



# CdL in Informatica - A.A. 2022 - 2023

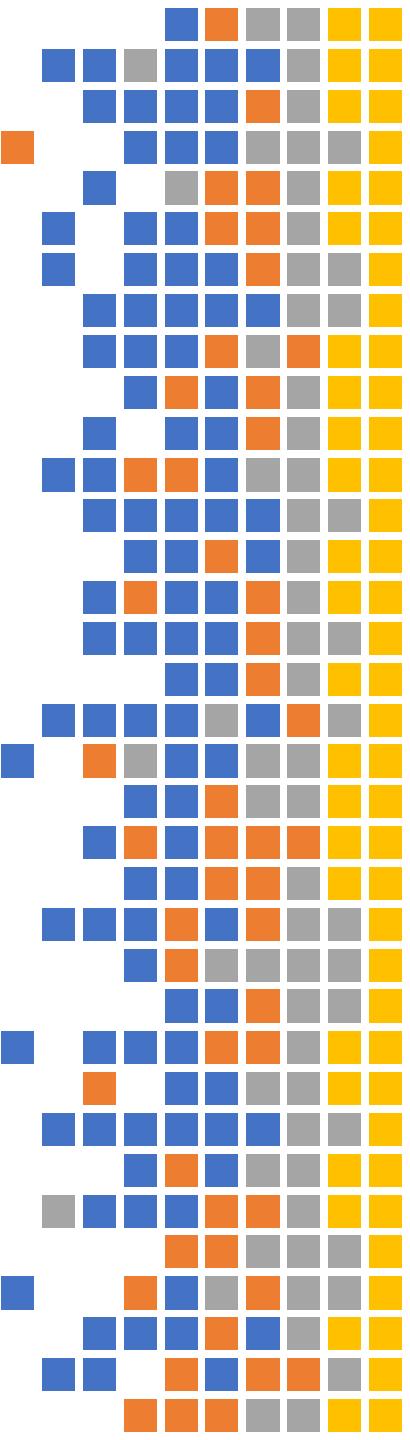
# Laboratorio di Programmazione 1

Progetto finale  
Specifiche

**Andrea Loddo**

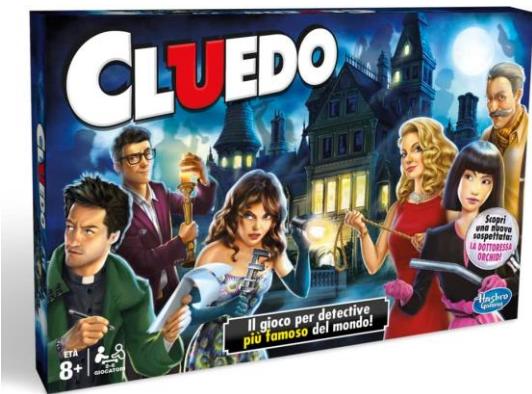
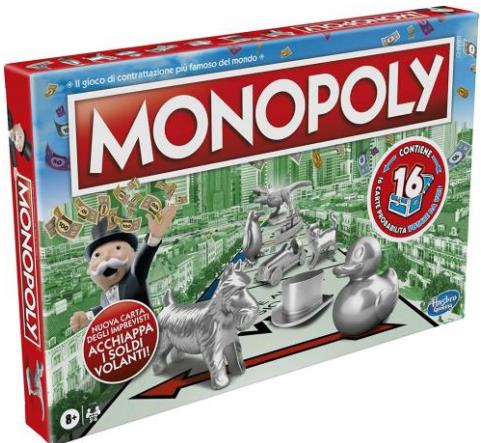
Luca Zedda

Federico Meloni



# Il progetto negli anni

Storicamente il progetto di PR1 prevedeva l'implementazione di un gioco di società  
Quest'anno riporteremo questa tradizione...

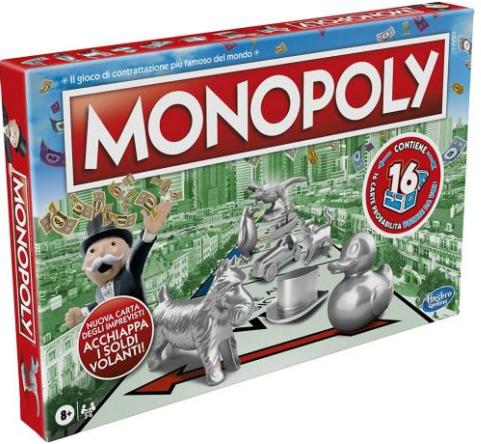


2016

2017

2018

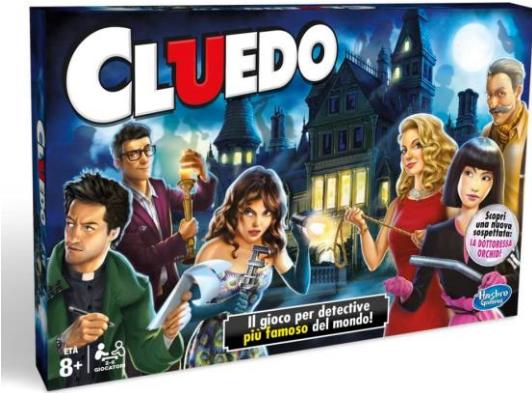
2019



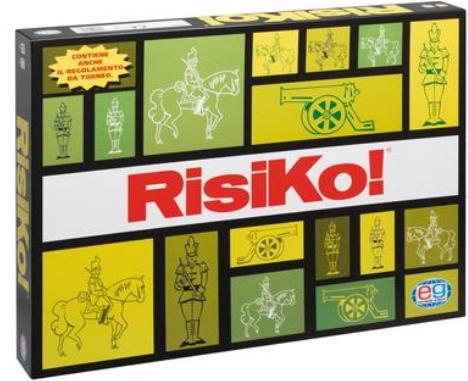
2016



2017



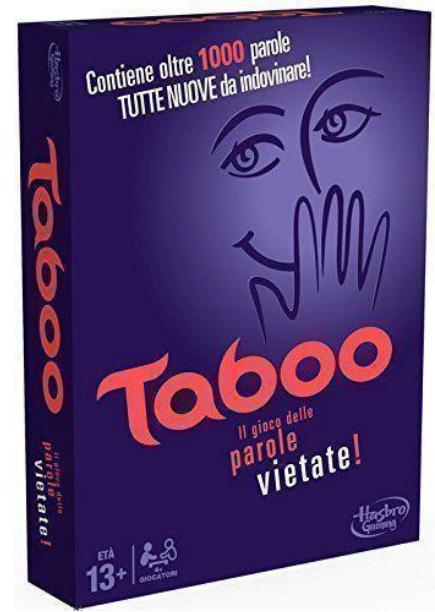
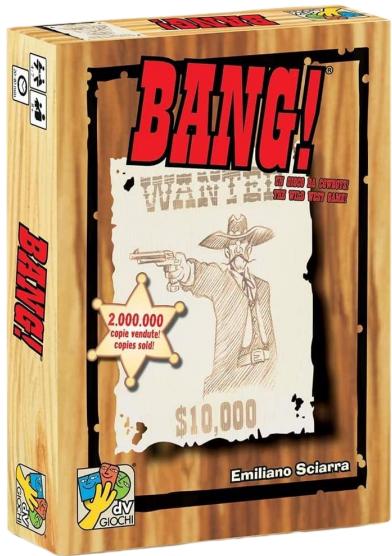
2018



2019

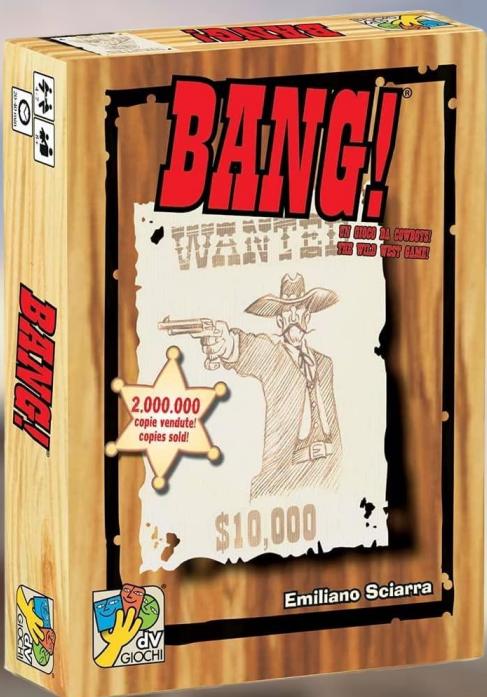
?

2022



Il progetto 2022 lo  
deciderete voi tra questi!

**AVETE INDOVINATO!**





# CdL in Informatica - A.A. 2022 - 2023

## Laboratorio di Programmazione 1

Progetto finale:



**Andrea Loddo**

**Luca Zedda**

**Federico Meloni**



# Organizzazione del progetto

---

- Il progetto è suddiviso in tre parti:
  - funzionalità BASE (20/30) consegnabile fino alle 23:59 del 23 maggio.
  - funzionalità MEDIE (30/30) consegnabile fino alle 23:59 del 31 luglio.
  - funzionalità AVANZATE (36/30) consegnabile fino alle 23:59 del 31 luglio.
- Il progetto può essere realizzato solo in modo incrementale: non possono essere implementate le funzionalità base e quelle avanzate tralasciando quelle medie.
- **Sufficienza:** voto  $\geq 15$ .
  - In caso di insufficienza si dovranno sostenere nuovamente le prove di laboratorio l'anno prossimo.
- Dopo la consegna del progetto, implementato nella difficoltà scelta, **si dovrà sostenere un colloquio orale**.
- Data e ora del colloquio verranno gestite a blocchi di consegna e/o concordati con lo studente.



# Organizzazione del progetto

---

- Il progetto è suddiviso in tre parti:
  - funzionalità BASE (20/30) consegnabile fino alle 23:59 del 23 maggio.
  - funzionalità MEDIE (30/30) consegnabile fino alle 23:59 del 31 luglio.
  - funzionalità AVANZATE (36/30) consegnabile fino alle 23:59 del 31 luglio.
- Deadline per non avere malus: 23:59 del 28 Febbraio 2023
- Dopo la 1<sup>a</sup> deadline: -0,4/30 per ogni settimana di ritardo sul voto finale (dal 1 marzo):
  - ogni mercoledì, a partire dal 1 marzo, verrà sottratto un malus di 0,4 dal voto del progetto.
- Tutto il progetto deve rispettare lo standard C99.
- È possibile utilizzare esclusivamente le librerie e i costrutti visti a lezione.



# Consegna

---

La consegna dovrà essere fatta tramite l'attività apposita su Teams corrispondente alla tipologia di progetto che avete scelto.

Le prime tre righe del file main dovranno rispettare il seguente formato:

//Nome: Alex Caruso (a.caruso6@studenti.unica.it)

//Matricola: 60/61/20204

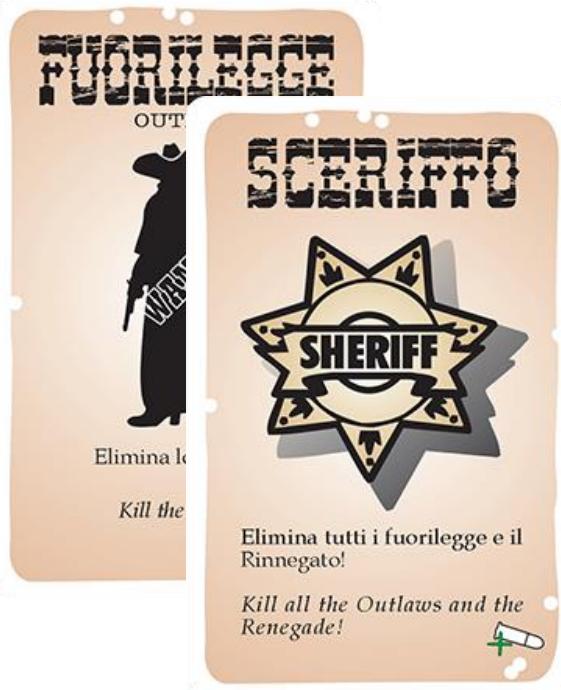
//Tipologia progetto: avanzato

Dopo la consegna attendete la convocazione per il colloquio orale.



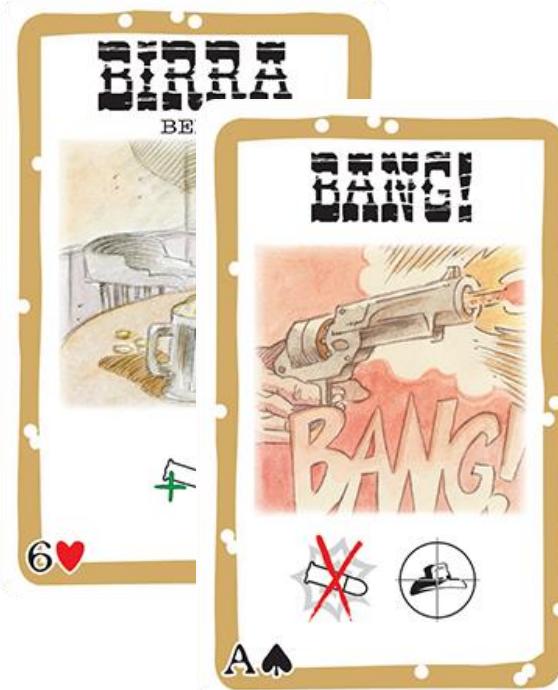
# Il gioco originale

Il gioco originale prevede:



Ruoli

NON per saranno i  
PERSONAGGI



Carte marroni



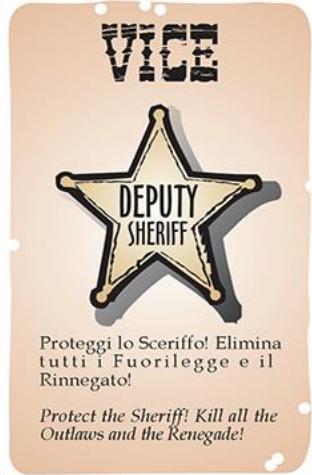
Carte blu



# Ruoli



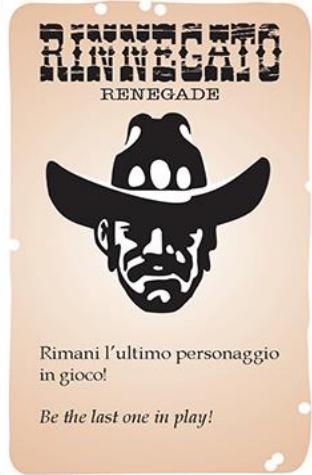
Il compito dello **Sceriffo** è quello di riportare l'ordine eliminando tutti i fuorilegge e rinnegato



I **Vice** aiutano e proteggono lo sceriffo, persegono gli stessi obiettivi anche a costo della vita



Il **Rinnegato** vuole diventare il nuovo sceriffo, per farlo deve rimanere l'unico in vita



I **Fuorilegge** vogliono eliminare lo sceriffo, ma non hanno scrupoli ad eliminarsi l'un l'altro

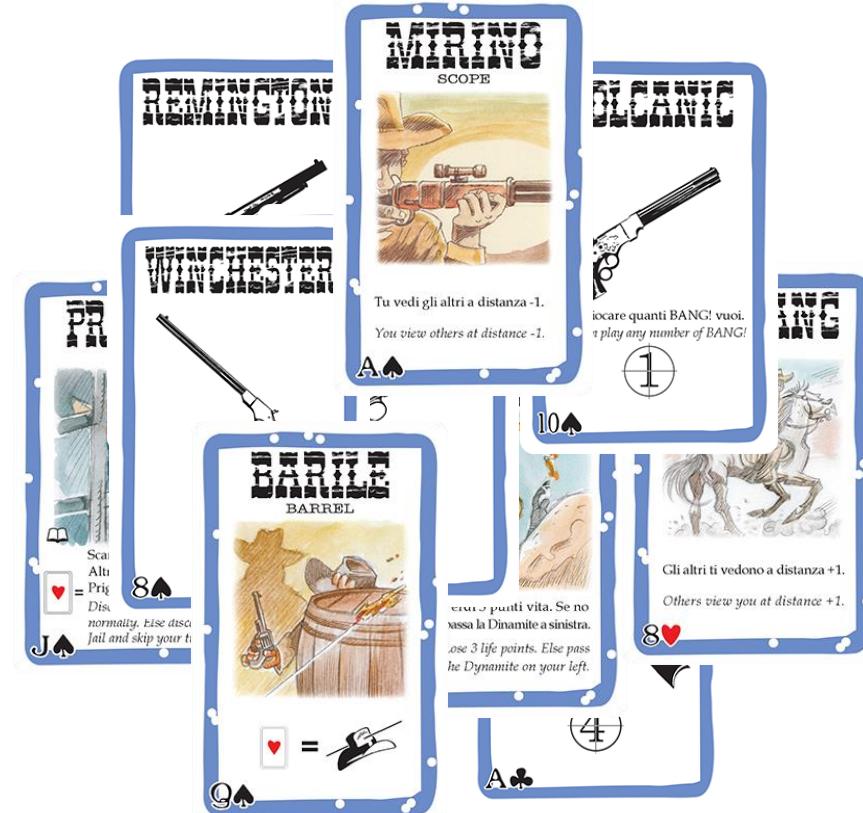


# BANG! Carte: le tipologie

Il gioco originale prevede:



Carte marroni: usa e getta



Carte blu: carte in gioco



# BANG!

## Carte: le tipologie

Il gioco da implementare prevede:



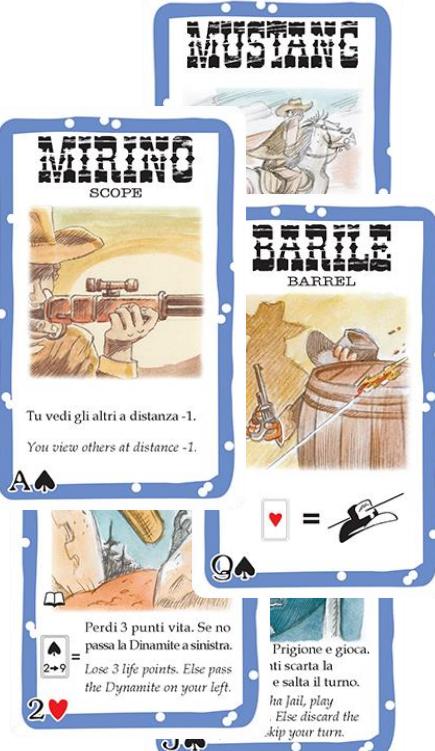
Tipo 1: istantanee



Tipo 2: istantanee speciali



Tipo 3: armi



Tipo 4: in gioco

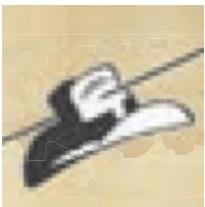


# Carte: comportamenti

Ogni carta, a meno di eccezioni, definisce il suo comportamento con i seguenti simboli:



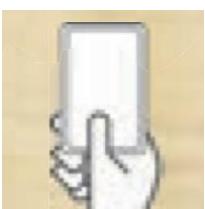
Attacco ad un giocatore, viaggia sempre in coppia con un simbolo che ne indica la gittata e/o i giocatori colpiti.  
Toglie un punto vita al giocatore che viene colpito.



Vale come Mancato!, quindi annulla l'effetto del simbolo precedente.



Recupera un punto vita, se non specificato diversamente solo chi gioca la carta beneficia dell'effetto.



Pesca una carta, se non specificato diversamente si pesca dal mazzo. Se accompagnato dal simbolo "un giocatore qualsiasi" prende il significato di "ruba una carta", che può essere o dalla mano (e viene scelta casualmente) oppure dalle carte in gioco. Le carte prese vengono aggiunte alla propria mano.



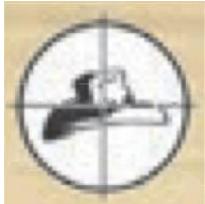
Accompagnato dal simbolo "un giocatore qualsiasi" permette di obbligare un giocatore a scartare una carta a caso dalla propria mano oppure una carta in gioco a scelta del giocatore che ha giocato la carta.



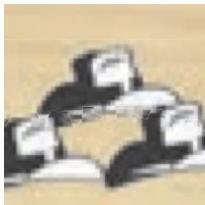
## Carte: bersagli

---

I comportamenti nella slide precedente possono essere accompagnati dai seguenti simboli che definiscono a chi si può applicare il loro effetto:



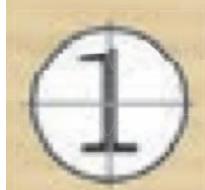
l'effetto si può applicare ad un qualsiasi giocatore che sia raggiungibile tramite la gittata dell'arma che possiede il giocatore che gioca la carta



l'effetto si applica a tutti i giocatori, a prescindere dalla distanza, escluso il giocatore che ha giocato la carta



l'effetto si può applicare ad un giocatore qualsiasi a scelta del giocatore che gioca questa carta, a prescindere dalla distanza

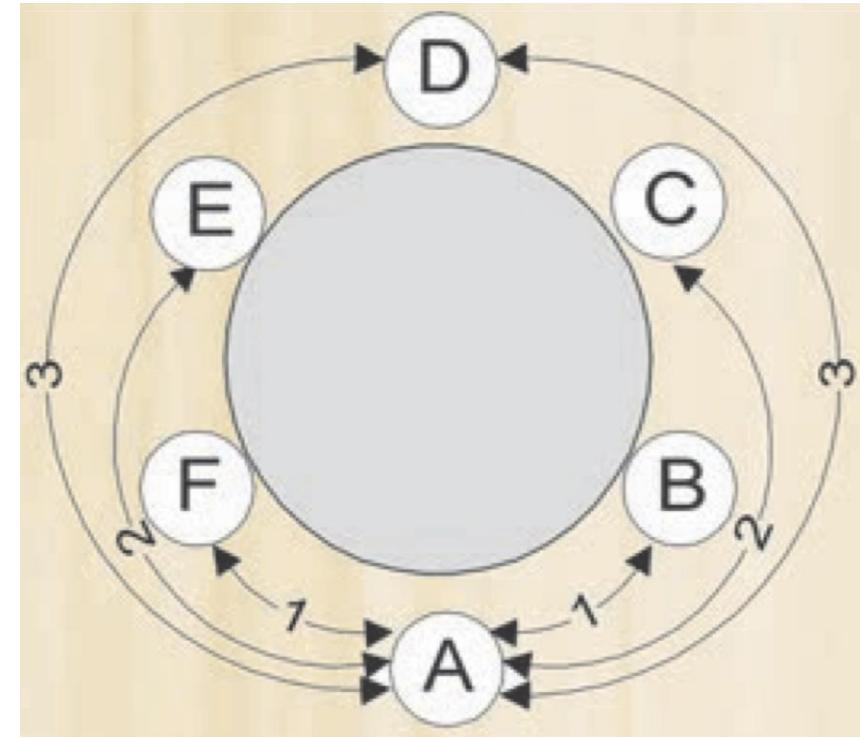


l'effetto si può applicare ad un qualsiasi giocatore che sia a distanza 1 dal giocatore che gioca la carta



# Calcolo della distanza ed eliminazioni

- Per distanza si intende la distanza che c'è tra un giocatore e l'altro in base alla loro disposizione.
  - Esempio: il giocatore A e il giocatore C distano 2.
- In caso di eliminazione di un giocatore, la distanza si riduce.
  - Esempio: se il giocatore B viene eliminato la distanza tra A e C si riduce a 1.
- Quando un giocatore viene eliminato viene svelato a tutti il suo ruolo, inoltre sono previsti i seguenti bonus/malus:
  - su ogni fuorilegge c'è una taglia sopra, chiunque ne elimini uno riceve una ricompensa pescando tre carte bonus (fuorilegge compresi);
  - se lo sceriffo elimina un vice sceriffo, è costretto a scartare tutte le carte nella sua mano come malus.





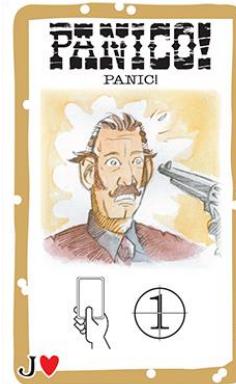
# BANG! Carte: tipologia 1



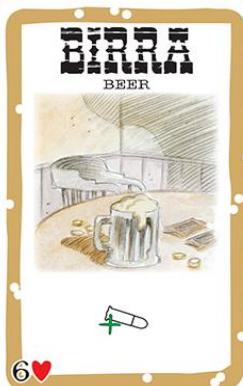
uno sparo a un giocatore raggiungibile dalla tua arma



pesca due carte dal mazzo



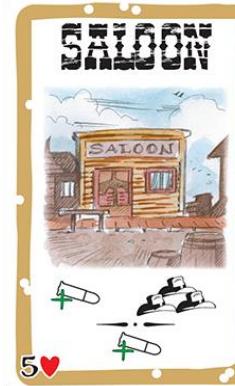
rubà una carta ad un giocatore a distanza uno



recupero di una vita, può essere utilizzata per salvarsi in extremis e non morire



spara a tutti i giocatori



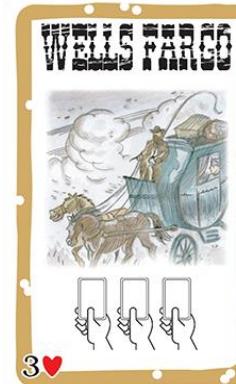
tutti i giocatori guadagnano un punto vita



fai scartare una carta ad un giocatore qualsiasi



annulla l'effetto di uno sparo



pesca tre carte dal mazzo



# BANG! Carte: tipologia 2



scelto un giocatore qualsiasi da sfidare a duello (a prescindere dalla distanza), il giocatore che ha giocato la carta e quello sfidato iniziano a scartare a turno carte BANG! partendo dal giocatore sfidato. Il primo che non può più scartare una carta BANG! perché le ha finite perde un punto vita.



viene rivelato dal mazzo di pesca un numero di carte pari al numero di giocatori ancora in vita e, a partire dal giocatore che ha giocato la carta e proseguendo in ordine di inserimento, ogni giocatore sceglie una carta da pescare fino ad esaurire le carte rivelate.



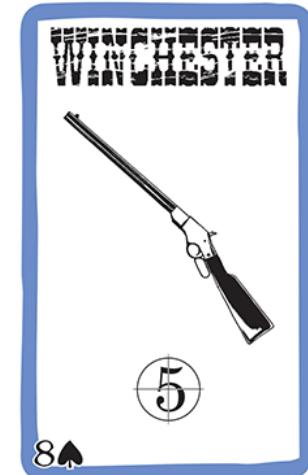
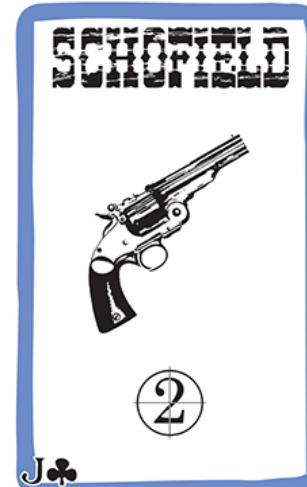
tutti i giocatori (escluso chi ha giocato la carta) sono costretti a scartare una carta BANG!, se non la hanno perdono un punto vita.



# Carte: tipologia 3

---

Le carte della tipologia 3 sono le armi, dove il numero indica la gittata dell'arma:

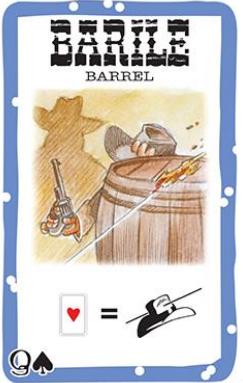


La Volcanic ha la particolarità di far sparare quanti BANG! si vuole in un turno.



# BANG!

## Carte: tipologia 4



Questa carta consente di estrarre quando si è il bersaglio di un BANG!:

- se esce Cuori sei Mancato! (come se avessi giocato un Mancato!);
- in caso contrario, non succede niente.

Es. supponiamo che tu abbia il Barile in gioco e sia il bersaglio di un BANG!, si estrae una carta: è un 4 di Cuori! Annulli quindi automaticamente il BANG!.

Se non esce una carta di Cuori, il Barile non ha effetto: hai comunque ancora la possibilità di giocare un Mancato! per annullare il BANG!



Gioca la Dinamite mettendola davanti a te, dove resterà innocua per un intero giro.

Quando inizi il tuo prossimo turno, e hai quindi la Dinamite già in gioco, prima ancora di pescare devi estrarre:

- se esce una carta tra il 2 ed il 9 di Picche, la Dinamite esplode! Scartala e perdi 3 punti vita;
- in caso contrario, passa la Dinamite al giocatore successivo (il quale all'inizio del proprio turno farà a sua volta lo stesso controllo, ecc.).

I giocatori continuano a passarsi la Dinamite, finché essa esplode, con gli effetti descritti, o viene presa o scartata da un Panico! o da Cat Balou.

Se la Dinamite è in gioco assieme alla Prigione, va verificata per prima la Dinamite. Se perdi punti vita (o esci dal gioco!) a causa dell'esplosione della Dinamite, questi danni non si considerano causati da alcun giocatore.

ESTRARRE: si pesca una carta dal mazzo di pesca per controllarne numero e seme, poi si scarta



# BANG! Carte: tipologia 4



Quando hai in gioco un Mirino vedi gli altri giocatori a una distanza diminuita di 1.

Gli altri invece continuano a vederti alla distanza normale. Distanze inferiori a 1 sono considerate uguali a 1.

Esempio. Nella figura della distanza (slide 16), se Andrea (A) avesse in gioco un Mirino, vedrebbe Beatrice (B) e Filippo (F) a distanza 1, Carlo (C) ed Emanuela (E) a distanza 1, Daniela (D) a distanza 2, mentre Andrea continuerebbe a essere visto alla distanza normale da tutti gli altri.



Quando hai in gioco un Mustang sei visto dagli altri a una distanza aumentata di 1.

Tu invece continui a vedere gli altri alla distanza normale.

Esempio. Nella figura della distanza, se Andrea (A) avesse in gioco un cavallo Mustang, Beatrice (B) e Filippo (F) lo vedrebbero a distanza 2, Carlo (C) ed Emanuela (E) a distanza 3, Daniela (D) a distanza 4, mentre Andrea continuerebbe a vedere gli altri alla distanza normale.



Gioca questa carta davanti a un giocatore a scelta (indipendentemente dalla distanza): lo metti in prigione! Se sei in prigione, per prima cosa al tuo turno devi "estrarre":

- se esce Cuori evadi: scarta la Prigione, e gioca il tuo turno normalmente;
- in caso contrario, scarta la Prigione e salta il turno.

Per il resto non cambia nulla: se sei in prigione resti un possibile bersaglio di BANG! e puoi difenderti usando carte in risposta, come Mancato! e Birra, se necessario. La Prigione non può essere giocata contro lo Sceriffo.

ESTRARRE: si pesca una carta dal mazzo di pesca per controllarne numero e seme, poi si scarta



# BANG! Carte: la disposizione



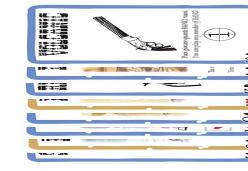
Federico  
**SCERIFFO**



Mano del  
giocatore



Pesca



Scarti



Carte in  
gioco



Luca  
**FUORILEGGE**





# Le strutture: il Giocatore

---

Per ogni giocatore sarà necessario salvare le seguenti informazioni:

- Nome utente (49+1 caratteri)
- Ruolo assegnatogli (corrispondenza numeri obbligatoria, nomi arbitrari)
  - 0: sceriffo
  - 1: vicesceriffo
  - 2: fuorilegge
  - 3: rinnegato
- Il numero pallottole (punti-vita)
- L'insieme delle carte nella propria mano (*vedi slide seguente*)
- L'insieme delle carte in gioco (*vedi slide seguente*)
- La distanza a cui può sparare i giocatori avversari (gittata)



# Le strutture: la Carta

---

Per ogni carta sarà necessario salvare le seguenti informazioni:

- **Nome della carta** (14+1 caratteri)
- **Tipologia della carta** (corrispondenza numeri obbligatoria, nomi arbitrari)
  - 1: ISTANTANEA
  - 2: INSTANTANEA\_SPECIAL
  - 3: ARMA
  - 4: EFFETTO
- **Il numero della carta** (tra 1 e 13)
- **Il seme della carta** (corrispondenza numeri obbligatoria, nomi arbitrari)
  - 0: CUORI
  - 1: QUADRI
  - 2: FIORI
  - 3: PICCHE



# Le strutture: il Mazzo

---

Per ogni mazzo sarà necessario salvare le seguenti informazioni:

- Tipologia del mazzo (corrispondenza numeri obbligatoria, nomi arbitrari)
  - 0: MAZZO\_PESCA
  - 1: MAZZO\_SCARTO
  - 2: GIOC\_MANO
  - 3: GIOC\_GIOCO
- Un intero N rappresentante il numero di carte all'interno
- Un array dinamico di N carte

I CAMPI DEL MAZZO VANNO DICHIARATI IN QUEST'ORDINE PER COMPATIBILITÀ CON I NOSTRI SALVATAGGI



# Fasi di gioco: fase 1 – preparazione (1/2)

---

- Si chiede all'utente il numero di giocatori, i nomi di ognuno e si assegna loro un ruolo casualmente secondo lo schema:
  - 4 giocatori: 1 sceriffo, 1 rinnegato, 2 fuorilegge
  - 5 giocatori: 1 sceriffo, 1 rinnegato, 2 fuorilegge, 1 vice sceriffo
  - 6 giocatori: 1 sceriffo, 1 rinnegato, 3 fuorilegge, 1 vice sceriffo
  - 7 giocatori: 1 sceriffo, 1 rinnegato, 3 fuorilegge, 2 vice sceriffo
- Ogni giocatore ha 4 punti vita e non rende pubblico il suo ruolo, tranne lo sceriffo che ha 5 punti vita ed è l'unico che rende il suo ruolo pubblico.
- Si carica il mazzo contenuto nel file mazzo\_bang.txt, dove ogni riga è formattata secondo il pattern descritto nella slide 32. Tutti gli insiemi di carte vanno allocati dinamicamente.
- Una volta caricato il mazzo questo va mescolato e successivamente si distribuisce ad ogni giocatore un numero di carte pari al suo numero di punti vita.  
Le carte distribuite finiscono tutte nella mano del giocatore.



## Fasi di gioco: fase 1 – preparazione (2/2)

---

- All'inizio della partita nessun giocatore ha carte in gioco
- A questo punto può cominciare il gioco, partendo dallo sceriffo, in ordine di inserimento.



## Fasi di gioco: fase 2 - turno (1/3)

---

All'inizio di ogni turno si dovrà stampare il numero del turno, il nome del giocatore e il suo ruolo.

Inoltre, all'inizio del turno va salvata la partita nell'apposito file binario di salvataggio (*vedi slide 36*).

Ogni turno si divide nei seguenti momenti:

1. effettuare eventuali azioni dettate dalle carte in gioco
2. pescare due carte: si pescano le prime due carte dalla cima del mazzo
3. giocare carte: si possono giocare carte a proprio vantaggio o contro gli altri, non si è obbligati a giocare carte e si possono giocare quante carte si vuole con le limitazioni:
  - si può giocare solo una carta col nome BANG! per turno
  - non si può avere più di una carta blu in gioco con lo stesso nome
  - si può avere in gioco una sola arma, quando si gioca una nuova arma si scarta la precedente
4. scartare carte in eccesso: quando si decide di passare il turno e si hanno più carte in mano rispetto al numero di punti vita (attuali), si devono scartare un numero di carte tali da averne in mano un numero pari al numero di punti vita



## Fasi di gioco: fase 2 - turno (2/3)

---

In ogni turno l'utente può scegliere se effettuare le seguenti operazioni:

1. Giocare delle carte
2. Vedere le carte che ha in mano
3. Vedere le carte che ha in gioco
4. Controllare a che distanza è rispetto agli altri giocatori (evidenziare lo sceriffo)
5. Controllare che carte in gioco hanno gli altri giocatori (evidenziare lo sceriffo)
6. Passare il turno
7. Chiudere il gioco

OPZIONALE: potete scegliere se implementare altre operazioni



## Fasi di gioco: fase 2 - turno (3/3)

---

Ogni giocatore, in assenza di altre armi in gioco, è sempre considerato come armato con una Colt. 45 che gli permette di colpire bersagli a distanza 1.

Ogni giocatore può avere zero o più carte in mano o in gioco, mentre le carte non distribuite sono a disposizione in un mazzo di pesca, comune a tutti i giocatori.

Tutte le carte scartate durante il gioco andranno invece in un mazzo di carte apposito, quello degli scarti, sempre comune a tutti i giocatori.

Esaurite le carte da pescare, il mazzo degli scarti deve essere mischiato e diventerà il nuovo mazzo di pesca.

Un giocatore non può avere più punti vita di quelli con cui ha iniziato il gioco.

Un giocatore, quando viene eliminato, scarta tutte le carte che ha in mano e in gioco nel mazzo degli scarti.



## Fasi di gioco: fase 3

---

I criteri di fine gioco sono i seguenti:

- viene **eliminato lo sceriffo**, se è rimasto vivo solo il **rinnegato** vince lui, altrimenti vincono i **fuorilegge**
- vengono eliminati tutti i **fuorilegge** e il **rinnegato**, la vittoria spetta a **sceriffo** e ai suoi vice

NB: se tutti i fuorilegge sono morti e muore lo sceriffo, ma ci sono ancora in vita dei vice, vincono comunque i fuorilegge e non il rinnegato.



# BANG! Carte: tipologie

---

- Il mazzo di carte si trova nel file "mazzo\_bang.txt", da noi fornito, e va caricato con una lettura da file.
- Le carte sono salvate in questo file con uno schema ricorrente.
- Prendiamo una generica riga di una carta Birra come esempio: **1 6 0 Birra**
- Dove:
  - in **rosso** troviamo la **tipologia** di carta
  - in **blu** troviamo il **numero** della carta
  - in **viola** troviamo il **seme** delle carte
  - in **verde** troviamo il **nome** della carta
- Ricorda: le carte sono divise in 4 tipologie:
  1. carte istantanee/marroni: hanno un effetto immediato sulla partita e utilizzano solo simboli standard
  2. carte istantanee/marroni: hanno un effetto immediato sulla partita ed è descritto sulla carta
  3. carte arma: cambiano la gittata di fuoco del giocatore
  4. carte effetto/blu: rappresentano bonus o malus per il giocatore a cui sono associate



# Implementazione: progetto BASE (max 20pt)

---

Implementazioni da realizzare nelle differenti versioni:

- BASE:
  - Il numero di giocatori è deciso a **runtime** e viene chiesto un **username** ad ogni giocatore.
  - Bisogna **implementare tutta la logica del gioco**, senza tener conto di bonus e malus legati ai ruoli che vengono eliminati (vedi slide 16) e tenendo conto **solo delle carte di tipologia 1 e 3 (Volcanic esclusa)**.
  - Il mazzo va caricato dal file **mazzo\_bang.txt** fornito.
  - Con l'implementazione solo di queste due tipologie di carta è possibile ignorare l'informazione di seme e numero per ogni carta.
  - All'inizio di ogni turno si deve salvare la partita in un file 'savegame.sav' strutturato come mostrato in slide 36.
  - Questo salvataggio, se presente, deve poter essere caricato ad ogni avvio del programma.



# Implementazione: progetto MEDIO (max 30pt)

---

Implementazioni da realizzare nelle differenti versioni:

- MEDIO:

Specifiche BASE +

- Si devono implementate tutta la logica del gioco, tenendo conto di bonus e malus derivanti dal giocatore eliminato (vedi slide 16) e tutte le carte (sarà necessario tener conto dei numeri e dei semi delle carte).
- Si devono permettere salvataggi multipli permettendo all'utente di scegliere il nome del file di salvataggio (nel caso sia già presente verrà sovrascritto) e di conseguenza scegliere quale file caricare, gestendo gli eventuali casi di input inconsistenti da parte dell'utente.
- Generare un file log.txt in cui viene scritta la cronologia della partita

TURNO 1: Riccardo pesca un Bang! e un Gatling

TURNO 1: Riccardo gioca un Bang! contro Federico

TURNO 1: Federico utilizza un mancato

TURNO 2: Andrea estrae un 2 di cuori per la Prigione e può giocare il turno

...



# Implementazione: progetto AVANZATO (max 36pt)

---

Implementazioni da realizzare nelle differenti versioni:

- AVANZATO:

Specifiche MEDIO +

- Dare la possibilità di eseguire la partita in modalità automatica, il tester dovrà solamente specificare il numero di giocatori e la partita dovrà essere eseguita da giocatori CPU, salvando ad ogni turno la partita.
- In questa modalità dovranno essere mostrati in chiaro tutti i dati della partita, le carte distribuite, i ruoli, ecc.
- Dati i ruoli, le scelte di ogni giocatore dovranno essere fatte in modo autonomo, seguendo comunque la logica del gioco. Ovvero: tutti i giocatori, comandati dal computer, dovranno cercare di vincere la partita.
- Maggiore sarà la fedeltà dell'intelligenza artificiale rispetto al comportamento umano, più alta sarà la valutazione. Quindi: maggiore sarà la completa casualità nelle scelte, minore sarà il voto, maggiore sarà la logica (con un senso e una strategia dietro), maggiore sarà il voto.
- Dopo ogni partita aggiornare un file con le statistiche che permetta di stabilire il numero di vittorie di ogni squadra, tenendo il conto separato tra modalità manuale e in modalità automatica. Si possono aggiungere un numero a piacere di informazioni a questo file. Visualizzare tale file prima di ogni nuova partita. Gestire il caso in cui il file non venga trovato o non sia presente.
- È necessario consegnare un file di statistiche con dei dati già presenti insieme al progetto.



# File di salvataggio

---

Il file di salvataggio è un file binario "savegame.sav", che contiene:

- un intero N indicante il numero di giocatori partecipanti alla partita (da 4 a 7).
- Per ognuno degli N giocatori:
  - Un blocco rappresentante il giocatore.
  - Un array dinamico di C carte appartenenti al Mazzo rappresentante la mano del giocatore.
  - Un array dinamico di D carte appartenenti al Mazzo rappresentante le carte in gioco del giocatore.
- Un intero indicante il prossimo giocatore a dover giocare.
- Un blocco rappresentante il mazzo di pesca.
  - Un array dinamico di P carte appartenenti al Mazzo di pesca.
- Un blocco rappresentante il mazzo degli scarti.
  - Un array dinamico di S carte appartenenti al Mazzo degli scarti.

È possibile memorizzare ulteriori informazioni a piacimento alla fine del file di salvataggio. Gestire il caso in cui queste non siano presenti.



# BEST PRACTICES ai fini della valutazione

---

Buone norme di programmazione:

- Macro maiuscole
- Nomi delle variabili camelCase o snake\_case (scegliete uno stile e mantenetelo)
- Blocchi di codice non troppo lunghi (nel caso valutare la suddivisione in subroutine)
- Suddividere le varie funzionalità in apposite subroutine
- Suddividere le varie funzionalità in appositi file .c/.h
- No variabili globali
- Evitare break e continue
- Ricordare i controlli sui puntatori e di liberare la memoria in caso di allocazione dinamica (DI TUTTO QUELLO ALLOCATO)
- Ricordare il controllo sui file e di chiuderli sempre
- L'uso del goto sarà un goto a riseguire il corso nel 23/24



# BEST PRACTICES ai fini della valutazione

---

## Stile, indentazione e leggibilità:

- Rendere il proprio codice leggibile
- Indentare correttamente i vari blocchi di codice
- Utilizzare uno stile e seguirlo per tutto il progetto
- Utilizzare nomi per variabili, subroutine etc.. che abbiano un senso con quello che fanno
- Commentare tutti i passaggi non ovvi
- Prima dell'implementazione di ogni subroutine descriverla brevemente con un commento

Per ogni subroutine è suggerito utilizzare il seguente formato di commenti Doxygen (due asterischi in apertura di commento):

```
/** La funzione randRange restituisce un numero intero casuale
 * compreso tra min e max. Va chiamata la srand() nel main prima
 * della sua prima invocazione.
 *
 * @param min il numero minimo
 * @param max il numero massimo
 * @return un numero intero casuale compreso tra min e max
 */
int randRange(int min, int max) {
    return rand() % (max - min + 1) + min;
```