Compito vacanze: Tombola.

## Appunti:

* Più cartelle in contemporanea.
* Ad ogni turno viene estratto un numero.
* Effettuare l’analisi di tutte le cartelle all’uscita del numero.
* Verificare le righe se hanno trovato più di un numero.
* Memorizzare il primo che ha fatto ambo, terzina, quartina e quintina.
* Verificare se il nuovo numero successivo generato è già stato trovato.
* Statistiche, come vittorie etc.
* I numeri vanno da 1 a 99.
* Possibilità di generare una nuova cartella all’inizio e quindi vederla e scegliere se prenderne un’altra.
* Ci sono 3 righe da 5 numeri, quindi in totale 15 numeri per cartella che non devono ripetersi e 4 caselle vuote a riga, verificare che le colonne non siano completamente vuote (non dovrebbero esserci 3 elementi in colonna vuote).

# Generazione della cartella:

1. Genero in colonna i numeri, la prima colonna avrà valori da 1 a 9, la seconda da 10 a 19, la terza da 20 a 29 e così via.
2. Una volta generati, posiziono per ogni riga 4 spazi vuoti (che saranno riconosciuti dal carattere di controllo 0) in modo casuale.
3. Una volta posizionati tutti i valori, verifico per ogni colonna se i valori dei 3 vettori (uno per ogni riga) ci sono alla stessa posizione 3 valori 0 ossia quello di controllo che corrisponde con lo spazio vuoto, nel caso positivo, rigenero una nuova cartella da zero.
4. Le singole cartelle sono salvate tramite struct apposite, dove ci sono 3 vettori di interi (uno per riga) e dei booleani che ricordano se sono avvenuti **delle ambo terna quaterna cinquina,** numeri trovati alle righe corrispondentiall’utente e delle variabili globali che verificano se qualcuno ha già fatto delle azioni.

## Gioco:

* Chiedo quante persone vogliono giocare.
* Una volta dette le persone, creo una cartella per il primo utente e la mostro, chiedo se va bene e se accetta passo al successivo altrimenti ricomincio la generazione.
* Una volta che tutti gli utenti hanno generato e scelto le loro cartelle, inizio a giocare.
* All’inizio del gioco, genero un numero casuale e in un vettore di booleani, setto su true o 1 il valore alla posizione del numero generato, che sarà da 0 a 90 (srand() % 90 quindi da 0 a 89 effettivamente), in modo che nella generazione successiva quando faccio una verifica se alla posizione appena controllato è su true (numero già uscito), ne creo uno nuovo fino a quando non ne trovo uno valido, il vettore quindi dei numeri avrà tutti valori di default 0 o false.
* Una volta generato il numero, inizio un ciclo di verifica per tutte le cartelle una alla volta, controllo quindi le righe per vedere se il numero è disponibile. Nel caso vero si modificano le variabili delle struct.
* Continuando nel gioco, nelle singole verifiche controllo se l’utente (nel caso ottenga 2 numeri sulla stessa riga) ha già fatto, prima a livello globale, qualcuno l’ambo/terna/quaterna/cinquina, nel caso nessuno l’abbia fatto, modifico una variabile globale per tenere traccia di cosa è stato fatto, e anche nella struct personale salvo che cosa si ha fatto.
* Se un utente alla fine trova tutti i numeri della sua cartella, comunico la fine della partita e la sua vittoria, rendendo pubblici tutti i valori di tutte le cartelle, in modo da mettere dei [] negli spazi vuoti e {} nei numeri trovati.

# Statistiche:

Utilizzando la seconda funzione, è possibile simulare una partita velocemente con delle impostazioni scelte, ma scegliendo anche la cartella.

La terza funzione permetti di fare la stessa cosa ma senza scegliere le cartelle.

Le statistiche comprendono diversi dati:

* Numero turni trascorsi per fare Ambo.
* Numero turni trascorsi per fare terna.
* Numero turni trascorsi per fare Quaterna.
* Numero turni trascorsi per fare Cinquina.
* Numero turni trascorsi per fare Tombola.
* Numero di numeri trovati da ogni singolo utente.
* Media numeri trovati dagli utenti.

Tra le varie funzioni di statistica, saranno creare altre varianti che simulano più partite, con e senza visualizzazione del riepilogo in terminale, ed eventualmente salvataggio su FILE.