Laboratorio di Calcolo, Canale Pet-Z
Prova di esame, appello gennaio 2026

Nome: _____ Cognome: _____
Matricola: _____ □ Ritirata/o

1. Il tempo a disposizione è di 3 ore. Sono ammessi libri di testo, prontuari, appunti. Non si può parlare con nessuno, utilizzare cellulari/tablet/laptop, pena l'annullamento del compito.
2. Il programma va scritto e salvato esclusivamente sul computer del laboratorio, a cui si deve accedere utilizzando come username **studente** e come password **informatica**.
3. **Tutti i file vanno salvati in una cartella chiamata `LCSR_COGNOME_NOME` nella home directory**, dove `NOME` e `COGNOME` indicano rispettivamente il tuo nome e cognome. Ad esempio lo studente *Nicolò De Rossi* deve creare una cartella chiamata `LCSR_DEROSSI_NICOLO` contenente tutti i file specificati nel testo. **Tutto ciò che non si trova all'interno della cartella suddetta non verrà valutato.**
4. Consegnare il presente testo indicando nome, cognome e numero di matricola (vedi sopra), barrando la casella "Ritirata/o" se ci si vuole ritirare.

Simulazione del gioco del ponte di vetro semplificato

In questa versione semplificata del gioco del *ponte di vetro* (ispirato alla serie *Squid Game* di Netflix) un concorrente deve attraversare un ponte di vetro formato da una sequenza di 25 moduli. In ogni modulo una sola delle due lastre è sicura, mentre l'altra è fragile e si rompe se calpestata. Le lastre sono indistinguibili tra loro ad occhio umano. Ad ogni turno, il concorrente sceglie di saltare su una delle due lastre di un modulo: se finisce su una lastra fragile, il concorrente precipita dal ponte e viene eliminato dal gioco; se invece sceglie la lastra temprata il concorrente supera il modulo e passa al successivo.

Il gioco termina quando il concorrente cade da una lastra fragile oppure quando riesce a superare l'ultimo modulo del ponte.

► Prima parte (20 punti):

Si scriva un programma in `C` di nome `cognome_nome.c` che simuli il gioco come segue:

1. Tramite una direttiva del precompilatore, definire il numero di moduli `MODULI` (25).
2. Si rappresenti il ponte mediante un **array unidimensionale** `ponte` di interi. In ciascuna posizione dell'array viene memorizzata esclusivamente l'informazione relativa alla lastra sicura:
   - valore `0`: la lastra **fragile** è a sinistra e la lastra **sicura** è a destra;
   - valore `1`: la lastra **fragile** è a destra e la lastra **sicura** è a sinistra.

3. Implementare una funzione `crea(...)`, di tipo e argomenti opportuni, per generare la configurazione iniziale del ponte, riempiendo l'array `ponte[]`; la lastra di vetro temprato può trovarsi con uguale probabilità a destra o a sinistra.

4. Si implementi una funzione `visualizza(...)`, di tipo e argomenti opportuni, che visualizzi sullo schermo la configurazione del ponte, stampando una riga per ciascun modulo:

   - il simbolo `+` nella posizione della lastra sicura;
   - il simbolo `0` nella posizione della lastra fragile.

   La prima coppia di lastre deve essere stampata in fondo.

5. implementare una funzione `giocaPartita(...)`, di tipo e argomenti opportuni, per implementare il gioco secondo lo schema descritto in seguito; la funzione deve restituire il numero `nSalti` di salti consecutivi riusciti durante la partita.

   (a) Utilizzando un ciclo `while` con opportune condizioni simulare il percorso di un concorrente che parte dal primo modulo e procede finché non cade oppure supera l'ultimo modulo.

   (b) Ad ogni salto il concorrente (ossia il programma) sceglie casualmente con uguale probabilità se saltare sulla lastra di sinistra o di destra. La scelta viene effettuata chiamando una funzione `scelta(...)`, che restituisce il valore `0` o `1`, corrispondenti rispettivamente alla scelta della lastra di sinistra o di destra.

   (c) Contare il numero `nSalti` di salti consecutivi riusciti prima della caduta oppure dell'attraversamento del ponte.

► **Seconda parte (8 punti):**

In questa seconda parte si vuole studiare la distribuzione del numero di salti consecutivi riusciti prima della caduta.
Modificare il programma come segue:

1. chiedere all'utente di inserire il numero `nPartite` di partite da simulare, compreso tra 1 e 10000, assicurandosi che il numero inserito sia valido;

2. ripetere la simulazione del gioco `nPartite` volte utilizzando sempre la stessa configurazione del ponte;

3. per ciascuna partita contare il numero `nSalti` di salti consecutivi riusciti prima della caduta;

4. aggiornare un array `conteggio`, di tipo e dimensioni opportune, che memorizzi quante partite terminano con esattamente `nSalti` di salti consecutivi riusciti;

5. al termine di tutte le partite:

   (a) stampare sullo schermo il valore massimo del numero di salti consecutivi riusciti.

   (b) creare un file di testo `salti.txt` contenente due valori per ciascuna riga:

   - il numero `nSalti` di salti consecutivi riusciti;

- la frequenza (conteggio diviso per il numero totale di partite) di partite terminate con quel valore di `nSalti`.

Bonus (2 punti) solo se tutto il resto implementato: calcolare la media pesata $\langle$`nSalti`$\rangle$ dei salti consecutivi riusciti definita come:

$$\langle \texttt{nSalti} \rangle = \frac{\sum_0^{\text{MODULI}} i \cdot N_i}{\sum_0^{\text{MODULI}} N_i} \tag{1}$$

dove N_i è il numero di partite con i salti consecutivi riusciti. Stampare il valore medio sullo schermo con 1 cifra decimale con un breve messaggio informativo

---

► **Esercizio in `Python` (4 punti):**

Scrivete uno script Python `grafico.py` per graficare l'istogramma della frequenza in funzione del numero di salti consecutivi riusciti, utilizzando i dati contenuti nel file `salti.txt`.
Il grafico dovrà:

- contenere una legenda
- opportuni titoli sugli assi
- essere salvato e **visibile** in un file `grafico.png`.