutore esecutore

Costruttori

- un costruttore di default
- un costruttore parametrico Esecutore (Area area, String filePath)

Metodi specifici

- void interpreta(): interpreta le righe del file logo passate dal Parser e richiama il metodo esegui(String[] riga) di Esecutore.

Getters/Setters

Solo l'attributo "area" presenta un getter e un setter.

APP

La classe App è una sottoclasse di Application. Essa ha la responsabilità di visualizzare a video le figure geometriche create in Area.

Attributi privati

- Area area
- Pane root

Costruttori

- Non presenta alcun costruttore

Metodi specifici pubblici

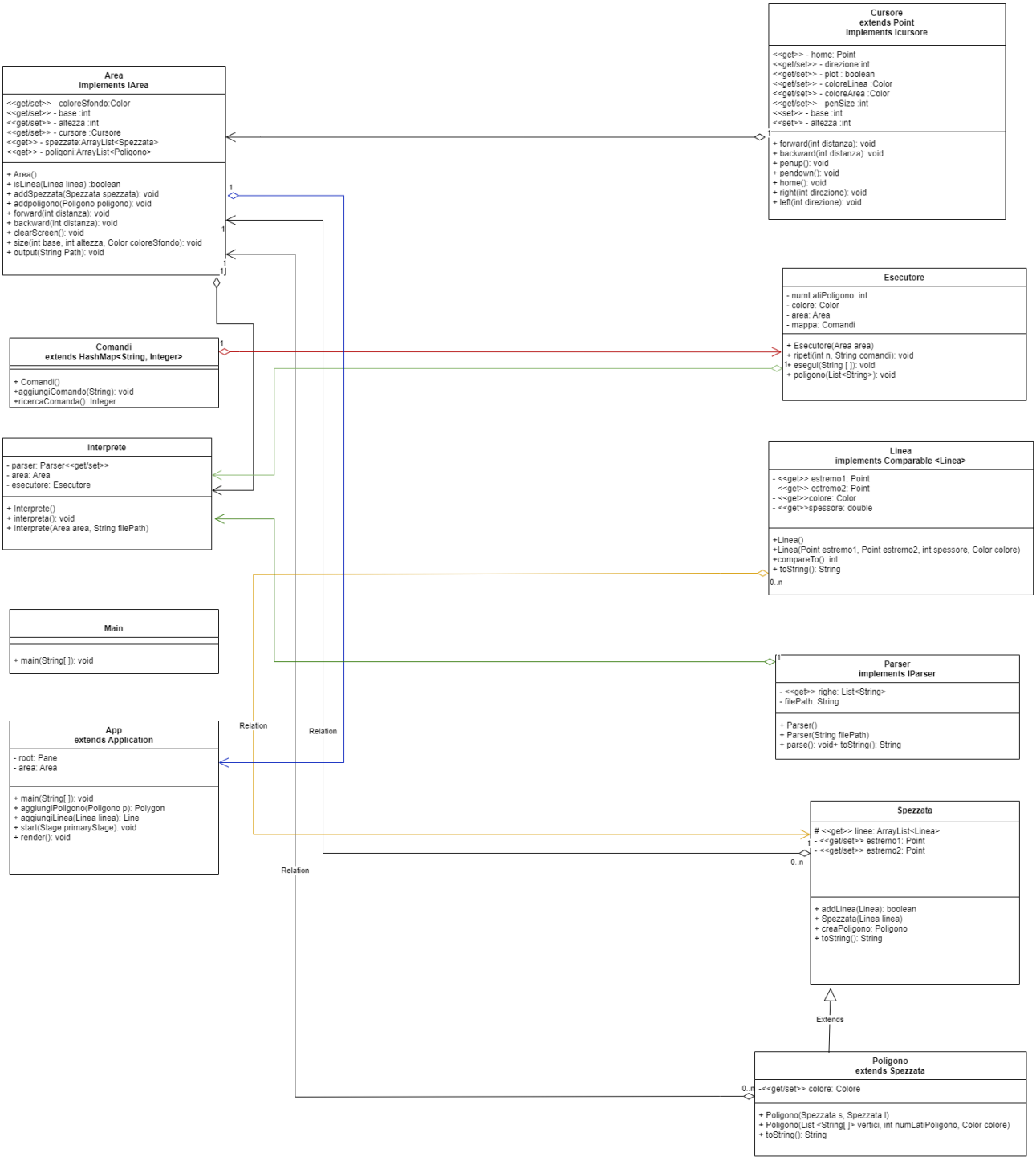
- static void main(Strings[] args): manda in esecuzione l'applicazione.
- void Polygon aggiungi Poligono (Poligono p): converte l'oggetto poligono in Polygon.
- void Line aggiungiLinea (Linea linea): converte l'oggetto linea in Line.
- void start (Stage primaryStage): manda in esecuzione i comandi del file logo e li visualizza.
- void render (): converte tutte le spezzate e i poligoni di area in oggetti di JavaFx e li aggiunge al Pane.

Getters/Setters

La classe App non presenta né getters né setters.

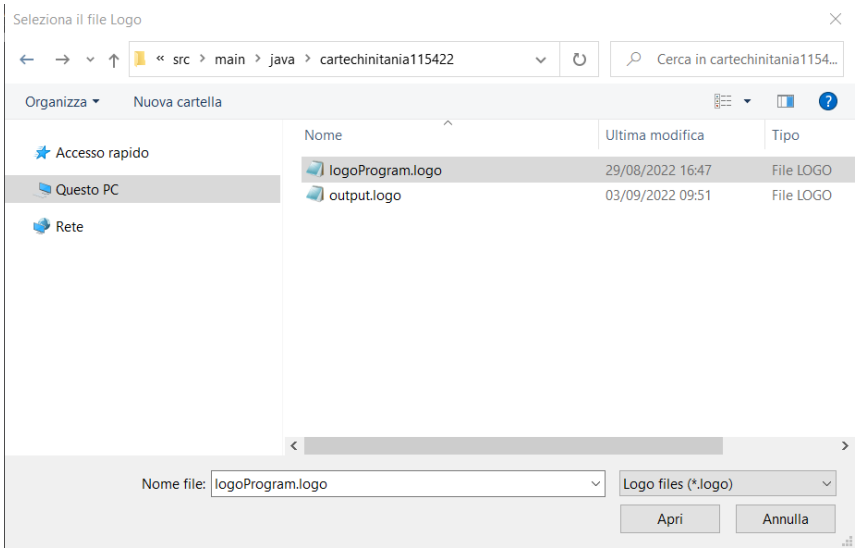
UML

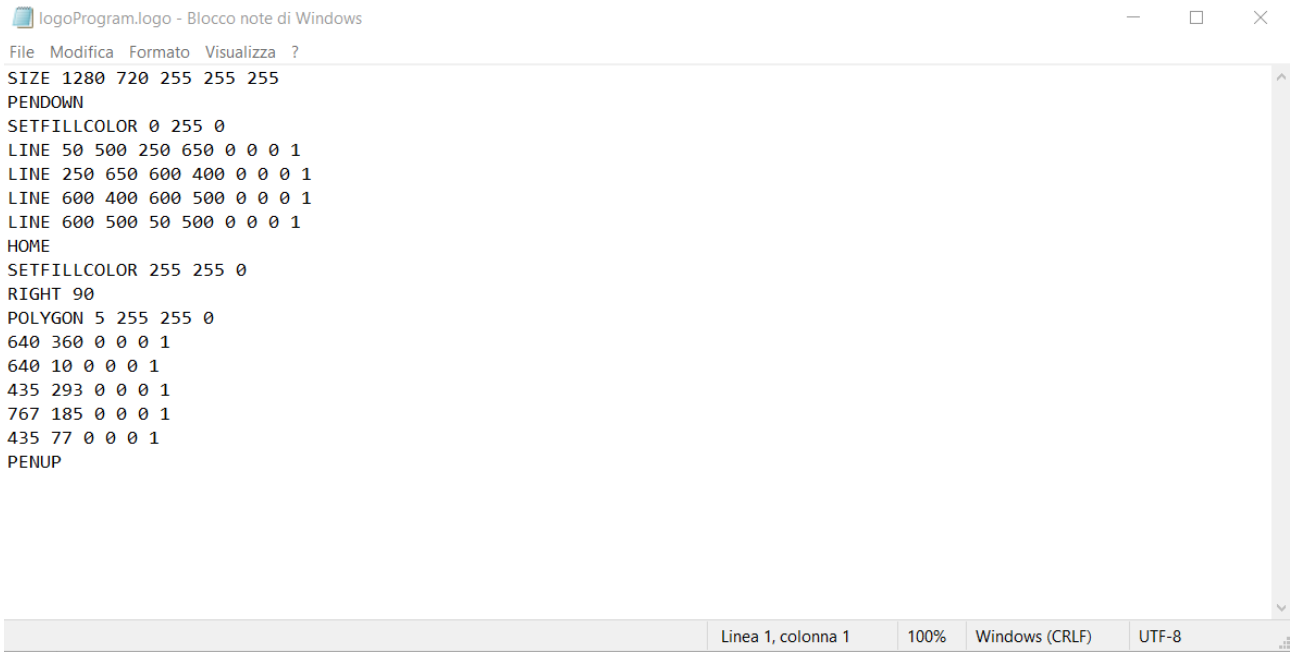
Link UML.drawio



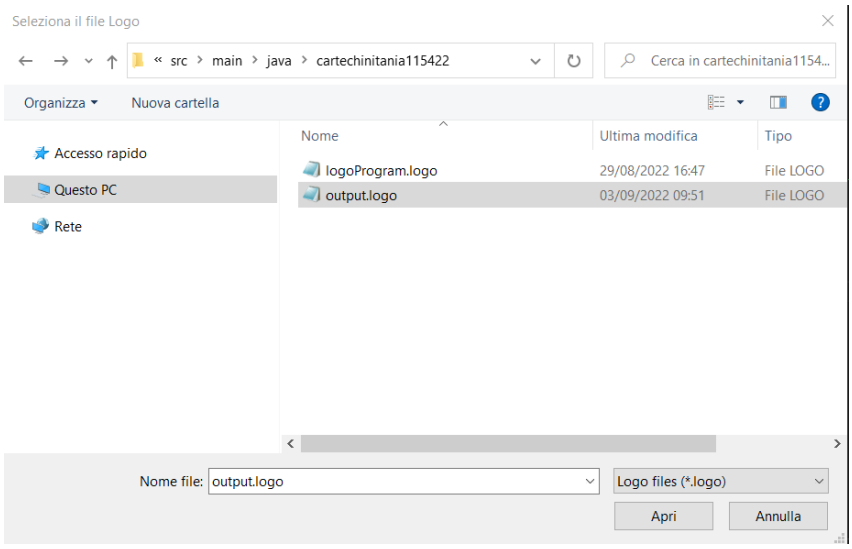
ESEMPI

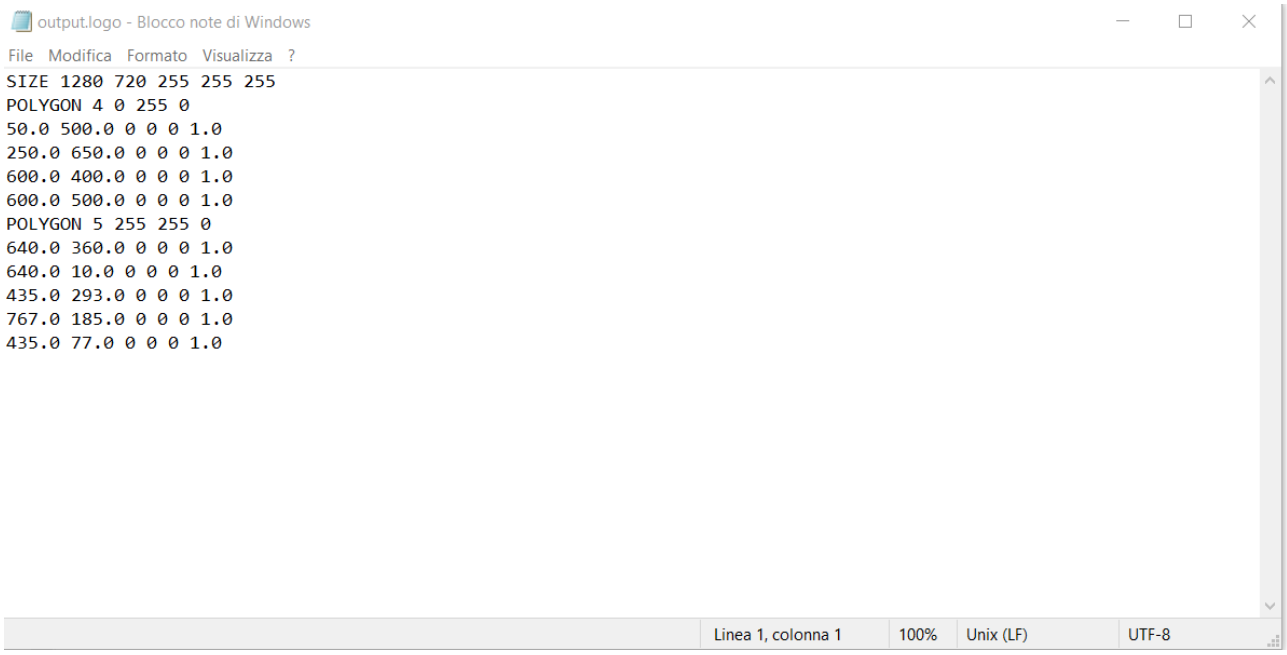
1) Selezione il file di input (logoProgram.logo) con estensione.logo





2) Salvataggio del file di output





3) Visualizzazione a video delle figure geometriche lette dal file.