NEUTRINO

Desarrollador Java profesional
Inicio de carrera: Java
Cimientos

Ejercicios y suplementos

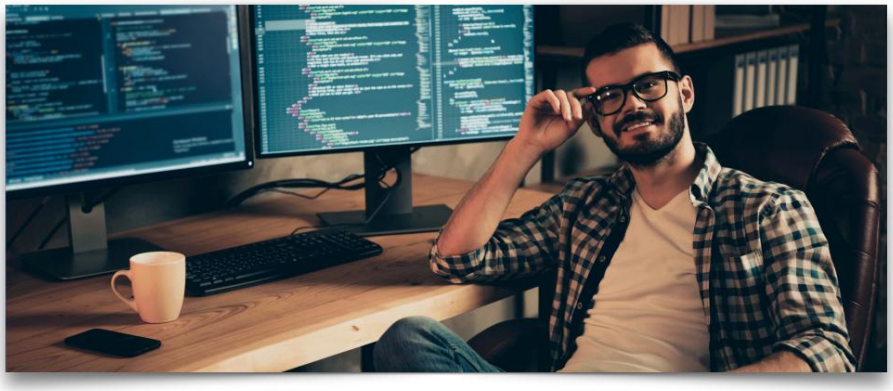
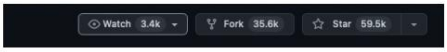


Tabla de contenido

Prefacio

Los ejercicios de este documento varían en dificultad, pero algunos de ustedes pueden encontrarlos bastante difíciles. Una razón para esto es que he intentado crear ejercicios que puedan hacer que tengan que pensar de manera más indirecta. Por ejemplo, podría decir: "Escriba un programa que almacene diez palabras en un
matriz, genera un entero entre cero y nueve, y luego usa ese entero para seleccionar una palabra aleatoria de la matriz". O, podría decir: "Escribe un programa que elija una palabra aleatoriamente
de diez opciones". La segunda oración es mejor en mi opinión porque es la forma en que los no programadores dirían algo y te obliga a ti, el programador, a traducir esa solicitud en mecanismos de programación que conoces. La primera oración básicamente te dice cómo resolver
El problema ya está.

La parte de la programación que creo que me resultará más difícil enseñarte es cómo traducir
solicitudes del mundo real en código que se compone de bloques fundamentales de programación. Sin embargo, creo que esa parte es la más divertida porque es donde puedes usar tu creatividad. Algunos de ustedes pensarán que cada problema que se les pida resolver ya tiene un
solución conocida y tu trabajo como programador es simplemente memorizar todas las soluciones y usar una cuando sea necesario. Por supuesto, hasta cierto punto, esto es parcialmente cierto porque hay muy
Hoy en día, nadie le planteará a un programador nuevas solicitudes. Sin embargo, la probabilidad de que conozca todas esas soluciones es baja. Y el tiempo que le puede llevar encontrarlas con Google puede ser mayor que el que le llevaría usar su creatividad para crear su propio enfoque. Tendrá que evaluar cuándo es apropiado hacer una cosa u otra. Puedo decirle que todos los desafíos (ejercicios) de este documento están pensados para que estén dentro de sus propias capacidades y no debería necesitar buscarlos en Google.
cómo hacerlos (bueno, puede que necesites/quieras buscar en Google sintaxis o métodos específicos cuyos nombres no recuerdas, pero no un enfoque/estrategia/algoritmo general) - pero muchos serán difíciles de hacer para algunos de ustedes. He intentado incluir sugerencias o notas (que encontrarás en el margen izquierdo de este documento - así que no busques si no quieres hacer trampa ;-)) para aquellos problemas en los que necesites más ayuda, pero intenta resolverlos por tu cuenta, sin las notas primero. Si, después de leer el
Notas: todavía no puedes encontrar una manera, también he proporcionado soluciones completamente desarrolladas para casi todos los problemas (solo dejé afuera algunas que sentí que eran obvias). Esas soluciones se pueden encontrar aquí en GitHub (se puede hacer clic). Por cierto, es posible que actualice con frecuencia el proyecto de soluciones en GitHub. Entonces, Si desea recibir notificaciones automáticas por correo electrónico cada vez que lo actualice, puede hacer clic en el
Botón "Mirar" en la parte superior derecha de
la página del repositorio del proyecto en GitHub.



Es posible que notes que, de una sección a otra, te
propongo ejercicios que te piden que uses las habilidades que aprendiste en las secciones anteriores.
intencional para que puedas tener más oportunidades de practicar poniendo en práctica todas estas técnicas

Juntos podemos resolver problemas más grandes. También puedo repetir ejercicios en secciones posteriores, pero te pido que los resuelvas nuevamente usando técnicas más nuevas que ya hayas aprendido.

Por último, permítanme decir que estos ejercicios están aquí para que los hagan como quieran. No piensen en ellos como tareas. No me importa si las haces o no. Algunos de ustedes no harán ninguna, eso está perfectamente bien para mí. Pero, mejorarán mucho si practican, y para eso están estas. Entonces, para aquellos que decidan probarlos… ¡buena suerte y DIVIÉRTANSE!

Clases, objetos y métodos

Suplemento sobre matrices

Las matrices también pueden ser multidimensionales. En la lección en video, solo muestro cómo crear matrices de una dimensión, que pueden considerarse como una columna de valores en una hoja de cálculo o una fila.

Pero las matrices pueden modelar una tabla de dos o más columnas y/o filas, o tener incluso más dimensiones que esas.

One Dimension

one	two	three

Two Dimension

one	two	three
a	b	c

Para crear una matriz bidimensional, puedes hacer lo siguiente:

```
String[][] my2d = new String[2][3];
my2d[0][0] = "one";
my2d[0][1] = "two";
my2d[0][2] = "three";
my2d[1][0] = "a";
my2d[1][1] = "b";
my2d[1][2] = "c";
System.out.println(my2d[0][2]);
```

O, alternativamente, puedes declarar e inicializar la matriz al mismo tiempo de la siguiente manera:

```
String[][] my2d = {{"one", "two", "three"}, {"a", "b", "c"}};
System.out.println(my2d[1][0]);
```

Ceremonias

1. Crea un nuevo proyecto Java e intenta modelar un dominio con el que te sientas cómodo. Podría ser algo como: Deportes, Negocios, Matemáticas, Compras en línea, etc. Por ejemplo, si Si está modelando una compra en línea, puede crear clases para representar: Cliente, Producto, Pedido, etc. Estas clases probablemente tengan propiedades y puede pensar en algunos métodos que también puedan tener sentido para ellas. No hay código de ejemplo para esto.
2. Cree una matriz e inicialícela con los días de la semana.
    2.1. Utilice System.out.println() para imprimir:
        2.1.1.El número de elementos en la matriz

2.1.2.El 4º día en la matriz

3. Cree una matriz e inicialícela con números del 1 al 10.

4. Intenta crear una matriz para representar el tablero de tres en raya a la derecha.

4.1.¿Cómo accederías al valor en el cuadrado inferior derecho con ¿Código Java?

5. Escriba un método que le permita pasar cualquier número de cadenas como entradas de parámetros sin utilizando una matriz.

6. ¿Cómo se puede crear un método que pueda llamarse sin crear una instancia de su clase?

7. Modela un automóvil, crea una instancia del mismo, pásala a System.out.println() para que sus propiedades ser impreso

8. Modele un banco con la capacidad de acceder a la bóveda. Modele también un gerente de banco y un cliente. Asegúrese de que el cliente no pueda acceder directamente a la bóveda del banco, pero el gerente de banco poder.

9. ¿Cómo se puede garantizar que no se pueda crear una instancia de Cliente sin un nombre y una ¿depósito inicial? con un constructor

10. ¿Cómo podrías modelar una clase de matemáticas para conocer el número de Euler?

11. Modela una clase con datos que enumera (asocia con un número) los primeros nombres de tres Amigos, compañeros de clase o de trabajo. Haga que estos datos de nombre se compartan entre todos. instancias de la clase.

Trabajar con texto

Ceremonias

1. Modele una Persona con un nombre y apellido y asegúrese de que incluso si el nombre es ingresado todo en minúsculas, se almacenará todo en mayúsculas.
2. Escribe código que reemplace la palabra "gato" por la palabra "perro" donde sea que aparezca en un oración.
3. ¿Cómo puede asegurarse de que cuando las personas ingresan texto en un programa, no haya errores no deseados? ¿Espacios al principio o al final del texto? Por medio de los métodos de los string
4. Si alguien ingresa “alfabeto” (observe los espacios) como entrada en un método, haga que en su lugar, el método emite “alphabeT”.
5. Dada la cadena “12345 Big St., Alphabet City, CA 90210” o cualquier otra dirección con la mismo formato, ¿puedes escribir código que pueda analizar e imprimir:
    1. Número del edificio: 12345
    2. La calle: “Big St.”
    3. Ciudad: “Ciudad Alfabética”
    4. Estado: “CA”
    5. Código Postal: 90210

Expresiones regulares

Suplemento sobre expresiones regulares

Referencias anteriores

Las expresiones regulares de Java también admiten referencias inversas. Esta es una capacidad de una expresión regular para hacer referencia volver a un grupo de captura anterior en la expresión. Por ejemplo, si se proporciona una cadena:

“aaabbb12345cccdddzzz12345”

Podríamos escribir una expresión regular que coincida con la misma secuencia de números de la siguiente manera:

“[az]+(\\d{5})[az]+\\1”

Dónde:

“[az]+” coincide con “aaabbb”

“(\\d{5})” coincide con “12345” Y se almacena en la referencia n.° 1

“[az]+” también coincide con “cccdddzzz”

“\\1” hace referencia al grupo de captura n.° 1, que es “12345”

También podemos utilizar grupos de captura con nombre y referencias posteriores de la siguiente manera:

“[az]+(?<núms>\\d{5})[az]+\\k<núms>”

Tenga en cuenta el uso de “\\k<nums>” para hacer referencia al grupo de captura con nombre de “nums”. Asegúrese de incluir la “k” en “\\k” para indicar que está haciendo referencia a un grupo con nombre.

Ceremonias

1. Escribe una expresión regular que coincida con las siguientes palabras: Dark, bark, Lark
    1. Para un desafío adicional, ¿podrías hacer que coincida con: Stark?
2. Utilice grupos de captura para escribir una expresión regular que pueda coincidir con: Abracadabra o Agracadagra
3. ¿Cómo se pueden utilizar paréntesis en una expresión regular para agrupar pero sin capturar? \\c
4. Escriba una expresión regular que pruebe si una cadena es una dirección y le permita extraer las partes
    (su elección para el formato de dirección).
5. Escriba una expresión regular que pruebe si una cadena es una dirección de correo electrónico.
    1. Nota: Hacer esto en expresiones regulares es en realidad notoriamente difícil de cumplir con TODAS las formas en que una Se puede escribir la dirección de correo electrónico. Sin embargo, puedes utilizar la forma más simple que se te ocurra, como por ejemplo: nombre.apellido@dominio.com

1. Terry Martín
7 de enero de 2022 a las 17:20:05
Puedes asignar a cada tipo de señal un bit de un byte. Para cada tipo de señal recibida, activa el bit correspondiente. Luego, convierte el patrón de bits binarios en un número.

2. Terry Martín
7 de enero de 2022 a las 17:17:36
Tome el patrón de bits binarios que representa los tipos de señales recibidas y combinelo con el patrón de bits binarios del tipo de señal que está probando.

Si recibiste 1 0 1 y quieres saber si el bit más a la izquierda es un 1, debes hacer AND.

1 0 1 Y
1 0 0
El resultado sería:
1 0 0
Y sabrías que el bit más a la izquierda está "activado".

3. Terry Martín
7 de enero de 2022 a las 17:25:03
Puedes utilizar una operación lógica OR. En Java, este es el símbolo de tubería, "|".

2 + 4 es equivalente a:
2 | 4

Nota: esto sólo funciona para números cuyos patrones de bits binarios no se superponen.
2 | 2 no es igual a 2 + 2
0 0 0 0 0 0 1 0 = 2
O
0 0 0 0 0 1 0 0 = 4
———————
0 0 0 0 0 1 1 0 = 6

PERO
0 0 0 0 0 0 1 0 = 2
O
0 0 0 0 0 0 1 0 = 2
———————
0 0 0 0 0 0 1 0 = 2

4. Terry Martín
8 de enero de 2022 a las 15:14:11
Necesitarás una forma de almacenar los 10 nombres en una clase.

Necesitará una forma de acceder a cada nombre cada vez que se llame al método next().

Considere utilizar una matriz, un número e incrementar el número.

5. Terry Martín
8 de enero de 2022 a las 15:46:35
Este problema se basa principalmente en matemáticas simples y en tu capacidad para reconocer un patrón simple. Cuando tienes un rango de datos en un conjunto que necesita clasificarse o agruparse en intervalos regulares, a menudo buscas dividir por ese intervalo regular. En este caso, el

Números

Ceremonias

1. Si pudieras recibir hasta ocho tipos diferentes de señales de radio simultáneamente (en tu computadora y en un programa que escribiste) y necesitaras poder registrar cuáles de esas ocho recibiste en un momento dado, ¿cuál sería una forma altamente compacta (o comprimida, usando la menor cantidad de memoria o almacenamiento) de registrarlas (usando lo que aprendiste en este curso)?
módulo)? Por ejemplo, podría recibir las señales A + B + C simultáneamente, o A + D, o solo G, etc.

    1. Si recibieras las señales A + D + C simultáneamente, ¿cómo podrías determinar eficientemente que D era una de las señales que recibiste, basándose en la solución anterior?

2. ¿Cuál es una forma alternativa de sumar 2 + 4 sin usar el símbolo "+"?

3. Usando solo lo que has aprendido hasta ahora en este curso, crea una clase que contenga 10 nombres en minúscula y tenga un método que pueda llamarse 10 veces por separado, cada vez, devuelve el siguiente nombre. Si se llama al método "next()", la primera vez que se llama, devuelve "nombre1", la segunda vez que se llama "next()", devuelve "nombre2", etc.

    1. Haga que el método next() escriba en mayúscula la primera letra de cada nombre a medida que los muestra.

4. Imagina que te han dado datos que representan el tiempo que los artículos han permanecido en un almacén. Tu trabajo es clasificarlos según el tiempo que han permanecido en el almacén. Los artículos se pueden clasificar como 0, 1, 2 o 3. Los artículos de la clase 0 tienen menos de 89 días de antigüedad. Clase 1 = 90-179 días, Clase 2 = 180-269 días, Clase 3 = 270-364 días. Para los tiempos de espera en el almacén dados: 13, 49, 90, 130 , 175, 181 , 255, 310, 330, 359, escriba una clase similar al Ejercicio 3 anterior, que tenga un método next() que se puede llamar 10 veces y genera un número que representa la clasificación de cada número en el conjunto de edades anterior. Ejemplo: un elemento que espera 5 días devolvería 0 y Un artículo que espera 92 días devolverá 1.

5. Cree un método, next(), que pueda llamarse 10 veces y genere un entero aleatorio entre 0 y 10 (no incluidos). Este método mantiene una suma continua de todos los números aleatorios que genera.
y devolver esa suma cada vez. Si la primera vez que se llama a next(), genera 5, y la segunda vez que se llama genera 3, debería devolver 8 de la segunda llamada, por ejemplo.

6. Escribe una función que tome una cadena como "149,32" y la formatee como dinero del lugar donde vives. Por lo tanto, si vives en Estados Unidos, devolvería "$149,32", Corea = 149, Francia/ UE = 149,32 €, etc.

7. Escriba una función que tome una entrada de cadena de "$12,345.83" e imprima ese valor dividido
Para el 32,19 debería devolver $383,53.

8. Utilice printf() para formatear las siguientes entradas para imprimir las siguientes salidas

El intervalo es 90. Al dividir cada valor por 90, traducirá los valores directamente a su clasificación numérica de 0, 1, 2 o 3.

Aporte	Producción
123456.783	$123,456.78
-9876.32532	(9,876.325)
23.19283928394829182	2.319284e+01f
123456	0000123456
-9876.35532	-9.876,4

9. ¿Cómo podría formatear cada una de las entradas en la tabla anterior y almacenarlas en variables de cadena en lugar de simplemente imprimirlas directamente?
10. Utilice instancias de DecimalFormat para realizar las mismas conversiones en la tabla anterior.
11. Escriba un método que tome las entradas de cadena, “37” y “13”, y devuelva un entero de su suma, 50.

6. Terry Martín
11 de enero de 2022 a las 16:40:40
Intente utilizar el operador ternario

7. Terry Martín
12 de enero de 2022 a las 14:16:25
Utilice la expresión de cambio
para determinar qué comida para un día
determinado y asignar esa comida a una variable.
Luego, fuera de la expresión
de cambio, use printf() para imprimir una
cadena formateada usando variables para la
comida y el día de la semana.

Flujo de control

Ceremonias

1. Escriba un método en 4 líneas de código (sin incluir el nombre del método, la firma y las llaves) que muestre los días de la
semana, uno por línea, utilizando un "bucle for" normal y una matriz.
1. Hazlo de nuevo utilizando el bucle for mejorado.
2. Hazlo de nuevo pero haz que todas las demás líneas de salida estén completamente en mayúsculas.

Domingo
LUNES
Martes
MIÉRCOLES
etc…

6 3. Hazlo de nuevo pero usa solo 4 líneas de código (sin incluir el método y las llaves de
método.

2. Repita todas las partes del ejercicio n.° 1, pero utilice “while-loop” en lugar de “for-loop” (cuando corresponda).
3. Repita el paso 2, pero utilice un bucle "do-while". (No se han proporcionado soluciones para este caso)
4. Utilice una matriz de días de la semana, un bucle for mejorado y if/else para crear lo siguiente
producción.

Comemos carne asada los domingos
Comemos espaguetis el lunes
Comemos tacos los martes
Comemos pollo el miércoles
Comemos pastel de carne el jueves.
Comemos hamburguesas el viernes
Comemos pizza el sábado

5. Repita el ejercicio n.° 4, pero en lugar de if/else, utilice el tradicional switch/case.
7 6. Repita el ejercicio n.° 5 pero utilice una expresión de cambio más nueva con la menor repetición posible.
1. Agregue un método privado que pueda poner en mayúscula la primera letra de cada palabra de la comida.
En lugar de “Comemos espaguetis los domingos”, se imprimiría: “Comemos espaguetis los domingos”.
No te limites a escribir con mayúscula los nombres de las comidas. Deja que tu nuevo método lo haga por ti.
2. Mejorar 6.1 para que sea lo suficientemente inteligente como para generar: "Comemos carne asada los domingos", es decir,
Debe escribirse con mayúscula cada palabra del nombre de una comida de varias palabras.
7. Itera sobre todos los días de la semana en una matriz y suma el número total de caracteres.
en todos los días de la semana. Por ejemplo, “domingo”=6, “lunes”=6, etc., por lo tanto, la respuesta debería
tener 50 años

8. Terry Martín

12 de enero de 2022 a las 16:35:58

Necesitará habilitar la opción MULTILINE en el motor de expresiones regulares de Java. Puede intentar generar la expresión regular pieza por pieza utilizando el comprobador de expresiones regulares en https://regex101.com. Si utiliza el comprobador de expresiones regulares, asegúrese de habilitarlo para “java” y tenga en cuenta que NO utiliza retrocesos dobles '\' como los necesarios en el IDE de Java.

8

8. Utilice Regex con parámetros nombrados y un bucle para analizar las direcciones a continuación e imprimir la dirección de la calle, la ciudad, el estado y el código postal. Transfiera esta lista de direcciones a su programa utilizando un bloque de texto de Java “”” String.

```
12345 Primera Calle, Primera Ciudad, AA 90210
22222 Segunda calle, Segunda ciudad, BB 22222
33333 Tercera Calle, Tercera Ciudad, CC 33333
44444 Calle Cuarta, Cuarta Ciudad, DD 44444
55555 Quinta Calle, Quinta Ciudad, EE 55555
66666 Sexta calle, Sexta ciudad, FF 66666
77777 Calle Séptima, Ciudad Séptima, GG 77777
88888 Calle Octava, Ciudad Octava, HH 88888
99999 Calle Novena, Ciudad Novena, II 99999
00000 Décima Calle, Décima Ciudad, JJ 00000
```

9. Terry Martín

14 de enero de 2022 a las 00:19:17

Probablemente necesitarás crear una matriz de todos los grupos de letras conocidos utilizados en la tabla de ejemplos, como: 'tr', 'fl', etc.

Luego, prueba cada palabra de la frase entrante, como “golpe aplastante”, para ver si comienza con algún grupo conocido. Si es así, quita el grupo de la palabra y guarda el grupo y la palabra eliminada restante para reutilizarla más adelante. Haz lo mismo con la segunda palabra también. Si una palabra no comienza con un grupo, simplemente quita la primera letra.

Reúne las dos palabras con el principio de la otra.

Pruebas

Ceremonias

1. Use TDD para escribir pruebas y un SUT (sistema bajo prueba, la clase de implementación real) para un método que toma una entrada de cadena y genera esa misma cadena con todas las demás letras en mayúscula. Por ejemplo, si ingresa "gato", devuelve "cAt". Si ingresa "manzana", devuelve "aManzana". Debería terminar con dos clases: una clase con el nuevo método y una
   Clase para las pruebas unitarias que prueban el nuevo método. La clase de prueba debe contener tantos métodos de prueba como sean necesarios para probar la implementación correcta.
2. Utilice TDD para escribir pruebas para un método que pueda recibir una entrada de cadena como:
   “Billy, Bob, 1234 Big St., Big City, California, 90210”
   Y devuelve un objeto que representa a esa persona. Puedes nombrar tu clase Ejercicio2 o Persona o lo que quieras. Para probar esto fácilmente, necesitarás que el IDE genere un método equals() y hashcode() en tu clase que modele una Persona (aprenderás más sobre eso en una sección posterior). Para hacerlo, puedes hacer clic derecho en una línea vacía dentro de tu clase y seleccionar, “Generar…” y hacer clic en “Siguiente” cada vez (probablemente serán tres veces) seguido de “Finalizar”. Ahora podrás continuar con tu(s) prueba(s).
   1. Modifique su método para aceptar una cadena como:
   “Billy, Bob, 1234 Big St., Big City, California, 90210 | Joe, Smith, 5678 Little St., Little Ciudad, Nueva York, 20109”
   Su nuevo método debe devolver una matriz de objetos que modelen una persona. Debe reutilizar la misma clase Persona que utilizó antes (independientemente del nombre que le haya dado). La entrada de texto Debe separar los detalles de las personas con el símbolo de barra vertical '|'. No debería haber límite en la cantidad de detalles de personas que puede pasar en una cadena. En la cadena de ejemplo anterior, presento solo dos personas, pero puede seguir agregando un símbolo | y más.
   Detalles de personas y deberías obtener una matriz con esa cantidad de objetos de personas. Intenta reutilizar lo que ya hiciste para la primera parte de este ejercicio y construye sobre esa base.
3. Utilice TDD para escribir un programa que tome una cadena de dos palabras y cree un "spoonerismo" de ellas. Un "spoonerismo " es cuando se transponen las primeras letras o fonemas de dos palabras.
   Un ejemplo sería “golpe aplastante ” -> “cuervo ruborizado ” o “mi culpa” -> “por enfado”.
   Estos ejemplos son para casos de uso de prueba:

ANTES	CUCHARREADO
Golpe aplastante	Cuervo ruborizado
Fragmento de sonido	Sitio enlazado

ANTES	CUCHARREADO
Gorra plana	Puerta para gatos
Balada triste	Mala ensalada
Plan maestro	Hombre de yeso
Snacks para el camino	Huellas de caracol

1. ¿Puedes hacer que funcione independientemente de la capitalización? (No hay una solución de ejemplo realizada para

Este... estás solo, hijo :-))

10. Terry Martín

15 de enero de 2022 a las 17:13:06

Recuerde que las enumeraciones son, en realidad, solo clases y, por lo tanto, la mayor parte de lo que se puede hacer con una clase también se puede hacer con una enumeración. En este caso, podría asociar una cadena de comida con cada constante de enumeración como una propiedad de la clase de enumeración.

11. Terry Martín

17 de enero de 2022 a las 15:41:42

Recuerde que todos los caracteres tienen un número asociado. Por ejemplo, A mayúscula = 65 en ASCII y Unicode. Por lo tanto, está convirtiendo de una letra a un número.
carácter, a un número, luego de un número, a otro.

Ceremonias

1. Utilice una enumeración para modelar los días de la semana e imprímalos usando un bucle.
    1. Imprimirlos con la primera letra en mayúscula (sin cambiar las constantes de enumeración).
    2. Alterne entre imprimir la primera letra en mayúscula o cualquier letra que esté aproximadamente en medio de la palabra. Por ejemplo: domingo, lunes, martes, miércoles, etc.
    3. Imprime 10 días de la semana elegidos al azar

10

2. Escriba el código para imprimir lo siguiente utilizando solo una enumeración y un bucle sin usar ningún condicional (si/de lo contrario/cambiar/etc.)

    Comemos carne asada los domingos
    Comemos espaguetis el lunes
    Comemos tacos los martes
    Comemos pollo el miércoles
    Comemos pastel de carne el jueves.
    Comemos hamburguesas el viernes
    Comemos pizza el sábado

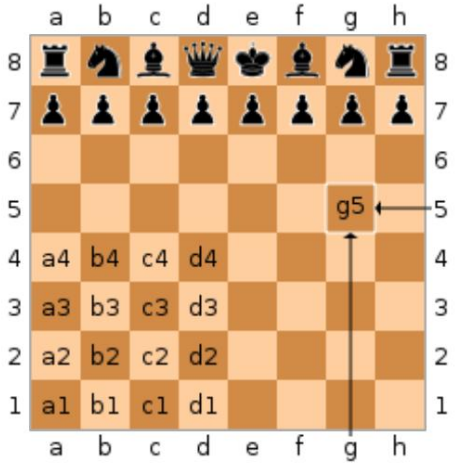
    1. Hazlo de nuevo, pero escribe con mayúscula también los nombres de las comidas. “carne asada” debería convertirse en “carne asada”. (No se proporciona ninguna solución porque ya lo has hecho en un ejercicio anterior, pero intenta hacerlo sin mirar trabajos anteriores, para practicar)
3. Escriba un método que tome la siguiente cadena de entrada (en negrita) y devuelva lo siguiente Cadena de salida usando una enumeración como (imprimir la salida):
    getMealsForDays(“viernes, jueves, lunes, sábado, martes”)
    Salida: hamburguesas, pastel de carne, espaguetis, etc…

11

4. Escriba un método que reciba una entrada de 1 letra minúscula (y solo 1 letra), como a - b - c - z, etc., y devuelva la posición ordinal de esa letra en el alfabeto, es decir, a = 1, b = 2, c = 3… z = 26. NO utilice ningún condicional (if/then/else/switch/etc.). No utilice una matriz. Escriba pruebas unitarias para probar las siguientes entradas: a, z, w, f, c, h, que deberían devolver: 1, 26, 23, 6, 3, 8 respectivamente
5. Escriba un método que haga lo opuesto al n.° 4: tomar un número, del 1 al 26, y devolver un letra minúscula. 1 devolvería a, 2 = b, 26 = z, etc.
6. Escriba un método que tome las siguientes entradas y devuelva las salidas correspondientes:
    8->0, 7->1, 6->2, 5->3, 4->4, 3->5, 2->6, 1->7
7. Si modelaras un tablero de ajedrez en Java usando una matriz bidimensional, ¿cómo podrías convertir la notación de ajedrez, o la notación de coordenadas del ajedrez, a un elemento en esa matriz bidimensional?
    ¿Matriz dimensional? Por ejemplo, en la imagen del tablero de ajedrez que aparece a continuación, la casilla a8 podría corresponder a

array[0][0] y h1 podrían asignarse a array[7][7]. Cree un método que tome una cadena de entrada de una coordenada de ajedrez y devuelva las coordenadas de la matriz. Este ejercicio simplemente se basa en los dos anteriores. Utilice los siguientes datos de prueba:

Aporte	Producción
a8	0,0
h1	7,7
g5	6,3
d4	3,4



8. Ahora, hagamos algo GRANDE. Modelemos parcialmente una partida de ajedrez. No te preocupes, no modelaremos una partida completa ni calcularemos el movimiento más eficiente. Modelaremos el tablero y al menos un par de piezas de ajedrez, como el peón y el caballo. Si no estás familiarizado
Para conocer los conceptos básicos del ajedrez, lea este artículo https://en.wikipedia.org/wiki/Chess#Movement.
Uno de los lugares donde la Programación Orientada a Objetos realmente brilla mejor es en el modelado.
Dominios con reglas o lógica complejas. Pocos juegos pueden llegar a ser tan complejos como el ajedrez.
Intentarás modelar cómo se permite que los peones y los caballos se muevan e interactúen con el tablero. Te proporcionaré pruebas JUnit que puedes copiar en tu propio proyecto. Luego necesitarás
para usar TDD para implementar las clases a las que se hace referencia en las pruebas y lograr que las pruebas pasen. Para aquellos que quieran intentar modelar estos ejercicios inicialmente sin ninguna ayuda,
Es posible que desee intentar no mirar las pruebas JUnit proporcionadas y crear las suyas propias basándose únicamente en las descripciones a continuación. Las pruebas proporcionadas le mostrarán cómo decidí modelar los conceptos de algunas maneras, aunque no verá cómo se implementaron sus métodos.
Eso te lo dejo a ti.
1. Deberás crear clases para: Tablero de ajedrez y Peón. Según la prueba JUnit,
canAddPawn(), escribe código para permitir que se agregue una instancia de la clase Pawn a una instancia del tablero de ajedrez en una casilla del tablero utilizando notación de ajedrez (a1, c3, etc.).
Luego, afirma que el peón está realmente en esa ubicación en el tablero llamando a un método en el tablero para recuperar una pieza por ubicación.

2. Haz lo mismo que en el paso 8.1, pero para un caballo, para que canAddKnight() funcione. El método que escribiste en el paso 8.1 para agregar un peón también debería poder aceptar un caballo. Ten en cuenta que necesitarás utilizar las técnicas orientadas a objetos que aprendiste en esta sección para obtenerlo.
   Todo esto funciona correctamente.
3. Cambiando a las pruebas JUnit en el archivo PawnTest.java, habilite la pawnCanMoveOneForward() para pasar la prueba implementando el código necesario. Aquí es donde realmente comenzarás a intentar implementar alguna lógica comercial (o de juego). El objetivo de esta prueba es simular la creación de un peón y colocarlo en una casilla y "pedirle" que haga algo.
   ¿Cuáles son todos los cuadrados válidos a los que podría moverse teóricamente? La prueba afirmará que uno de sus movimientos válidos es avanzar un cuadrado. Estos movimientos no tienen en cuenta si la casilla ya puede estar ocupada. Tenga en cuenta que los movimientos válidos de una pieza siempre deben ser relativos al lugar donde esa pieza comience. Por lo tanto, aunque proporciono ejemplos de posiciones iniciales/finales para muchas de las
   En los siguientes ejercicios, los movimientos deberían funcionar de manera uniforme sin importar dónde comiencen las piezas. Avanzar de A2 a A3 es lo mismo que hacerlo de B2 a B3 o E5.
   a e6, etc. No codifique los movimientos en su código de implementación, en otras palabras palabras.
4. pawnCanMoveTwoForwardOnFirstMove() - con suerte, esta prueba es obvia a partir de su nombre. Afirma que de todos los movimientos posibles que un peón podría hacer, si el peón nunca se ha movido antes (esto supone que comienza en su posición inicial estándar de un juego), uno de sus movimientos permitidos es moverse dos casillas hacia adelante (de nuevo, ignorando la posibilidad de que otras piezas se interpongan en su camino).
5. Obtenga la prueba canNotMoveTwoAfterFirstMove() . Esta prueba confirma que después de un peón ha realizado su primer movimiento, si le “preguntas” cuáles son sus próximos movimientos válidos, avanzar dos casillas no será una de sus opciones.
6. Obtenga la prueba canMoveOneDiagonallyRight() que pasa. Esto confirma que uno de los peones
   Los movimientos válidos son poder moverse en diagonal hacia adelante y hacia la derecha. Aunque esto solo está permitido cuando el peón puede capturar una pieza, no lo tendremos en cuenta por ahora.
7. Obtenga la función de prueba canMoveOneDiagonallyLeft() . Igual que 8.6 pero hacia la izquierda.
8. Haga que blackPawnCanMoveForward() pase. Suponiendo que todas las pruebas de peón anteriores fueron
   Para un peón blanco que se mueve “hacia arriba” en el tablero, eso significaría que un peón negro moverse en la dirección opuesta en el tablero. Si vinculamos la dirección del movimiento al color de la pieza, entonces esta prueba confirmará que los movimientos permitidos de una pieza negra
   El peón se moverá “hacia abajo” en el tablero, hacia las piezas blancas en general. Si un peón negro está en la casilla a7 y avanza una casilla (en relación con su lado del tablero), entonces debería estar en la casilla a6.

9. Obtenga blackPawnCanMoveTwoForwardOnFirstMove() . Solo la versión del peón negro de 8.4. Si el peón estaba en b7, en este caso terminaría en b5.
10. Pasando al archivo KnightTest.java, la primera prueba que pasa es knightCanMoveNorthEast(). La idea aquí es afirmar que uno de los movimientos válidos del caballo es moverse dos pasos hacia adelante (Norte) y un paso a la derecha (Este) desde donde comienza. Entonces, si comienza en la casilla c1, terminaría en d3. Elegí incluir caballos para ser modelados porque su movimiento es bastante extraño y considerablemente diferente de todas las demás piezas de ajedrez. Esto puede obligarlo a pensar realmente en esos movimientos orientados a objetos. ideas ;-) .
11. knightCanMoveNorthWest() - desde c1, el caballo terminaría en b3.
12. knightCanMoveEastNorth() - desde c1, el caballo terminaría en e2.
13. knightCanMoveEastSouth() - desde c3, el caballo terminaría en e2.
14. knightCanMoveWestNorth() - desde c3, el caballo terminaría en a4.
15. knightCanMoveWestSouth() - desde c3, el caballo terminaría en a2.
16. knightCanMoveSouthEast() - desde c3, el caballo terminaría en d1.
17. knightCanMoveSouthWest() - desde c3, el caballo terminaría en b1.
18. blackKnightCanMoveSouthEast() - desde d5, el caballo terminaría en c7.
19. Ahora que has enseñado a tus clases de Peón y Caballo cómo se les permite moverse, vamos a permitir que la clase de Tablero de Ajedrez los mueva. Volviendo a las pruebas para la clase de Tablero de Ajedrez, Tablero de ajedrez, archivo ChessBoardTest.java (si se basa en los archivos proporcionados), por favor Implementa la capacidad canMoveC1KnightToD3(). La prueba creará un tablero de ajedrez y agrega un caballo en c1. Afirmará que el caballo está en la casilla c1. Luego, "solicitará" al tablero de ajedrez que mueva el caballo a d3. A continuación, afirmará que el caballo está en la casilla c1. ya no está en c1 y ahora está en d3.
20. Según canNotMoveC1KnightToInvalidSquare(), implemente la capacidad de evitar que el caballo se mueva a casillas no válidas, como de c1 a d4. En este caso, el caballo simplemente debería permanecer en c1 (por ahora). Afirme que el caballo comenzó en c1, luego Se produjo el intento de movimiento. Luego afirma que el caballo NO está en d4 (como se solicita) pero todavía está en c1 en este caso.
21. Según canNotMoveC1KnightToFriendlyOccupiedSquare(), implemente la capacidad de evitar que el caballo sea movido a una casilla que ya está ocupada por otro pieza de ajedrez del mismo color (una amiga). Entonces, si un caballo comienza en c1 y hay un peón A partir de d3, el caballo no puede moverse a d3 si el caballo y el peón son ambos iguales. color.

22. Según canMoveC1KnightToEnemyOccupiedSquare(), implementa la capacidad de permitir que el caballo se mueva a una casilla válida que ya esté ocupada por una pieza enemiga (una pieza del color opuesto). Afirma que la pieza que estás moviendo ya no está en su lugar. Posición inicial. Afirmar que termina en la casilla de destino. Afirmar que la pieza enemiga ha sido “capturada” al haber sido movida hacia una matriz de piezas capturadas de las blancas.

12. Terry Martín
7 de febrero de 2022 a las 11:42:01
Puedes utilizar la interfaz Comparable existente que probablemente hayas implementado en la clase Car. Luego, simplemente llama a Collections.sort(cars).

13. Terry Martín
7 de febrero de 2022 a las 11:48:47
Para ello, ni siquiera es necesario implementar Comparable en la clase Car. En su lugar, puede llamar a Collections.sort(cars, Comparator) y pasar una instancia de Comparator como segundo parámetro a sort(), o una clase anónima o una expresión Lambda.

Colecciones

Ceremonias

1. Escriba un código que le permita modelar y almacenar una colección de al menos 5 automóviles y mantenerlos en el orden en que fueron ingresados. Imprímalos también en la pantalla.
2. Igual que el ejercicio 1 excepto que no nos importa conservar el orden y queremos asegurarnos que no existirán duplicados.
3. Igual que el ejercicio 1 , pero asocie el nombre del propietario a cada automóvil. No agregue el nombre del propietario. Nombre de la clase del modelo de su automóvil. Imprima cada propietario con su automóvil. Ejemplo:

Chelín	Coche[marca=Tesla, modelo=X, año=2015]
Jenny	Coche[marca=Tesla, modelo=Y, año=2016]
Sara	Coche[marca=Tesla, modelo=3, año=2019]

4. Si implementó el ejercicio 2 con un registro (en lugar de una clase), hágalo nuevamente usando una clase o viceversa.
5. Permitir que los autos del ejercicio 2 se impriman en orden “natural” por modelo.
   1. Almacene los mismos vehículos en una lista y ordénelos. (No se proporciona ninguna solución de código)
   2. Almacene los mismos autos en una lista y ordénelos sin implementar ninguna interfaz en la lista. Clase de coche.
   3. ¿Cómo podría invertir el orden de clasificación?
6. Igual que el ejercicio 5, pero permite que el programa elimine un modelo pasando el nombre del modelo como un argumento para el método main(). Por ejemplo, si hubiera obtenido el siguiente resultado inicialmente:

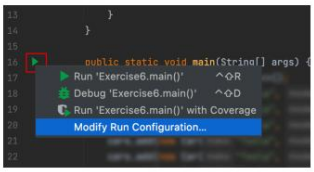
Coche[marca=Tesla, modelo=3, año=2016]
Coche[marca=Tesla, modelo=Roadster, año=2009]
Coche[marca=Tesla, modelo=S, año=2014]
Coche[marca=Tesla, modelo=X, año=2015]
Coche[marca=Tesla, modelo=Y, año=2017]

Y luego modificas el programa según las instrucciones y pasas “Roadster” como argumento y lo vuelves a ejecutar, obtendrás el siguiente resultado:

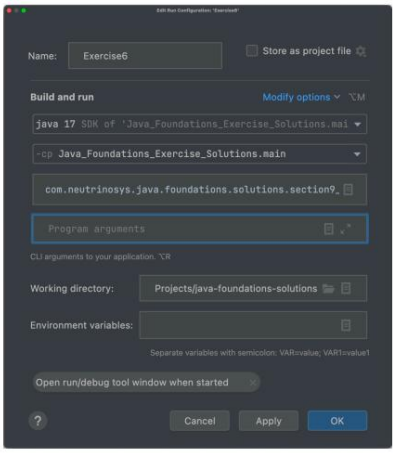
Coche[marca=Tesla, modelo=3, año=2016]
Coche[marca=Tesla, modelo=S, año=2014]
Coche[marca=Tesla, modelo=X, año=2015]
Coche[marca=Tesla, modelo=Y, año=2017]

A continuación se muestran las instrucciones sobre cómo pasar argumentos.

1. Haga clic derecho en el triángulo verde



2. Seleccione “Modificar ejecución…”



3. Ingrese el nombre del modelo a eliminar Campo “Argumentos del programa” y haga clic "DE ACUERDO"

14. Terry Martín

7 de febrero de 2022 a las 10:59:37

Consideraría seriamente usar una lista enlazada. En primer lugar, debes conservar el orden en el que llegan (a ti), por lo que necesitarás algún tipo de lista. En segundo lugar, estás tratando con millones o miles de millones. ArrayList funcionaría PERO preasigna grandes cantidades de memoria cada vez que necesita crear más espacio. Esto puede ser ineficiente y desperdiciar memoria temporalmente. Una lista enlazada puede crecer dinámicamente de manera muy eficiente.

15. Terry Martín

7 de febrero de 2022 a las 13:16:02

La matriz bidimensional del tablero de ajedrez interno es la única matriz que debería considerar conservar. Todas las demás deberían poder reemplazarse por una colección. Además, todos los métodos que escribimos explícitamente para agregar, quitar y buscar en una matriz deberían simplificarse o eliminarse en gran medida porque las clases de Colecciones tienen estas capacidades integradas.

14

7. ¿Cuál podría ser una colección más eficiente en términos de memoria para usar para almacenar millones o miles de millones de objetos en el orden en que aparecen, si no sabe exactamente cuántos objetos necesitará? ¿Es necesario almacenarlo de antemano?

8. ¿Cómo podrías tomar tu colección de autos del ejercicio 6 y almacenarlos en una matriz de manera eficiente? ¿Cómo podrías convertir esa matriz de autos nuevamente en una lista? (La solución es incluido en la solución del ejercicio 6)

15

9. Ahora que ya conoce las colecciones de Java, intente revisar el código de ajedrez de los ejercicios de la sección 8 para usar colecciones en lugar de matrices (excepto para el tablero de ajedrez interno). Representación). No hay ninguna solución de código disponible en este momento

16. Terry Martín

7 de febrero de 2022 a las 13:36:43

Stream.of(nuevo Auto(), nuevo Auto(),… etc.)

Flujos y Lambdas

Ceremonias

1. Utilizando el mismo enfoque de colección de automóviles que en los ejercicios de la sección anterior, cree una colección de automóviles pero use la API Streams para imprimir solo los nombres de los modelos de los automóviles.
2. ¿Cómo se podría utilizar la API Streams para filtrar (no mostrar) todos los coches fabricados antes de un determinado año? ¿año? (Si modeló el año del automóvil utilizando la clase de fecha y hora Año, tiene un método isAfter())
3. ¿Cómo podría crear un flujo de automóviles sin crear primero explícitamente una colección de automóviles?
4. Agregue un campo de precio entero adicional a su clase/registro de automóvil. Use Streams para agregar el costo total de todos los autos en tu colección/stream. (Hay dos formas de hacer esto)
    1. Use Streams para encontrar el precio promedio de todos los autos (hay dos formas de hacerlo también): sin código proporcionó
    2. ¿Cómo podrías hacer el ejercicio 4 con Streams y BigDecimal (para precisión decimal)?
    3. ¿Cómo podría hacer el ejercicio 4.2 pero generar una cadena de dinero formateada utilizando solo el ¿API de Streams? Solución que se muestra en la solución 4.2
5. Utilice la API Streams para ordenar un flujo de automóviles en orden inverso por modelo
    1. Hazlo ordenando por marca, luego por modelo y luego por año, todo en uno.
6. Utilice únicamente la API Streams y una colección o flujo de objetos Car para producir el siguiente salida: “S, X, 3, Y, Roadster”
7. Agrega más marcas diferentes de autos a tu colección de autos y luego

Hacer	Modelo	Año	Precio
Tesla	S	2014	$90,000.99
Tesla	...	2015	$110,000.99
Tesla	3	2016	$55,000.99
Tesla	Y	2017	$60,000.99
Tesla	Coche de turismo	2009	$135,000.99
Lúcido	Aire	2021	$77,399.99
Hyundai	Iónico	2021	$34,250.00
Hyundai	Kona	2021	$38,575.00
Porsche	Taycan	2021	$81,250.00
Audi	e-tron	2021	$66,995.00

Hacer	Modelo	Año	Precio
Volkswagen	Identificación 4	2021	$41,190.00

1. Determinar el precio total por marca.
2. Determinar el precio medio del coche por marca
    1. Este será divertido y desafiante. Te recomendamos echarle un vistazo a Collectors.teeing() función, que no enseñé explícitamente, pero eche un vistazo a su javadoc. Si esto demuestra ser demasiado difícil pero aún así quieres intentarlo sin mirar mi solución, será MUCHO más fácil si convierte los precios a cualquier otro tipo numérico antes de usarlos. API de transmisiones (pero no tan divertidas…)
    2. ¿Ya entendiste la versión 7.2.1? Hagámoslo aún más divertido. Formatee los precios promedio para moneda, dentro de la API Streams.
3. Determinar el número de coches por año y luego por marca.
4. Determine la cantidad de automóviles por marca creando primero un nuevo mapa vacío y luego iterando sobre la colección de coches (que has estado usando para ejercicios anteriores) y usando un método funcional de la interfaz de Mapa para rellenar su Mapa vacío.

17. Terry Martín

7 de febrero de 2022 a las 20:50:23

Crearía una subclase personalizada de RuntimeException y la arrojaría en la línea 27 de ChessBoard en lugar de simplemente devolverla como lo hace actualmente. También pasaría a su constructor información sobre las coordenadas de destino a las que intentamos movernos y tal vez incluso qué pieza ya está allí para que esa información pueda transmitirse en el extremo receptor de la excepción.

Cabos sueltos

Ceremonias

17

1. Revise el modelo de ajedrez de la Sección 8, ejercicio 8 para generar una excepción cuando una pieza de ajedrez intenta moverse a una casilla ocupada por una pieza amiga. Considere qué es una Tipo de excepción adecuado para esto. No se proporcionó ninguna solución de código.
2. Intente utilizar valores opcionales en el ejercicio de modelado de ajedrez en lugar de valores nulos en todo el código.
3. Cree una colección de al menos cinco objetos Persona con campos de nombre y apellido. Agregue algunos objetos nulos a la colección en lugares aleatorios. Use un bucle for mejorado para iterar sobre la colección e imprimir los nombres de cada persona. Utilice Opcional para evitar NullPointerException y para imprimir "Desconocido" para los nombres de pila de los elementos nulos.
    1. Utilice la API opcional para mostrar "Desconocido" para entradas nulas y entradas cuyo nombre Tiene menos de 3 caracteres.
4. Suponiendo que haya capturado la entrada del usuario de una fecha/hora en una cadena como "12 de julio de 1984 13:47:00" y suponiendo que la fecha/hora ocurrió en GMT-8, escriba el código para analizar esa cadena y convertirlo a GMT-0.
    1. ¿Cuál sería esa misma fecha/hora después de 179 días, 7 horas y 27 minutos en Japón?
5. Escribe el código para determinar la edad actual de tu propio país.
    1. Escribe el código para determinar tu edad con exactitud. No se proporciona ningún código
    2. Escribe el código para determinar cuántos días faltan para el próximo Año Nuevo (o cualquier día festivo que desees). elegir). Nota/pista: ChronoUnit tiene enumeraciones para varios intervalos de tiempo que también tienen métodos útiles.
    3. Si comenzaste una actividad a las 9:37:20 y terminaste a las 19:13:41, ¿cómo puedes escribir mejor? ¿Código para determinar cuánto tiempo transcurrido?
        1. ¿Qué pasa si quieres saber el tiempo transcurrido en minutos?
6. ¿Cuánto tiempo habrá transcurrido si sales de Seúl, Corea del Sur a las 13:15:00 del 3 de febrero? 2022 y llegará a Londres, Reino Unido a las 20:02:13, 3 de febrero de 2022?
    1. Cuando llegues a Londres, ¿qué hora será en Los Ángeles, California?

7. Escriba el código para analizar cada una de estas fechas e imprimir el objeto de fecha/hora correspondiente. Proporciono soluciones de código para analizar los puntos 1 a 5 y 13 en la tabla aquí. El resto debería ser bastante similar a aquellos.

09-02-2022 05:02:01Z
09-02-2022 05:02:01 +0000
Mié, 09 Feb 2022 05:02:01 +0000
Miércoles, 09-Feb-22 05:02:01 UTC
Mié, 09 Feb 22 05:02:01 +0000
Mié, 09 Feb 2022 05:02:01 +0000
Mié, 09 Feb 22 05:02:01 +0000
09-02-2022 05:02:01 +00:00
09-02-2022 05:02:01 +00:00
miércoles, 09-feb-2022 05:02:01 UTC
Mié, 09 Feb 2022 05:02:01 +0000
09-02-2022 05:02:01 +00:00
1644382921
2 de septiembre de 2022 05:02:01
2 de septiembre de 2022 05:02:01 a. m.
09-02-2022 05:02:01
09-02-2022 05:02:01