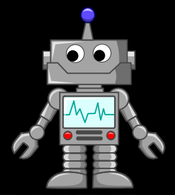
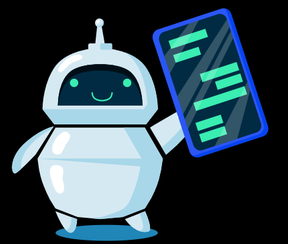
MICROROBOT MAZE EXPLORER

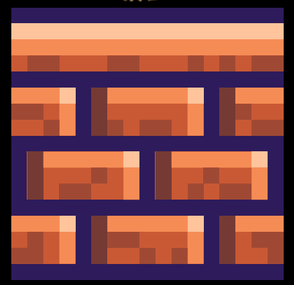
DIJKSTRA EDITION

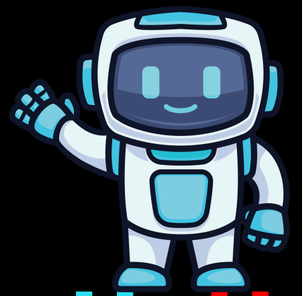


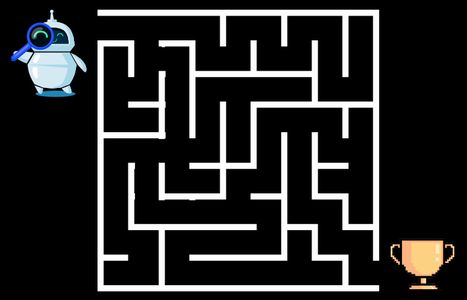


ANTONIO IANNONE 0124002543

FRANCESCO PIO GRAVINO 0124002703







PRESENTAZIONE DEL PROGETTO

Microrobot Maze Explorer: Dijkstra Edition è un coinvolgente gioco in cui il giocatore sfida se stesso a guidare un microrobot attraverso un intricato labirinto. Con l'aiuto dell'intelligenza artificiale basata sul celebre algoritmo di Dijkstra, il microrobot deve individuare l'uscita del labirinto, evitando le mura interne ed esterne del percorso e gli oggetti colorati nascosti lungo il cammino. Il giocatore può personalizzare l'esperienza scegliendo uno dei tre livelli di difficoltà disponibili. Attraverso una registrazione che include nome e cognome, ogni utente può sfidare se stesso a migliorare le proprie abilità.

START

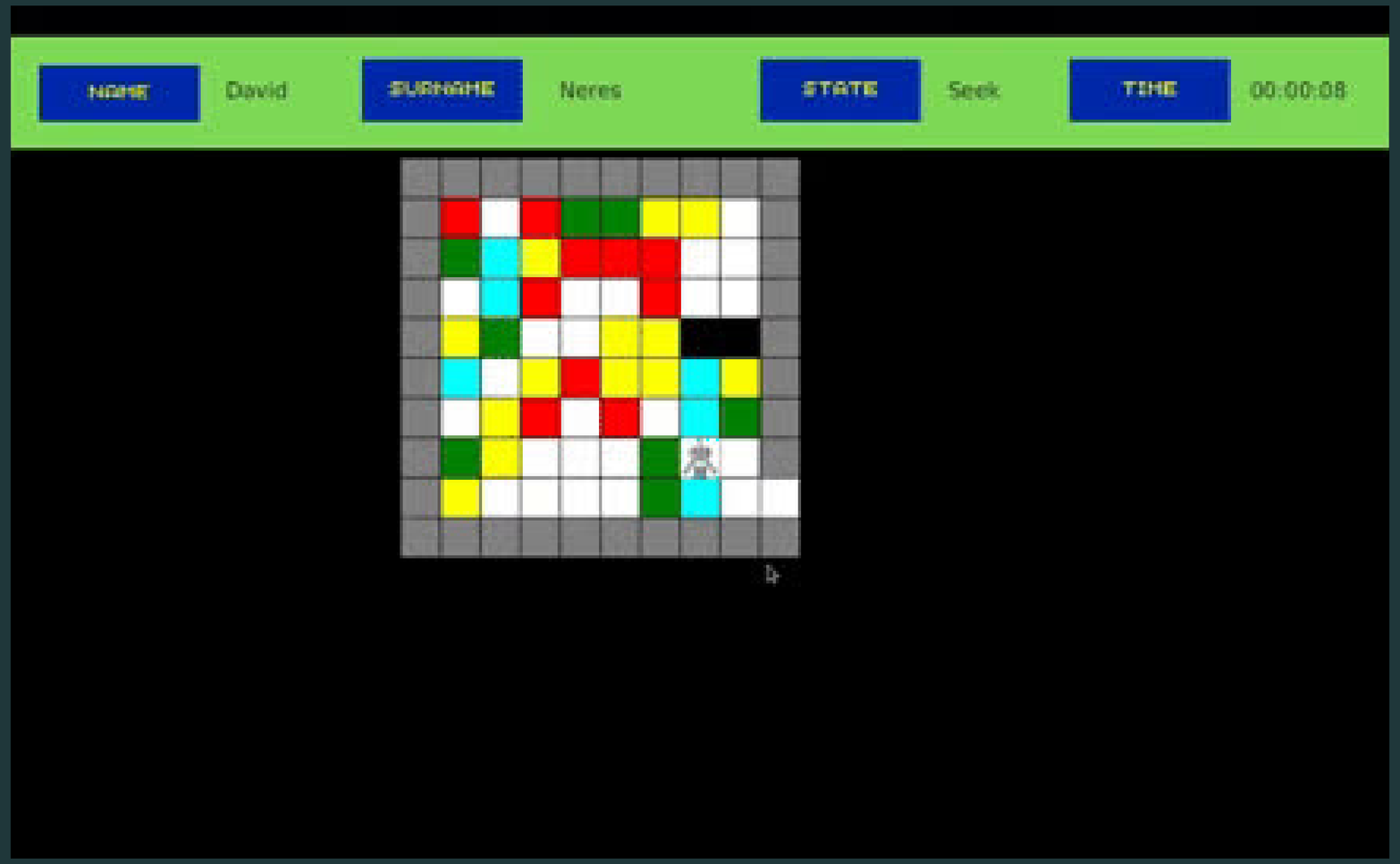
SCHERMATA INIZIALE

Link al repository GitHub del progetto per maggiori informazioni

Chiudere la schermata

Schermata per la selezione del livello di difficoltà e dei dati personali

LET'S PLAY



SCALIAMO LE CLASSIFICHE

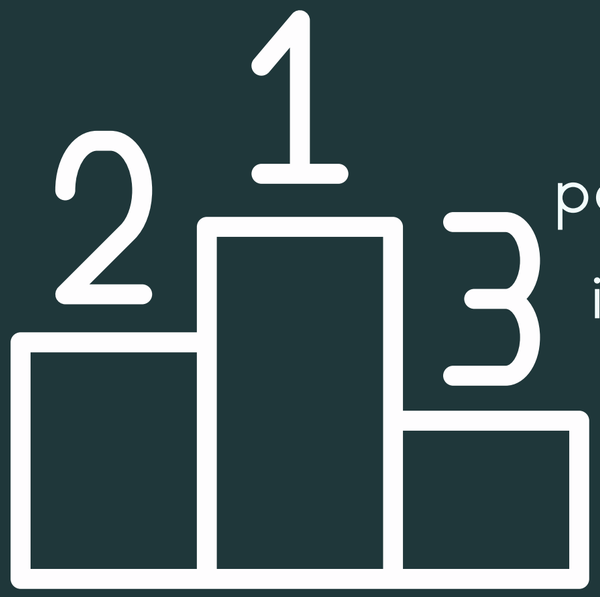
MICROROBOT MAZE EXPLORER : DIJKSTRA EDITION

YOUR TOP 5 SCORES

1. Daniel Ricco 00:00:13
2. Peppe Col 00:00:13
3. Kekko Tech 00:00:13
4. Pippo Benza 00:00:30
5. Eduardo Mid 00:00:28

FILE

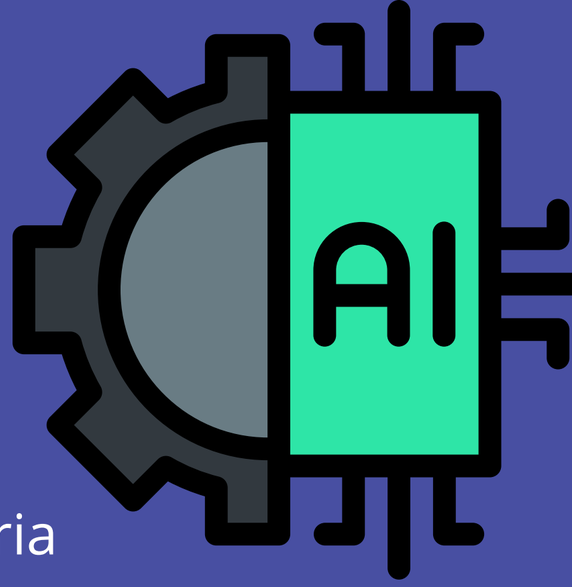
USER



BEST_RESULT

Per soddisfare la necessità di conservare la classifica dei giocatori e i relativi dettagli di completamento del labirinto, si è deciso di utilizzare un sistema basato su file. Questa scelta è stata motivata dalla semplicità e dall'efficienza nella gestione di dati strutturati in un contesto di applicazione locale, evitando la complessità aggiuntiva di un database. I file permettono di archiviare e recuperare le informazioni in modo rapido e diretto, garantendo una soluzione leggera e facilmente integrabile nel progetto.

dijkstra algorithm



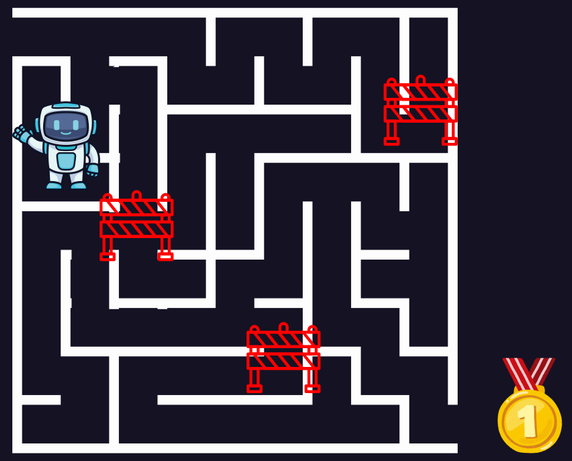
L'algoritmo di Dijkstra è un algoritmo utilizzato nell'informatica per trovare il cammino minimo in un grafo pesato. Appartiene alla categoria degli algoritmi di ricerca dei percorsi ottimali ed è stato progettato per calcolare i percorsi più brevi da un nodo sorgente a tutti gli altri nodi di un grafo.

Questo algoritmo si basa su un approccio greedy, che seleziona iterativamente il nodo con la distanza minima ancora non visitato e aggiorna le distanze dei nodi adiacenti. Grazie alla sua efficienza e semplicità, l'algoritmo di Dijkstra è ampiamente utilizzato in applicazioni come la navigazione, le reti di telecomunicazione e l'ottimizzazione dei percorsi.

LabyrinthMind

MICROROBOT

L'algoritmo di Dijkstra è stato adattato per risolvere il problema della ricerca del percorso ottimale nel labirinto da parte del microrobot. Il microrobot utilizza l'algoritmo per calcolare il cammino minimo verso l'uscita, tenendo conto delle caratteristiche del labirinto come ostacoli e celle accessibili.



DIJKSTRA EDITION

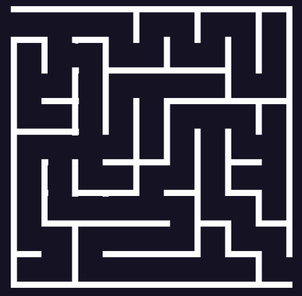
DESIGN PATTERN

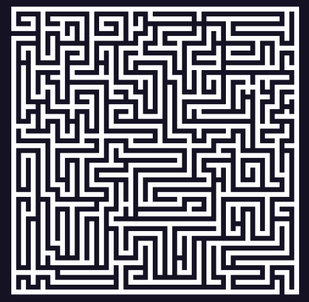
Factory Method	€3.90
Strategy	€4.90
Observer	€3.90
State	€2.90
Proxy	€3.10

FACTORY METHOD

Il Factory Method pattern è stato utilizzato nel progetto per gestire la creazione di diverse istanze di labirinti con diverse difficoltà in modo flessibile e senza accoppiamento eccessivo tra le classi , in particolar modo consente di creare diverse varianti di labirinti (facili, medi, difficili)

senza dover modificare il codice esistente. Le sottoclassi come EasyMaze, MediumMaze, e HardMaze forniscono implementazioni specifiche del metodo generateMaze(), che è il Factory Method, per creare labirinti con diverse configurazioni e difficoltà.

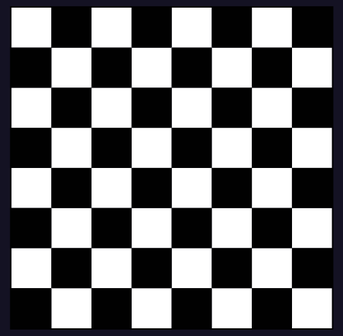




DIJKSTRA EDITION

STRATEGY

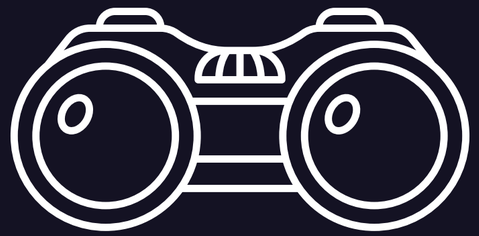
Il pattern Strategy è stato utilizzato per separare l'algoritmo di movimento del microrobot (strategia) dalla classe principale Microrobot, consentendo di cambiare dinamicamente la strategia di movimento senza modificare il codice del microrobot stesso.



DIJKSTRA EDITION

OBSERVER

I pattern Observer è stato adottato per permettere la notifica agli osservatori (observer) sugli aggiornamenti dello stato del gioco senza accoppiare strettamente i soggetti osservabili con gli osservatori stessi.



DIJKSTRA EDITION

STATE

Il pattern State è stato implementato per gestire il comportamento dinamico del microrobot in base alle diverse situazioni del labirinto. Grazie a questo pattern, il robot può adattarsi automaticamente agli ostacoli e alle caselle colorate che compaiono, modificando la sua strategia e le sue azioni in modo flessibile senza rendere il codice rigido o complesso.

DIJKSTRA EDITION

PROXY

il Pattern Proxy è stato implementato per gestire l'accesso al file della classifica in modo controllato ed efficiente. La classe Classification funge da proxy, intermediando le operazioni di lettura e scrittura del file. Grazie a questo pattern, viene garantita la corretta gestione del file (creazione, aggiornamento, ordinamento dei dati)