Unità Formativa (UF): Programmazione Java
Docente: Nauro Bogliaccino
Titolo argomento: Verifica finale

in collaborazione con:
MINISTERO DELL'ISTRUZIONE, DELL'UNIVERSITÀ E DELLA RICERCA
UNIONE EUROPEA
REGIONE PIEMONTE
per una crescita intelligente, sostenibile ed inclusiva
www.regione.piemonte.it/europa2020
INIZIATIVA CO-FINANZIATA CON FSE

Esercizi metodi e control-flow

Nome progetto - cognome
package name its.tuocognome.esercizi

Esercizio01 - TrovaNumeriPrimi

Un numero primo è un numero maggiore di uno e divisibile solo per se stesso e uno.
Ad esempio: 2, 3, 5, 7 e 11 sono numeri primi.

Scrivere un metodo che restituisce true se l'argomento è un numero primo.
Scrivere un altro metodo che stampa tutti i fattori di un numero quando questo non è primo.

Esercizio02 - Controllo Flusso

Scrivere un'applicazione per stampare i numeri da 1 a N nel seguente modo:

1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 ...

Acquisire il numero intero N da scanner e stampare la serie

Esercizio03 - String e Character

Scrivere un metodo che ritorni il numero di lettere maiuscole in un oggetto String passato al metodo come parametro.

Il metodo di classe isUpperCase della classe Character restituisce true se il carattere passato come argomento è maiuscolo.

Esercizio04 - BodyMassIndex

Una quantità nota come BMI (Body Mass Index, indice di massa corporea) viene utilizzata per calcolare la probabilità di problemi di salute associati al peso.

Il Body Mass Index (BMI) o Indice di Massa Corporea (IMC) è un parametro molto utilizzato per ottenere una valutazione generale del proprio peso corporeo.

Esso mette in relazione con una semplice formula matematica l'altezza con il peso del soggetto.

Si ottiene dividendo il peso in Kg del soggetto con il quadrato dell'altezza espressa in metri.

Il risultato di tale formula classifica il soggetto in un'area di peso che può essere: normale - sottopeso - sovrappeso - obesità di medio grado - obesità di alto grado.

La formula per il calcolo del BMI è la seguente:

`BMI = w / (h * h)`

dove w è il peso in chilogrammi e h è l’altezza in metri.

Un BMI da 20 a 25 è considerato “nella norma”.

Il valore ottimale di BMI

Studi statistici hanno dimostrato che il valore di BMI associato a minore morbilita' e mortalita' per tutte le patologie per cui il sovrappeso è un fattore di rischio (malattie cardiovascolari, diabete, ipertensione, osteoartrosi, alcune neoplasie) è compreso tra 20 e 25.

IN PARTICOLARE IL VALORE DI BMI OTTIMALE E' RISPETTIVAMENTE:

- da 19 a 35 anni: BMI= 22 (19-25)
- più di 35 anni: BMI= 24 (21-27)

Scrivere un programma che dati peso e altezza calcola il BMI.

Il programma deve presentare un menu utente in formato testuale:

0.inserisci nuovo utente
1.calcola peso (inserire peso e altezza)
2.stampa elenco utenti (gestire un array o una collezione)
3.esci

In base ai parametri, stampare una delle seguenti etichette:

BMI	CONDIZIONE
< 16.5	GRAVE MAGREZZA
16-18,49	SOTTOPESO
18.5-24,99	NORMOPESO
25-29,99	SOVRAPPESO
30-34,99	OBESITÀ CLASSE I (lieve)
35-39,99	OBESITÀ CLASSE II (media)
> 40	OBESITÀ CLASSE III (grave)

Esercizio OOP - Uso delle classi

Nome progetto - cognome
package name its.tuocognome.classi

Esercizio01 - Scrivi la classe corretta

Si consideri la classe Test:

```
public class Test {

    public static void main(String[] args) {

        A a1 = new A(5);

        A a2 = new A();



        System.out.println(a1.getN());

        System.out.println(a2.getN());

        a1.raddoppiaN();

        a2.raddoppiaN();

        System.out.println(a1.getN());

        System.out.println(a2.getN());

    }

}
```

Si progetti una classe A affinché la classe Test compili ed il risultato dell'esecuzione sia:

5
0
10
0

Esercizio02 - libretti di risparmio

Scrivere un applicazione per la gestione di libretti di risparmio.
Il programma può gestire diversi libretti di risparmio.
Per ogni libretto, l'utente titolare può depositare una somma, prelevare una somma, e chiedere i movimenti ed il saldo.

Sia i prelievi, sia i depositi sono numerati in ordine crescente.

In una classe LibrettoDemo, istanziare qualche Libretto di prova, testare i metodi e stampare l'elenco dei movimenti e il saldo di ciascun libretto

Esercizio03 - conversioni di valuta

Realizzare una classe per effettuare conversioni di valuta.
La classe deve essere in grado di gestire la conversione tra valute multiple.

Fornire un metodo exchange che funziona come segue:

```
euro = converter.exchange( "dollar", "euro", 250.0 );
```

per convertire 250 dollari in una somma corrispondente in euro.

Fornire un metodo per impostare i tassi di cambio.

Ad esempio:

```
converter.setRate( "dollar", "euro" , 0.98 );
```

significa che 1 dollaro equivale a 0.98 euro.

Esercizio04 - Grossista di caffé

Un grossista di caffè utilizza sacchi da 1 kg per spedire il prodotto ai clienti.

Ogni sacco costa 6 euro. Quando un cliente fa un ordine, la merce viene spedita imballando i sacchi in tre tipi di scatole:

- grande (20 sacchi),
- media (10 sacchi)
- piccola (5 sacchi)

del costo, rispettivamente, di 2, 1 e 0.5 euro.

L'ordinativo viene inviato al
cliente minimizzando il numero di cartoni.

Per esempio un ordine di 25 sacchi verrà spedito usando un cartone grande e un cartone piccolo.

Scrivere un programma che calcoli il costo totale dell'ordine.

Si utilizzi uno Scanner per richiedere il numero dei sacchi ordinati e in output occorre visualizzare:

- il costo totale del solo caffè
- il numero di cartoni utilizzato per ogni formato
- il costo totale dell'ordine

Esercizio 01 Servlet e Jsp - Ristorante MVC

Nome progetto - cognome
package name its.tuocognome.servlet

In un ristorante l’ordinazione ai tavoli è automatizzata mediante un programma che
consente all’utente di scegliere primo, secondo, dessert e bevanda. Le scelte con i relativi
prezzi, sono le seguenti:

PRIMI	SECONDI	DESSERT	BEVANDE
Ravioli al ragù 5 euro	Torta pasqualina 6 euro	Tiramisù 3 euro	Acqua 1 euro
Pansotti in salsa di noci 5.5 euro	Stoccafisso alla Genovese 5 euro	Torta di mele 2.5 euro	Vino rosso 2 euro
Trenette al pesto 4 euro	Polpo in aglio e prezzemolo 4.5 euro	Torta di pinoli 3 euro	Vino bianco 2 euro
Minestrone 3.5 euro	Cima 4.5 euro	Torta di rose 2.5 euro	Birra 1.5 euro

Si utilizzi una vista jsp, con un modulo (form) html per gestire l'ordine dei piatti per un singolo tavolo,
mostrare le voci dei menu divise per categoria.

Un tavolo (di cui gestite l'ordine dei piatti) ha come proprietà il numero del tavolo e il numero di coperti
utilizzati effettivamente dai clienti (p.es. Tavolo n. 13, coperti:3)

Si utilizzi una vista jsp per mostrare il riepilogo e il prezzo totale dell’ordine (considerando un costo aggiuntivo di 1.5 euro per pane e coperto).

Esercizio 01 - Spring Boot - Magazzino REST

Nome progetto - cognome
package name its.tuocognome.spring.magazzino

Esercizio01 - Magazzino

Modellare la classe Prodotto con le seguenti proprietà:

- codProdotto
- nomeProdotto
- categoria
- quantita

e la classe Movimento con le seguenti proprietà:

- id
- codProdotto
- data
- carico
- scarico

Un movimento di magazzino causa la modifica della quantità disponibile di prodotti.

Realizzare un applicativo con SpringBoot per gestire il magazzino, implementando
correttamente l'architettura multi-tiers: model, dao, service e integration layer.
Utlizzare correttamente classi, interfacce e annotazioni Spring.

Fornire le API per le operazioni di CRUD sulla tabella creata da Spring Data JPA.
Occorre inserire una pagina index.html nellinterfaccia web dell'app,
indicando gli endpoint disponibili.

In particolare, testando gli endpoint con un rest client, occorre fornire in formato json:

- tutti i prodotti
- un prodotto dato il codice
- i prodotti data una categoria
- aggiungi prodotto
- modifica prodotto
- elimina prodotto
- nuovo movimento attivo (carica il magazzino)
- nuovo movimento passivo (scarica il magazzino)

NB: Fornire il dump del database usato per il test dell'applicativo.