SWARM INTELLIGENCE LIBRARY

ACO algorithm

I metodi rossi sono quelli che andranno inseriti nella funzione `setup()` in Processing, quelli verdi nel metodo `draw()`.

Position
Entità che descrive lo spazio in cui si muovono le formiche. Serve per sapere lo "stato" di una posizione, ovvero se vi è cibo e la quantità di feromone.

Space
Entità che si occupa di gestire lo spazio in cui si muovono le formiche.

`Space (Papplet sketch)` → costruttore della classe Space.

Type	Name and Description
`void`	`initSpace(Position [][] matrix, int spaceWidth, int spaceHeight)` metodo per inizializzare lo spazio in cui si andranno a muovere le formiche. • le variabili `spaceWidth` e `spaceHeight` sono larghezza e altezza dello spazio che si vuole definire. • `matrix` è una matrice di tipo position che verrà inizializzata all'interno della funzione.

Food
Entità che serve per gestire le fonti di cibo.

`Food (PApplet sketch, int anthillx, int anthilly)` → costruttore della classe Food

Type	Name and Description
`void`	`setFoodParameter(int foodSource, int foodPerSources, int foodMaxDistance, int foodMinDistance)` • la variabile `foodSources` serve per indicare quante risorse di cibo si vuole posizionare sullo spazio. • `foodPerSources` serve per descrivere quanto cibo deve esserci per ogni risorsa di cibo. • `foodMaxDistance` e `foodMinDistance` servono per indicare

	rispettivamente la distanza massima e minima del cibo rispetto al formicaio
void	initFood(Position[][] matrix, int spaceWidth, int spaceHeight) • variabili spaceWidth e spaceHeight sono larghezza e altezza dello spazio creato. • matrix è la matrice di spazio in cui si muovono le formiche
void	drawFood(int foodColorR, int foodColorG, int foodColorB) metodo che serve per mostrare a video le risorse di cibo, prende come parametri i tre interi che descrivono il colore che si decide di utilizzare

Pheromone

Entità che gestisce il feromone rilasciato dalle formiche

Pheromone (Papplet sketch) → costruttore della classe Pheromone

Type	Name and Description
void	setPheromoneParameter(int maxPheromone, float evapRatePheromone, float diffRatePheromone, float alpha, float beta) • maxPheromone è la variabile che descrive la quantità massima consentita di feromone che può trovarsi in una posizione. • evapRatePheromone serve per indicare il coefficiente di evaporazione del feromone • diffRatePheromone è il coefficiente per la diffusione del feromone • alpha e beta sono parametri costanti che vengono usati per il calcolo delle probabilità di scelta percorso.
void	initPheromone(Position[][] matrix, int spaceWidth, int spaceHeight) • variabili spaceWidth e spaceHeight sono larghezza e altezza dello spazio creato. • matrix è la matrice di spazio in cui si muovono le formiche
void	drawPheromone()

	metodo che serve per mostrare a video il feromone rilasciato dalle formiche

Anthill

Entità che gestisce il formicaio

`Anthill(PApplet sketch, int anthillx, int anthilly, Pheromone object)` → costruttore della classe anthill

Type	Name and Description
`void`	`releaseAnts(int antReleaseRate, int maxAntLife, int maxAnts)` metodo che serve per rilasciare le formiche dal formicaio. • `antReleaseRate` è l'intero che indica quante formiche vogliamo rilasciare alla volta (non può essere maggior di maxAnts) • `maxAntLife` è la vita massima che si vuole assegnare ad una formica (quante iterazioni di ciclo dell'algoritmo ACO resisterà) • `maxAnts` è il numero massimo di formiche che vogliamo nella nostra applicazione
`void`	`moveAntsForFood(int timeGearing, Food object)` metodo che implementa l'algoritmo ACO, muove le formiche verso il cibo • `timeGearing` è la variabile che indica quante formiche muovere (consigliato che sia pari a maxAnts)
`void`	`drawAnthill(int anthillColorR, int anthillColorG, int anthillColorB)` metodo che serve per mostrare a video il formicaio, prende come parametri i tre interi che descrivono il colore che si decide di utilizzare
`void`	`drawAnts(int antColorR, int antColorG, int antColorB, int antColorWithFoodR, int antColorWithFoodG, int antColorWithFoodB)` metodo che serve per mostrare a video le formiche, prende come i parametri 6 interi, i primi 3 per indicare il colore delle formiche che non hanno cibo, gli ultimi 3 per settare il colore delle formiche che hanno trovato cibo e stanno quindi ritornando al formicaio.

BOIDS algorithm

Boids
Entità che si occupa di gestire tutti gli stormi di uccelli che si andranno a creare.

`Boid(PApplet sketch)` → costruttore classe Boids

`void`	`addBoid()` funzione che aggiunge un nuovo boid allo sketch
`void`	`addObstacle()` funzione che aggiunge un nuovo ostacolo allo sketch
`String`	`getTool()` metodo che serve per ottenere la modalità in cui ci troviamo: • modalità "**boids**" vuol dire che possiamo aggiungere un nuovo boid allo sketch • modalità "**avoids**" vuol dire che possiamo aggiungere un ostacolo allo sketch • modalità "**erase**" vuol dire che possiamo eliminare oggetti dallo sketch
`void`	`setTool(String tool)` metodo che serve per settare il tipo di modalità che ci serve: • modalità "**boids**" se vogliamo aggiungere un nuovo boid allo sketch • modalità "**avoids**" se vogliamo aggiungere un nuovo ostacolo allo sketch • modalità "**erase**" se vogliamo cancellare oggetti dallo sketch

`void`	`setupWalls()`
	
	`metodo che mostra a video un muro sulla parte inferiore e superiore dello sketch, i boids hanno l'obiettivo di non scontrarsi`

`void`	`setupCircle()` metodo che mostra a video un cerchio al centro dello sketch, i boids hanno l'obiettivo di non scontrarsi contro esso

`void`	`setOption_friend()` funzione che si occupa della regola “allineamento”.
`void`	`setOption_crowd()` funzione che si occupa della regola “separazione”
`void`	`setOption_cohese()` funzione che si occupa della regola “coesione”
`void`	`setOption_avoid()` funzione che si preoccupa di far evitare ai boids di scontrarsi contro gli ostacoli
`void`	`setOption_noise()` funzione che consente lo spostamento iniziale di un singolo boid quando è alla ricerca di uno stormo a cui aggiungersi
`void`	`recalculateConstants()` calcola i parametri costanti per la realizzazione dell’algoritmo.