**Esercizi di programmazione per il capitolo 4**

LA SUA PAGINA CONTIENE diversi esercizi per il Capitolo 4 in [Introduzione alla programmazione con Java](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/index.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhidfZzTu7sHSMmrsVmwrJ-4EUsVUw) . Per ogni esercizio viene fornito un collegamento a una possibile soluzione. Ogni soluzione include una discussione su come un programmatore può affrontare il problema e punti interessanti sollevati dal problema o dalla sua soluzione, oltre al codice sorgente completo della soluzione.

**Esercizio 4.1:**

Per "capitalizzare" una stringa significa cambiare la prima lettera di ogni parola nella stringa in maiuscolo (se non è già in maiuscolo). Ad esempio, una versione maiuscola di "Ora è il momento di agire!" è "Ora è il momento di agire!". Scrivi una subroutine denominata printCapitalized che stamperà una versione maiuscola di una stringa sullo standard output. La stringa da stampare dovrebbe essere un parametro per la subroutine. Testare la subroutine con una routine main () che ottiene una riga di input dall'utente e applica la subroutine ad essa.

Si noti che una lettera è la prima lettera di una parola se non è immediatamente preceduta nella stringa da un'altra lettera. Ricordiamo dall'esercizio [3.4](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c3/ex4-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhi3bm3E9CuWZsfcMbbBag_GEOr-5Q) che esiste una funzione standard valutata booleana Character.isLetter (char) che può essere usata per verificare se il suo parametro è una lettera. Esiste un'altra funzione con valore char standard, Character.toUpperCase (char) , che restituisce una versione in maiuscolo del singolo carattere passata ad esso come parametro. Cioè, se il parametro è una lettera, restituisce la versione maiuscola. Se il parametro non è una lettera, restituisce solo una copia del parametro.

[Vedi la soluzione](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex1-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhjN8xrQmusBo3chZUMJmXxu_bDUUg)

**Esercizio 4.2:**

Le cifre esadecimali sono le normali cifre da 10 in base 10 a 9 e le lettere da A a F. Nel sistema esadecimale, queste cifre rappresentano i valori da 0 a 15, rispettivamente. Scrivi una funzione denominata hexValue che utilizza un'istruzione switch per trovare il valore esadecimale di un determinato carattere. Il carattere è un parametro della funzione e il suo valore esadecimale è il valore di ritorno della funzione. Le lettere minuscole da 'a' a 'f' devono avere lo stesso valore delle lettere maiuscole corrispondenti. Se il parametro non è una delle cifre esadecimali legali, restituire -1come valore della funzione.

Un numero intero esadecimale è una sequenza di cifre esadecimali, come 34A7, ff8, 174204 o FADE. Se str è una stringa che contiene un intero esadecimale, allora il numero intero base-10 corrispondente può essere calcolato come segue:

valore = 0;

for (i = 0; i <str.length (); i ++)

valore = valore \* 16 + hexValue (str.charAt (i));

Naturalmente, questo non è valido se str contiene caratteri che non sono cifre esadecimali. Scrivi un programma che legge una stringa dall'utente. Se tutti i caratteri nella stringa sono cifre esadecimali, stampare il valore di base 10 corrispondente. In caso contrario, stampare un messaggio di errore.

[Vedi la soluzione](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex2-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhhz1XCtvCtfy1Iof8Qnad4oYTBrfA)

**Esercizio 4.3:**

Scrivi una funzione che simula il lancio di una coppia di dadi finché il totale dei dadi non diventa un numero dato. Il numero per cui si sta svolgendo è un parametro della funzione. Il numero di volte che devi tirare i dadi è il valore di ritorno della funzione. Il parametro dovrebbe essere uno dei possibili totali: 2, 3, ..., 12. La funzione dovrebbe lanciare un *IllegalArgumentException* se questo non è il caso. Usa la tua funzione in un programma che calcola e stampa il numero di rotoli necessari per ottenere gli occhi di serpente. (Occhi di serpente significa che la visualizzazione totale dei dadi è 2.)

[Vedi la soluzione](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex3-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhjuxBpZPl_bjc7PTUiHKdVstjZong)

**Esercizio 4.4:**

Questo esercizio si basa [sull'esercizio 4.3](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex3-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhjuxBpZPl_bjc7PTUiHKdVstjZong) . Ogni volta che tiri ripetutamente i dadi, cercando di ottenere un dato totale, il numero di lanci necessari può essere diverso. La domanda sorge spontanea, qual è il numero medio di tiri per ottenere un dato totale? Scrivi una funzione che esegue l'esperimento di rotazione per ottenere un dato totale di 10000 volte. Il totale desiderato è un parametro per la subroutine. Il numero medio di rotoli è il valore di ritorno. Ogni singolo esperimento dovrebbe essere fatto chiamando la funzione che hai scritto per l' [Esercizio 4.3](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex3-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhjuxBpZPl_bjc7PTUiHKdVstjZong) . Ora, scrivi un programma principale che chiamerà la tua funzione una volta per ciascuno dei possibili totali (2, 3, ..., 12). Dovrebbe fare una tabella dei risultati, qualcosa come:

Numero totale di rotoli sui dadi

------------- -----------------------

2 35,8382

3 18.0607

. .

. .

[Vedi la soluzione](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex4-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhgx1CVXQsSqXTAruSAo_ii4Vkq3tw)

**Esercizio 4.5:**

Il programma di esempio [*RandomMosaicWalk.java*](http://math.hws.edu/eck/cs124/javanotes7/source/chapter4/RandomMosaicWalk.java) della [Sezione 4.6](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/s6.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhgwjoiwIhkQCgoqAQs5xeFtq7DD9Q) mostra un "disturbo" che si aggira intorno a una griglia di quadrati colorati. Quando il disturbo visita un quadrato, il colore di quel quadrato è cambiato. Ecco un'idea per una variazione su quel programma. Nella nuova versione, tutti i quadrati iniziano con il colore predefinito, nero. Ogni volta che il disturbo visita un quadrato, viene aggiunta una piccola quantità alla componente verde del colore di quel quadrato. Il risultato sarà un effetto visivamente interessante, poiché il percorso seguito dal disturbo gradualmente diventa un verde più luminoso e luminoso.

Scrivi una subroutine che aggiungerà 25 al componente verde di uno dei quadrati nel mosaico. (Ma non lasciare che il componente verde superi 255, poiché questo è il valore legale più grande per un componente di colore.) I numeri di riga e di colonna del quadrato dovrebbero essere dati come parametri per la subroutine. Ricorda che puoi scoprire il componente verde corrente del quadrato nella riga r e nella colonna c con la funzione chiamata Mosaic.getGreen (r, c) . Utilizzare la subroutine come sostituto della subroutine changeToRandomColor () nel programma [*RandomMosaicWalk2.java*](http://math.hws.edu/eck/cs124/javanotes7/source/chapter4/RandomMosaicWalk2.java) . (Questa è la versione migliorata del programma della [Sezione 4.7](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/s7.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhiM0R6V8lb68xQmpzbTxfw2smDK1w) che utilizza costanti nominate per il numero di righe, il numero di colonne e le dimensioni quadrate.) Imposta il numero di righe e il numero di colonne su 80. Imposta le dimensioni quadrate su 5 .

Per impostazione predefinita, i rettangoli nel mosaico hanno un aspetto "3D" e un bordo grigio che li rende più belli nel programma random walk. Ma per questo programma, vuoi disattivare quell'effetto. Per fare ciò, chiama Mosaic.setUse3DEffect (false) nel programma principale.

Non dimenticare che avrai bisogno di [*Mosaic.java*](http://math.hws.edu/eck/cs124/javanotes7/source/chapter4/Mosaic.java) e [*MosaicPanel.java*](http://math.hws.edu/eck/cs124/javanotes7/source/chapter4/MosaicPanel.java) per compilare ed eseguire il tuo programma, poiché definiscono le classi non standard richieste dal programma.

[Vedi la soluzione](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex5-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhgVnxQsLXM_sjF8wm4KaHsDc0VxUQ)

**Esercizio 4.6:**

Per questo esercizio, farai qualcosa di ancora più interessante con la classe *Mosaic* discussa nella [Sezione 4.6](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/s6.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhgwjoiwIhkQCgoqAQs5xeFtq7DD9Q) . (Ancora una volta, non dimenticare che avrai bisogno di [*Mosaic.java*](http://math.hws.edu/eck/cs124/javanotes7/source/chapter4/Mosaic.java) e [*MosaicPanel.java*](http://math.hws.edu/eck/cs124/javanotes7/source/chapter4/MosaicPanel.java) per compilare ed eseguire il tuo programma.)

Il programma che scrivi per questo esercizio dovrebbe iniziare compilando un mosaico con colori casuali. Quindi ripeti quanto segue fino a quando l'utente chiude la finestra del mosaico: Seleziona uno dei rettangoli nel mosaico a caso. Quindi seleziona uno dei rettangoli vicini, sopra di esso, sotto di esso, a sinistra o a destra di esso. Copia il colore del rettangolo selezionato in origine nel vicino selezionato, in modo che i due rettangoli ora abbiano lo stesso colore.

Poiché questo processo viene ripetuto all'infinito, diventa sempre più probabile che i quadrati vicini abbiano lo stesso colore. Il risultato è di creare patch di colore più grandi. D'altra parte, una volta scomparso l'ultimo quadrato di un determinato colore, non c'è modo che quel colore riappaia mai. (L'estinzione è per sempre!) Se lasci che il programma funzioni abbastanza a lungo, alla fine l'intero mosaico avrà un colore uniforme.

[Vedi la soluzione](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex6-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhixnDl0Ii5VSjDSrvb7N6AUDRbedg)

**Esercizio 4.7:**

Scrivi un programma che amministri un quiz di addizione di base per l'utente. Dovrebbero esserci dieci domande. Ogni domanda è un semplice problema di addizione come 17 + 42 , in cui i numeri nel problema sono scelti a caso (e non sono troppo grandi). Il programma dovrebbe chiedere all'utente tutte e dieci le domande e ottenere le risposte dell'utente. Dopo aver fatto tutte le domande, l'utente dovrebbe stampare di nuovo ogni domanda, con la risposta dell'utente. Se l'utente ha la risposta giusta, il programma dovrebbe dirlo; in caso contrario, il programma dovrebbe fornire la risposta corretta. Alla fine, comunica all'utente il loro punteggio sul quiz, dove ogni risposta corretta conta per dieci punti.

Il programma dovrebbe utilizzare tre subroutine, una per creare il quiz, una per amministrare il quiz e una per classificare il quiz. Può utilizzare tre array, con tre variabili globali di tipo int [] , per fare riferimento agli array. Il primo array contiene il primo numero da ogni domanda, il secondo contiene il secondo numero da ogni domanda e il terzo contiene le risposte dell'utente.

[Vedi la soluzione](https://translate.googleusercontent.com/translate_c?depth=1&hl=en&rurl=translate.google.it&sl=auto&sp=nmt4&tl=it&u=http://math.hws.edu/eck/cs124/javanotes7/c4/ex7-ans.html&xid=17259,15700002,15700022,15700124,15700149,15700168,15700173,15700186,15700201&usg=ALkJrhjwx_Vg4OI1wRTyCZ3f8SnucaVzOQ)