PROGRAMMAZIONE (9 CFU) – A.A. 2016-2017	ES. 01

1) Digitare, salvare, compilare ed eseguire il programma *HelloWorld*.

```
public class HelloWorld{
        public static void main(String args[]){
                System.out.println("Hello World!");
        }
}
```

- Eliminare il modificatore static del metodo main() dalla classe HelloWorld. Compilare il programma e interpretare il messaggio di errore del programma.
- Eliminare la prima parentesi graffa aperta incontrata dalla classe HelloWorld. Compilare il programma e interpretare il messaggio di errore del programma.
- Eliminare l'ultima prima parentesi chiusa (ultimo simbolo del programma) dalla classe HelloWorld. Compilare il programma e interpretare il messaggio di errore del programma.
- Provare a far stampare una stringa a piacere al programma HelloWorld al posto di "*Hello World!*".
- Provare a far stampare un numero al programma HelloWorld al posto della stringa "*Hello World!*".
- Provare a far stampare la somma di due numeri al programma HelloWorld al posto della stringa.

2) Copiare, salvare e compilare la seguente classe:

```
public class NumeroIntero {
        public int numeroIntero;

        public void stampaNumero() {
                System.out.println(numeroIntero);
        }
}
```

Questa classe definisce il concetto di numero intero come oggetto. In essa vengono dichiarati una variabile intera ed un metodo che stamperà la variabile stessa.
Scrivere, compilare ed eseguire una classe (contenente ovviamente un metodo main()) che:

- istanzierà almeno due oggetti dalla classe NumeroIntero;
- cambierà il valore delle relative variabili e testerà la veridicità delle avvenute assegnazioni, sfruttando il metodo stampaNumero();
- aggiungerà un costruttore alla classe NumeroIntero che inizializzi la variabile d'istanza.

Due domande ancora:
- A che tipologia di variabili appartiene la variabile numeroIntero definita nella classe NumeroIntero?
- Se istanziamo un oggetto della classe NumeroIntero, senza assegnare un nuovo valore alla variabile numeroIntero, quanto varrà quest'ultima?

3) Creare una classe Quadrato che dichiari una variabile d'istanza intera lato. Creare un metodo pubblico che si chiami perimetro() che ritorni il perimetro del quadrato, e un metodo pubblico area() che ritorni l'area del quadrato.

- Creare una classe TestQuadrato contenente un metodo main() che istanzi un oggetto di tipo *Quadrato*, con lato di valore 5 (con una istruzione simile alla seguente: nomeOggetto.lato = 5;). Stampare poi il perimetro e l'area dell'oggetto appena creato.
- Si crei un costruttore nella classe Quadrato che prenda in input il valore della variabile *lato*. Fatto questo si compili la classe Quadrato.
- Ricompilare la classe TestQuadrato e interpretare l'errore.
- Modificare il codice della classe TestQuadrato in modo tale che compili e sia eseguita correttamente.
- Nella classe Quadrato sostituire il valore 4 usato per calcolare il perimetro con una costante d'istanza NUMERO_LATI.

4) Creare una classe Rettangolo equivalente alla classe Quadrato. Prima di codificare la classe decidere che specifiche deve avere questa classe (variabili e metodi).
   - Si crei una classe TestRettangolo contenente un metodo main() che testi la classe Rettangolo, equivalentemente a come fatto nell'esercizio precedente. Istanziare almeno due rettangoli diversi.

5) Creare una classe Moneta caratterizzata da due facce (Testa o Croce). Utilizzando la classe Random (è necessario importare la classe: import java.util.Random;), creare il metodo lancia() che restituisca un numero intero (0 per Testa e 1 per Croce)].
   Creare una classe TestMoneta dotata del metodo main() in cui s'istanzia un oggetto della classe Moneta. Simulare il lancio della moneta per 5 volte e stampare un messaggio con il numero delle volte in cui è uscito Croce.
   Nota: Istanziando un oggetto della classe Random, è possibile invocare il metodo nextInt(int max_value) per generare un numero intero. A titolo di esempio, nextInt(6) restituisce un numero random intero tra 0 e 5

6) Si scriva un insieme di classi per la gestione delle carte fedeltà degli utenti di un grande magazzino. La carta fedeltà è nominativa (appartiene ad un solo cliente) e consente l'accumulo dei punti (attraverso il metodo accumulaPunti()) calcolati sulla base della spesa effettuata. Il cliente può decidere, in ogni momento, di usufruire di una parte dei punti accumulati per l'ottenimento di un premio (implementare un opportuno metodo utilizzaPunti()). In tal caso, il totale dei punti residui dev'essere comunicato al possessore della carta (prevedere un metodo stampa()).

   - Si supponga che il grande magazzino regali 1 punto ogni 10€ di spesa e che abbia solo due clienti. Il primo cliente fa una spesa pari a 150€. Il secondo cliente fa una spesa di 300€. In seguito, il primo cliente fa una spesa pari a 1500€ e, dopo aver pagato, decide di utilizzare 100 dei punti accumulati per ritirare un premio. Il secondo cliente chiede di utilizzare 50 dei punti accumulati. Stampare a video opportuni messaggi per mostrare il saldo punti di ogni cliente.
   - Si definisca una costante VALORE_PUNTO che rappresenti il valore in € della spesa che permette di ottenere un punto. Si testi il software prodotto con diversi valori della costante VALORE_PUNTO.

7) La classe Macchina possiede 4 istanze della classe Ruota. Ogni Ruota è caratterizzata da un livello di pressione che varia tra 0 e 100, con 100 che indica il valore massimo di pressione. Inoltre, ogni Ruota possiede un coefficiente che indica di quanto essa si sgonfia ogni Km percorso. Tale coefficiente, compreso tra 0 e 1, è inizializzato a random al momento della creazione di un oggetto della classe Ruota (si utilizzi la funzione nextDouble() della classe Random che ritorna un numero fra 0.0 e 1.0).
   - Una Macchina possiede il metodo cammina() che accetta in input il numero di Km da percorrere. Durante il tragitto, le ruote tendono a sgonfiarsi (ciascuna in base al proprio coefficiente).
   - La classe Macchina ha un metodo monitora() che stampa il livello di pressione delle ruote.
   - La macchina è sottoposta periodicamente a revisione. Il metodo revisione() rigonfia tutte e solo le ruote che hanno un livello di pressione inferiore a PRESSIONE_MINIMA (si definisca tale valore costante e pari a 65).
   - Testare la classe Macchina supponendo di percorrere 40 Km, di revisionare la macchina, di percorrere 40 Km e di revisionare nuovamente la macchina. Stampare opportuni messaggi per monitorare la pressione.

NOTE PER COMPILAZIONE E TEST A RIGA DI COMANDO IN AMBIENTE LINUX:

Creare una cartella col proprio cognome sulla scrivania e i relativi file sorgenti al suo interno.
*Aprire una finestra di **terminale** e digitare:*
`cd Desktop/cognome` *oppure* **`cd Scrivania/cognome`** ***(****si posiziona nella directory)*
*Creare i file sorgente con **gedit** e salvarli nella propria directory.*
Digitare:
`javac nomeClasse.java` *(compila e genera il bytecode)*
`java nomeClasse` *(esegue il bytecode sulla JVM)*