Esercizio assegnato per pausa didattica

Analizza il problema sotto descritto, evidenziando il **tipo dei dati** usati, definisci l'**algoritmo** risolutivo utilizzando come tecnica di rappresentazione lo *pseudocodice* ed **implementa** nel linguaggio di programmazione **Java** la soluzione proposta *potendo estrarre casualmente l'intero "cerca" e digitare da tastiera il numero "trovo" cercando di indovinare*:

Si vuole realizzare un programma che permetta una *partita* virtuale tra due giocatori/trici (tu e il PC) al gioco del "*cerca e trovo*" con le seguenti regole:

1) Un giocatore (il PC) pensa ad un numero intero tra 1 e 100 che si chiama appunto "cerca" e rimane nascosto per te.

2) L'altro/a giocatore/trice (tu) pronuncia un numero intero che chiama "trovo" tentando di indovinare.
Nel caso abbia indovinato, ci si complimenta con un "bravo" altrimenti si darà indicazione, affermando il vero, con le frasi "troppo grande" o "troppo piccolo".

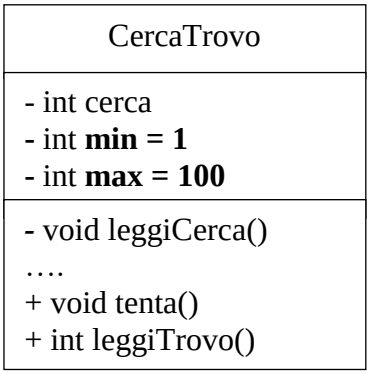
3) proporre *unico tentativo* e, al termine, stampare il numero inizialmente nascosto per te.

4) modificare il programma proponendo *più tentativi* ad esempio in numero noto oppure fino a quando non si indovina.

NB: valutazione massima se si propone soluzione *modulare* (illustrando con pseudocodice sia il flusso principale sia il/i modulo/i progettato/i)

Risolvere l'esercizio con OOP

Seguendo il **metodo di progettazione OO,** rappresentata con linguaggio UML la classe, si usi lo pseudocodice come tecnica di rappresentazione dei moduli significativi nel proporre la strategia risolutiva:



pseudocodice del flusso implementando con **modulo tenta**:

```
{
   leggiCerca
   fai
                  messaggio amichevole
                  trovo leggiTrovo
                  se (trovo> cerca)
                     messaggio troppo grande
                 altrimenti se (trovo< cerca)
                              messaggio troppo piccolo
   mentre (trovo != cerca)
   messaggio bravo  infatti sono uguali
}
```

Compiti assegnati per le vacanze di Natale

Realizzare *applicazioni* Java ***ben commentate***:

- Per leggere da tastiera: anni e altezza dell'utente
- Per trovare il massimo tra due numeri interi letti da tastiera
- Per sommare 2 numeri interi letti da tastiera
- Per stampare a monitor in formato tabellare una retta passante per l'origine, leggendo da tastiera il coefficiente angolare ed il termine noto

Ripassando le **strutture di controllo del flusso**: con riferimento agli esempi proposti nel manuale (fino a pg.8) all'indirizzo: http://new345.altervista.org/Dispense/java_corso_3AI_new.pdf

Risolvere **due** dei problemi seguenti:

- Calcolare la somma dei primi 10 valori estratti dalla tombola. Se viene estratto il numero precedentemente *digitato da tastiera* (ad esempio 50) il programma deve fermarsi.
- Contare quante volte durante i 30 lanci di 3 dadi escono le triplette 2-2-2 e 1-1-1. Se esce la tripletta realizzata col numero precedentemente *inserito da tastiera* (ad esempio la tripletta 6-6-6 con 6 digitato da tastiera) arrestare il programma.
- Simulare il lancio di due dadi per un numero di volte N *digitato da tastiera* (ad esempio 50 volte) e contare quanti sono i numeri pari e quanti i dispari usciti.
- Lanciare un dado a 20 facce per 100 volte e contare quante volte esce un valore precedentemente *inserito da tastiera* dall'utente.
- Un'urna contiene i 90 numeri della tombola. Simulare l'estrazione di un numero N di numeri precedentemente *digitato da tastiera* (ad esempio 30) e stampare la somma dei valori estratti.
- Simulare un numero N di lanci di un dado a 6 facce con N *inserito da tastiera* (ad esempio 100 lanci) e determinare quale faccia è uscita il maggior numero di volte.

Nb:

Analizzare il problema descritto, evidenziando il **tipo dei dati** usati,
definire l'**algoritmo** risolutivo utilizzando come tecnica di rappresentazione lo *pseudocodice*
implementare nel linguaggio di programmazione **Java** la soluzione proposta

Se si propone soluzione *modulare* : illustrare con pseudocodice sia il flusso principale sia il/i modulo/i progettato/i

Esercizi ARRAY - OOP

Esercizio 5.0: Scrivere un'applicazione Java per **leggere** da tastiera e **stampare un array**

Esercizio 5.1: Scrivere un'applicazione Java **StampaZigZag** che prevede un array di 10 numeri interi contenente valori a piacere (senza bisogno di chiederli all'utente) e ne stampa gli elementi secondo il seguente ordine: il primo, l'ultimo, il secondo, il penultimo, il terzo, il terz'ultimo, ecc...
Il nome dell'array può essere scelto a piacere. (Il programma deve essere scritto facendo finta di non sapere quali siano i valori inseriti nell'array)

Esercizio 5.2: Scrivere u un'applicazione Java **SommaPariDispari** che prevede un array di 10 numeri interi contenente valori a piacere (senza bisogno di chiederli all'utente) e stampa "*Pari e dispari uguali"* se la somma dei numeri in posizioni pari dell'array è uguale alla somma dei numeri in posizioni dispari, altrimenti il programma stampa "*Pari e dispari diversi"*. (Il programma deve essere scritto facendo finta di non sapere quali siano i valori inseriti nell'array)

Suggerimento: **pari** con **i** inizialmente 0 : **i ←i+2**
dispari … se **i** pari : **i+1**

Esercizio 5.3: Scrivere un'applicazione Java **SecondoArray** che chiede all'utente di inserire 10 numeri interi da tastiera (*anche negativi*) e li memorizza in un array. Successivamente, crea un nuovo array di dimensione pari al numero di valori maggiori o uguali a zero inseriti dall'utente. Copia tutti i valori maggiori o uguali a zero nel nuovo array e ne stampa i valori in ordine inverso.

Esercizio OOP: Progettare ed implementare la classe **Rettangolo** che risponda all'esigenza seguente:

costruire un rettangolo partendo da una base, un'altezza e dalle coordinate del piano.

Si desiderano due costruttori:

- uno ***di default***: crea un rettangolo con le dimensioni definite (base = 1, altezza = 1, x=0, y=0)
- uno ***parametrico***: permette all'utente di assegnare i valori

Si progettino i seguenti metodi:

- metodi di accesso
- metodi che eseguono operazioni specifiche:
  - traslazione delle coordinate
  - calcolo dell'area
  - calcolo del perimetro

Si proponga un'applicazione Java **RettangoloTester** che testa i metodi progettati.

Chiarimento: per test dei metodi progettati si intende anche l'uso dei due costruttori istanziando sia con costruttore di default che parametrico

Esercizio (*array di oggetti*):
In una gara il punteggio di ciascun atleta è dato dal pubblico. I voti possono andare da 1 a 10. Progettare ed implementare un'applicazione Java che per ogni atleta rilevi il numero di occorrenze dei vari voti.

array di oggetti: ad esempio chiamando **team** l'insieme di oggetti di tipo **Atleta**

Ricordando la *sintassi* per dichiarare e contemporaneamente creare un array:

tipo [] n*omeArray* = **new** tipo [dimensione];

NomeClasse [] *nomeArray* = new *NomeClasse* [*DIM*]; *// per DIM componenti …*
// precedentemente inizializzato

Atleta[] team = new **Atleta**[DIM];