



Esercitazione 10 (11-25)



10.1 Funzioni per cilindro (C++)







- Definire una funzione per il calcolo del volume di un cilindro
 - Parametri: raggio e altezza (`float`)
 - Risultato: volume (`float`)
- Definire una funzione per il calcolo della superficie di un cilindro
 - Parametri: raggio e altezza (`float`)
 - Risultato: superficie (`float`)
- In una funzione `main`
 - Chiedere all'utente raggio e altezza di un cilindro
 - Chiamare le due funzioni e mostrare i risultati

Definire $PI = 3.14159$, oppure usare `acos(-1)`, dalla libreria `<cmath>`



10.2 Classe per cilindro (C++)

- Definire una classe dei cilindri
 - Parametri del costruttore: raggio e altezza (**float**)
 - Metodo che restituisce il volume
 - Metodo che restituisce la superficie
- In una funzione **main**
 - Chiedere all'utente raggio e altezza di un cilindro
 - Chiamare i due metodi e mostrare i risultati

Name of the solid	Figure	Volume	Lateral/Curved Surface Area	Total Surface Area
Cuboid		lbh	$2lh + 2bh$ or $2h(l+b)$	$2lh+2bh+2lb$ or $2(lh+bh+lb)$
Cube		a^3	$4a^2$	$4a^2+2a^2$ or $6a^2$
Right circular cylinder		$\pi r^2 h$	$2\pi r h$	$2\pi r h + 2\pi r^2$ or $2\pi r(h+r)$
Right circular cone		$\frac{1}{3}\pi r^2 h$	$\pi r l$	$\pi r l + \pi r^2$ or $\pi r(l+r)$
Sphere		$\frac{4}{3}\pi r^3$	$4\pi r^2$	$4\pi r^2$
Hemisphere		$\frac{2}{3}\pi r^3$	$2\pi r^2$	$2\pi r^2+\pi r^2$ or $3\pi r^2$

10.3 Conteggio caratteri (C++)

- Definire una funzione
 - Parametri: due stringhe
 - Risultato: quanti caratteri della prima stringa sono presenti anche nella seconda?
- Esempio
 - Parametri: "Ciao, Python!", "aeiou"
 - Risultato: 4

Si può definire prima una funzione ausiliaria booleana `char_in_string`, per controllare se una stringa contiene un certo carattere



10.4 Configurazioni di simboli (C++)

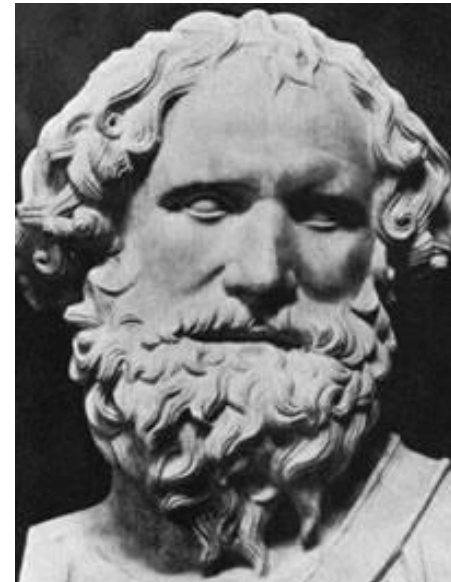
- Scrivere una funzione per generare tutte le configurazioni di lunghezza 3
 - Parametri: `string` contenente l'insieme dei simboli validi (presenti su ogni ruota)
 - Risultato: `vector` di `string` (configurazioni)
- Possibile approccio: 3 cicli `for` annidati
- Per test, fornire in input le vocali "AEIOU", per ottenere:
 - ["AAA", "AAE", "AAI", "AAO", "AAU", "AEA", "AEE", "AEI", "AEO", "AEU", "AIA", ...]



10.5 Crivello di Eratostene (C++)

- Trovare tutti i numeri primi fino ad n , scelto dall'utente
- Algoritmo
 - Inizializzare un `vector<bool>` con n valori `true`
 - Per ciascun numero x tra 2 ed n (ancora a `true` nella lista)...
 - Mettere a `false` i multipli di x (escluso x stesso)
 - Alla fine, gli elementi rimasti a `true` corrisponderanno ai numeri primi cercati

Usare un `vector<int>` è problematico ed inefficiente



10.6 Hitori, vicolo cieco (Py)

- Metodo booleano **wrong**, senza parametri
- Controlla se una delle annotazioni inserite impedisce la soluzione del gioco
 - *Non si può risolvere il gioco senza rimuovere una annotazione*
 - Es. Due celle annerite contigue
 - Es. Due numeri cerchiati uguali e allineati
 - Es. Una regione bianca isolata
- Può restituire **False** anche se la partita non è finita
 - Non ci sono violazioni evidenti, dovute ai simboli già inseriti
 - Ci sono però delle celle ancora non marcate



10.7 Hitori, suggerimenti (Py)

- Cercare automaticamente la prossima mossa
 - Senza backtracking, senza ricorsione
- *Provare* ad annerire una cella non annotata
 - Applicare gli automatismi dell'es. 9.7
 - Se le regole sono violate (**wrong**), cerchiare la cella
- *Provare* a cerchiare una cella non annotata
 - Applicare gli automatismi dell'es. 9.7
 - Se le regole sono violate (**wrong**), annerire la cella

Se invece una ipotesi non risulta wrong, non è detto che sia giusta; non si possono trarre conclusioni!



10.8 Hitori, altri suggerimenti (Py)

- Considerare ogni cella non annotata, singolarmente
 - Senza backtracking, senza ricorsione
- *Provare ad annerirla*
 - Applicare gli automatismi dell'es. 9.7
- *Provare a cerchiarla*
 - Applicare gli automatismi dell'es. 9.7
- *Confrontare* tutte le annotazioni, aggiunte nei due casi
 - Se una qualsiasi cella è annerita in entrambe le ipotesi, annerirla nel gioco
 - Se una qualsiasi cella è cerchiata in entrambe le ipotesi, cerchiarla nel gioco

