Laboratorio di Calcolo per Fisici,
Appello di settembre, AA 2023/24

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

Lo scopo della prova è scrivere un programma che sia in grado di riconoscere i palindromi in un insieme di parole generate casualmente.

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. Il programma va creato all'interno di un file `NOME_COGNOME.c`, salvato su una cartella di nome `NOME_COGNOME`, creata all'interno della home directory. **Tutto ciò che non si trova all'interno della cartella suddetta non verrà valutato.** Seguite le istruzioni date dai docenti in aula per fare un backup periodico della cartella sul server dell'aula.
4. Consegnare il presente testo indicando nome, cognome e numero di matricola (vedi sopra), barrando la casella "Ritirata/o" se ci si vuole ritirare.

► **Esercizio in `C`:** Lo scopo di questa esercitazione valutata è scrivere un programma che generi casualmente una serie di parole (stringhe) di caratteri di lunghezza fissata, e riconosca tra queste la presenza di eventuali *palindromi*.
Una frase o una parola si definiscono *palindrome* se risultano identiche quando vengono lette da sinistra a destra o da destra a sinistra. Ad esempio, le parole italiane `oro`, `anna`, `otto`, `effe`, `radar` sono palindrome. Si scriva un programma in `C` che generi una serie di stringhe di caratteri di lunghezza fissata (parole), e che conti quante di esse sono palindrome. Per semplicità si considerino solo stringhe di caratteri minuscoli ('a', 'b', 'c', ecc); nel programma verrà richiesto di generare stringhe con tutti i caratteri dell'alfabeto inglese, che sono 26, o con un sottoinsieme di essi, definito dalla variabile `nchar`, che rappresenta la dimensione dell'alfabeto da utilizzare. Quindi, se `nchar` è uguale a 26, le stringhe potranno contenere tutte le lettere minuscole dell'alfabeto, se `nchar` è uguale a 2, verranno usati solo i caratteri 'a' e 'b', se `nchar` è uguale a 5 verranno usati i caratteri 'a', 'b', 'c', 'd' e 'e' e così via.
Per testare il codice che scriverete, considerate che la probabilità che una parola di lunghezza N generata casualmente con un alfabeto di M caratteri sia palindroma è

$$p = \frac{1}{M^{\lfloor N/2 \rfloor}},$$

dove $\lfloor N/2 \rfloor$ è uguale a $N/2$ se N è pari, o $(N-1)/2$ se N è dispari. Ad esempio, per $N = 5$, $\lfloor N/2 \rfloor = 2$ e quindi $p = 1/M^2$.
Il programma che scriverete deve rispettare le seguenti specifiche:

- La lunghezza `LEN` delle stringhe da generare e il numero `NWORDS` di parole da generare vengono specificati tramite direttive `define`. In particolare, `LEN` va fissato a 5, mentre il valore di `NWORDS` va inizialmente fissato a 1000. **Nota Bene:** nella seconda parte `NWORDS` diventa una variabile.

- Il programma chiede all'utente di inserire la dimensione dell'alfabeto da utilizzare per generare ciascuna stringa, `nchar`, compresa tra 2 e 26 (estremi inclusi); la richiesta viene iterata finché non viene inserito un numero valido.
- Il programma genera quindi una serie di `NWORDS` parole di lunghezza fissata, e conta quante di esse sono palindrome.
- In particolare, una funzione `genstring()`, di tipo e argomenti opportuni, genera una sequenza di `LEN` caratteri, scelti tra gli `nchar` possibili, e la salva su di una stringa `parolad[]`.
- Una funzione `reversestring()`, di tipo e argomenti opportuni, inverte la sequenza di caratteri contenuta nella stringa `parolad[]` e la salva in una nuova stringa `parolar[]`.
- Una funzione `ispalindrome()` confronta le due stringhe `parolad[]` e `parolar[]` e restituisce 1 se le due stringhe sono identiche, e 0 altrimenti.
- Per le prime dieci parole generate il programma stampa su schermo un riepilogo, secondo il formato:

```
parola ( 1) =  nmwwq  reverse( 1) = qwwmn palindromo = 0
parola ( 2) =  jnfnj  reverse( 2) = jnfnj palindromo = 1
...
```

- Alla fine del programma, viene stampato su schermo un messaggio di riepilogo generale, contenente il numero totale di parole generate (`NWORDS`), la lunghezza delle parole (`LEN`), la dimensione dell'alfabeto `nchar`, e il numero e la frequenza di parole palindrome generate.

Una volta verificato che il programma funzioni, modificarlo come segue:

- Invece di fissare con una macro il valore di `NWORDS`, il programma gira automaticamente per valori di una variabile `nwords` pari a 1000, 2000, 4000, 8000, 16000, 32000 e 64000 parole, e salva su di un file di due colonne di nome `palindromi_nchar.dat` (esempio: palindromi_26.dat se `nchar == 26`) il numero di parole generate e la *frequenza* con cui sono state generate parole palindrome. Quest'ultima va stampata con quattro cifre dopo la virgola. **Suggerimento:** potete costruire il nome del file di output utilizzando il valore di `nchar` direttamente all'interno del programma:

```
char nomefile[100];
sprintf(nomefile, "palindromi_%d.dat", nchar);
FILE *out = fopen(nomefile, ...
```

- Si faccia girare il programma con un numero di caratteri `nchar` pari a 2, 10 e 26, creando un file di dati `palindromi_nchar.dat` per ciascun valore di `nchar`.

► **Esercizio in Python:** A partire dai file `palindromi_nchar.dat` creare uno script `python` chiamato `NOME_COGNOME.py` che riporti su di un grafico tre curve che rappresentano la frequenza delle parole palindrome in funzione del numero di parole generate per ciascun valore di `nchar`, salvandolo sul file `NOME_COGNOME.png`. Il grafico dovrà riportare una legenda ed opportuni label agli assi.