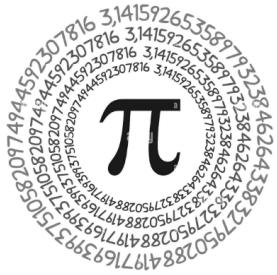
Comenzado el	viernes, 16 de diciembre de 2022, 16:39
Estado	Finalizado
Finalizado en	viernes, 16 de diciembre de 2022, 19:05
Tiempo empleado	2 horas 25 minutos
Puntos	1,40/5,00
Calificación	**2,80** de 10,00 (**28**%)

Pregunta **1**

Sin contestar

Puntúa como 1,00

Tres veces el diámetro y un poquito más ¡En kotlin!



El 14 de Marzo es el día internacional de PI. Todos sabemos que pi vale 3,14... De ahí la elección del día como Marzo(mes 3) y día 14. Pero, ¿qué significado tiene realmente este número?

Para que los niños entendieran lo que es pi, un magnífico y genial maestro, un maestro de los de antes, explicaba a sus alumnos lo siguiente:

PI es el número 3,141516... pero no lo veas como un número extraño y caprichoso, es un número que describe que la relación entre el perímetro de una circunferencia cualquiera y su radio siempre es la misma. Da igual que la circunferencia sea enorme o pequeñísima, si divides el perímetro de la circunferencia entre su diámetro siempre obtendrás la misma cantidad: el número PI.

Y explicaba de nuevo el maestro:

Imagina que rodeas una tapa circular con un hilo o alambre. Tienes una circunferencia de hilo. Mides su radio r , y por lo tanto sabes que su diámetro es 2r. si esa circunferencia de hilo la cortas y a continuación la mides, estas midiendo su perímetro, pues bien puedes observar que el perímetro va a ser 3 veces el diámetro y un poquito más. Pruebas esto con muchas circunferencias y siempre ocurre los mismo, que el perímetro es "Tres veces el diámetro y un poquito más"

Perímetro

—————— = 3 y un poquito más

diámetro

El poquito más es 0,141516.... y no sabés el trabajo que le dio a matemáticos de todos los tiempos determinar cada vez con más precisión ese poquito más.

Aprovechamos entonces las enseñanzas del sabio maestro y estipulamos el poquito más(*pm*) de la siguiente forma:

0.1415<=pm<=0.1417

y pretendemos saber si dados dos números p>0 y d>0, que representan respectivamente al perímetro y al diámetro de una supuesta circunferencia, son valores que pertenecen realmente a una circunferencia o no.

ENTRADA

La entrada comienza con una línea indicando el número de casos que pueden ser 0 o más, a continuación cada caso se específica en una línea con dos números separados por un espacio en blanco. El primer número representa el perímetro y el segundo el diámetro.

SALIDA

La salida indicará para cada caso de entrada si p y d pertenecen a una circunferencia con la frase ES CIRCUNFERENCIA, o en caso contrario con la frase PARECE UN HUEVO

Para usar un compilador kotlin sigue las instrucciones del profesor, en principio puedes utilizar el IntelliJ de tu equio o bien usar el siguiente compilador en linea: https://www.jdoodle.com/compile-kotlin-online/

EJEMPLO DE ENTRADA/SALIDA

```
5
31.416 10
ES CIRCUNFERENCIA
31.416 12
PARECE UN HUEVO
6.2832 2
ES CIRCUNFERENCIA
31.416 10.01
PARECE UN HUEVO
15.708 5
ES CIRCUNFERENCIA
```

Ten en cuenta: para obtener la máxima puntuación debes poder indicar en la misma línea, separados por un espacio en blanco, el perímetro y el diámetro como en el ejemplo anterior.

Recuerda también que quizá en tu equipo la introducción por teclado de números con decimales en lugar de punto decimal requiera coma decimal.

Para comprobar el ćodigo puedes utilizar IntelliJ. Si hubiera problemas con este IDE y con la autorización del profesor podrías usar el siguiente IDE online

https://www.jdoodle.com/compile-kotlin-online/

Pregunta **2**

Correcta

Se puntúa 1,00 sobre 1,00

Dibujar el perímetro de un un cuadrado

Crea un **método estático** denominado cuadrado**(l, c)** que dibuje el perímetro (borde) de un cuadrado de lado l y con caracter de dibujo c. Se garantiza que l es un int y que l>=2. Para que el dibujo sea un cuadrado aceptablemente proporcionado los caracteres de la misma linea tiene un espacio en blanco intercalado. Debe resolverse el problema imprimiendo caracter a caracter sin la ayuda de los métodos de la clase String, StringBuilder etc.

Por ejemplo:

Test	Resultado

`cuadrado(5, '#')`	`# # # # #`
	`#  #`
	`#  #`
	`#  #`
	`# # # # #`

`cuadrado(2, '*')`	`* *` `* *`

Respuesta: (sistema de penalización: 0 %)

```
1  public static void cuadrado(int l, char c){
2          for(int i= 0; i < l; i++){
3              System.out.print(c + " ");
4          }
5          System.out.println();
6
7          for(int i= 0; i < l-2; i++){
8              for(int j= 0; j < l; j++){
9                  if(j == 0 || j == l-1){
10                     System.out.print(c + " ");
11                 }else{
12                     System.out.print("  ");
13                 }
14
15             }
16             System.out.println();
17         }
18
19         for(int i= 0; i < l; i++){
20             System.out.print(c + " ");
21         }
22     }
```

	Test	Esperado	Se obtuvo	

✔	`cuadrado(5, '#')`	`# # # # #`
		`#  #`
		`#  #`
		`#  #`
		`# # # # #`
			`# # # # #`
			`#  #`
			`#  #`
			`#  #`
			`# # # # #`
				✔

✔	`cuadrado(2, '*')`	`* *` `* *`	`* *` `* *`	✔

✔	`cuadrado(6, '@')`	`@ @ @ @ @ @`
		`@  @`
		`@  @`
		`@  @`
		`@  @`
		`@ @ @ @ @ @`
			`@ @ @ @ @ @`
			`@  @`
			`@  @`
			`@  @`
			`@  @`
			`@ @ @ @ @ @`
				✔

Todas las pruebas superadas. ✔

Correcta

Puntos para este envío: 1,00/1,00.

Pregunta **3**

Sin contestar

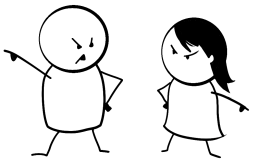
Se puntúa 0,00 sobre 1,00

Picos y Valles

Mini y Max son inseparables desde la infancia aunque, en realidad, son muy diferentes. Cuando uno sube, la otra baja. Si para uno es blanco, para la otra es negro.

Bueno, ya se sabe lo que se dice de los extremos opuestos...

Si hay algo que tienen en común es su afición por el montañismo. A ambos les encanta subir a lo alto de una cima y otear el horizonte que los rodea. Eso sí, mientras uno deja volar su imaginación al tiempo que divisa las altas cumbres circundantes, a la otra se le van los ojos a los verdes y profundos valles. Qué se le va a hacer...



Ahora bien, dada la lista de las cotas (alturas sobre el nivel del mar) del terreno que tienen ante ellos, ¿serías capaz de contar el número total de picos y valles que ven? Se entiende por "pico" a una cota más alta que las inmediatamente anterior y posterior. Y "valle" si es más baja.

Ten en cuenta que, dado que la lista de cotas abarca los 360º de horizonte a su alrededor (o sea, es circular), tanto la primera como la última cota ¡podrían ser un pico o un valle!

Entrada

El programa leerá de la entrada estándar múltiples casos de prueba, cada uno con una lista de cotas.

Cada caso de prueba contendrá una primera línea en la que se indicará el número de cotas ($2 \leq n \leq 1.000$). A continuación, en la siguiente línea, vendrán n números positivos (menores que 10.000) con las diferentes cotas.

La entrada termina con un caso de prueba sin cotas (n=0) que no debe procesarse.

Salida

Por cada caso de prueba, el programa escribirá el número total de picos y valles que hay.

Por ejemplo:

Entrada	Resultado
4 400 1000 350 250 4 1000 350 250 400 5 400 1000 1000 500 400 0	2 2 0

Respuesta: (sistema de penalización: 0 %)

```
1
```

Pregunta **4**

Parcialmente correcta

Se puntúa 0,40 sobre 1,00

Veni, vidi, vici

En criptografía, el cifrado César es una de las técnicas de cifrado más simples y más usadas. Es un tipo de cifrado por sustitución en el que una letra en el texto original es reemplazada por otra letra que se encuentra un número fijo de posiciones más adelante en el alfabeto. Por ejemplo, con un desplazamiento de **3**, la A sería sustituida por la D (situada 3 lugares a la derecha de la A), la B sería reemplazada por la E, etc. Para descifrar el mensaje, no tenemos más que realizar el mismo desplazamiento pero en sentido contrario. Este método debe su nombre a Julio César, que lo usaba para comunicarse con sus generales.

Deberás implementar un **método** denominado **cesar** que aceptará **dos parámetros**: el texto a cifrar y el desplazamiento. El método **devolverá** el texto cifrado en **mayúsculas**. Se **omitirá** el cifrado de cualquier carácter no alfabético (dígitos, signos de puntuación, espacios,...), que se **mantendrán** en el texto tal cuál están.

Debes tener en cuenta que el alfabeto de sustitución es **rotativo**. Es decir, con el desplazamiento de 3 caracteres del ejemplo anterior, una X se convertiría en una A, una Y en una B y una Z en una C.

Con objeto de simplificar el problema, se garantiza:

- que las entradas sólo contienen caracteres del alfabeto inglés, aunque podría haber mayúsculas y minúsculas
- un desplazamiento d tal que -26<=d<=26

Por ejemplo:

Test	Resultado
`cesar("Alea iacta est", 3)`	`DOHD LDFWD HVW`
`cesar("Ave, Caesar, morituri te salutant", 7)`	`HCL, JHLZHY, TVYPABYP AL ZHSBAHUA`

Respuesta: (sistema de penalización: 0 %)

```
1  ▾    private static String cesar(String msg, int dist) {
2        String input = msg.toUpperCase();
3        String result = "";
4  ▾     for (int i = 0; i < input.length(); i++) {
5  ▾         if (input.charAt(i) < 'A' || input.charAt(i) > 'Z'
6              result += input.charAt(i);
7  ▾         } else {
8                 result += (char) (input.charAt(i) + dist);
9
10
11           }
12       }
13       return result;
14   }
15
```

	Test	Esperado	Se obtuvo	
✔	`cesar("Alea iacta est", 3)`	`DOHD LDFWD HVW`	`DOHD LDFWD HVW`	✔
✘	`cesar("Ave, Caesar, morituri te salutant", 7)`	`HCL, JHLZHY, TVYPABYP AL ZHSBAHUA`	`H]L, JHLZHY, TVYP[\YP [L ZHS\[HU[`	✘
✔	`cesar("VW VCODKGP, DTWVQ?", -2)`	`TU TAMBIEN, BRUTO?`	`TU TAMBIEN, BRUTO?`	✔
✘	`cesar("a",-1)`	`Z`	`@`	✘
✘	`cesar("a",-26)`	`A`	`'`	✘

Mostrar diferencias

Parcialmente correcta

Puntos para este envío: 0,40/1,00.

Pregunta **5**

Incorrecta

Se puntúa 0,00 sobre 1,00

MyScanner

Se trata de escribir una clase Java que llamaremos *MyScanner*, una versión sencilla de la famosa clase *Scanner* del API java.

Nuestros objetos MyScanner sólo trabajan con los Strings introducidos en el constructor. No trabaja con teclado, ni ficheros. Los caracteres delimitadores que utiliza para hacer su trabajo son los mismos que usa por defecto la clase Scanner: \n \t y ' ' .

La clase MyScanner consta de dos atributos

```
class MyScanner {
   private int pos;
   private String datos;
   ..... resto de clase .....
}
```

La solución debe basarse en el recorrido y tratamiento del String datos, por descontado, no se permite utilizar la clase Scanner en el interior de MyScanner para solucionar el problema. . MyScanner es un escaner que avanza en su análisis de caracter en caracter de forma que en el atributo datos se almacena el String sobre el que se produce el escaneo y en pos se almacena la posición del siguiente caracter a procesar.

MyScanner consta de los siguientes métodos:

- *getPos().* Devuelve el entero almacenado por el atributo pos.
- *getDatos().* Devuelve el String almacenado en Datos que es justamente el String que se indica al constructor. Este es un valor constante a lo largo de la vida del objeto.
- *hasNextLine().* Devuelve true si quedan más caracteres por escanear que constituyan una línea . En caso contrario devuelve false. Si entre la posición actual del escaneo y el final del String hay al menos un caracter siempre se va a poder devolver un String que se considera una línea, aunque sea el string vacío.
- *hasNext().* Devuelve true si queda algun token en los datos por analizar, es decir, sihay algun caracter no delimitador entre pos y datos.length(). En caso contrario devuelve false
- *nextLine(). D*evuelve un String que se corresponde con la línea leida. Recuerda la posibilidad de que nextLine() devuelva el string vacío "" cuando se encuentra inmediatamente un \n.
- *next(). D*evuelve un String que se corresponde con el token leido. Token es un conjunto caracteres contiguos delimitador por un caracter delimitador que no se considera parte del token.
- nextInt(). Devuelve un int que se corresponde con el token leido.

Para poder pasar los test, previamente tienes que evitar los errores de compilación. Si no declaras todos los métodos del enunciado, que son métodos usados por los test se provocará un error de compilación sin llegar a comprobarse ni siquiera el primer test. Por lo tanto lo primero que tienes que hacer es escribir un armazón de la clase indicando los atributos y a continuación al menos los métodos indicados en el enunciado. Para cada método tienes que declarar como mínimo el tipo de datos que devuelve y una instrucción return devolviendo un valor de dicho tipo. Luego, a medida que avances en la pregunta iras esribiendo el código detallado de cada método. Por ejemplo, no sé resolver el método next() o todavía no empecé a solucionarlo porque estoy con otras partes de la pregunta, pero escribo

```
String next(){
   return "nolohice",//cualquier String
}
```

Ten encuenta, que el funcionamiento de next(), nextInt() y nextLine() es similar al de la clase Scanner **que se asume que conoces y entiendes**. Observa en los ejemplos, que si nextLine() y next() en su procesamiento de caracteres, llegan al final del string datos, devuelve los caracteres correspondientes aun que no hubieran llegado a un delimitador,. Así es además como funcione el Scanner del API java.

Por ejemplo:

Test	**Resultado**

`MyScanner ms=new MyScanner("hola a todas");`
`System.out.println(ms.getPos());`
`System.out.println(ms.getDatos());`
	`0`
	`hola a todas`

`MyScanner ms=new MyScanner("hola a todas\n y hola a todos");`
`System.out.println(ms.getPos());`
`System.out.println(ms.getDatos());`
	`0`
	`hola a todas`
	`  y hola a todos`

`MyScanner ms=new MyScanner("");//String vacio`
`System.out.println(ms.getPos());`
`System.out.println(ms.hasNextLine());`
	`0`
	`false`

Test	Resultado

`MyScanner ms=new MyScanner("hola a todos\ny adios");`
`ms.nextLine();`
`System.out.println(ms.getPos());`
`System.out.println(ms.getDatos());`
	`13`
	`hola a todos`
	`y adios`

`MyScanner ms=new MyScanner("hola a todas\n y hola a todos");`
`System.out.println(ms.nextLine());`
`System.out.println(ms.nextLine());`
`System.out.println(ms.getPos());`
`System.out.println(ms.hasNextLine());`
	`hola a todas`
	` y hola a todos`
	`28`
	`false`

`MyScanner ms=new MyScanner("hola a todas\n y hola a todos");`
`System.out.println(ms.getPos());`
`System.out.println(ms.hasNextLine());`
	`0`
	`true`

`MyScanner ms=new MyScanner("hola a todas\n y hola a todos");`
`System.out.println(ms.nextLine());`
`System.out.println(ms.nextLine());`
`System.out.println(ms.hasNextLine());`
	`hola a todas`
	` y hola a todos`
	`false`

`MyScanner ms=new MyScanner("hola a todas\n y hola a todos");`
`System.out.println(ms.nextLine());`
`System.out.println(ms.getPos());`
`System.out.println(ms.getDatos());`
	`hola a todas`
	`13`
	`hola a todas`
	` y hola a todos`

`MyScanner ms=new MyScanner("hola a todos");`
`while(ms.hasNextLine()){`
`  System.out.println(ms.nextLine());`
`}`
	`hola a todos`

`MyScanner ms=new MyScanner("hola a todas \ny también hola a todos");`
`while(ms.hasNextLine()){`
`  System.out.println(ms.nextLine());`
`}`
	`hola a todas`
	`y también hola a todos`

`MyScanner ms=new MyScanner("hola a todas \ny también hola a todos\n");`
`while(ms.hasNextLine()){`
`  System.out.println(ms.nextLine());`
`}`
	`hola a todas`
	`y también hola a todos`

`MyScanner ms=new MyScanner("hola a todas \n\ny también hola a todos\n");`
`while(ms.hasNextLine()){`
`  System.out.println(ms.nextLine());`
`}`
	`hola a todas`
	
	`y también hola a todos`

`MyScanner ms=new MyScanner("hola a todos");`
`System.out.println(ms.getPos());`
`System.out.println(ms.getDatos());`
	`0`
	`hola a todos`

`MyScanner ms=new MyScanner("hola a todos");`
`System.out.println(ms.getPos());`
`System.out.println(ms.next());`
`System.out.println(ms.next());`
`System.out.println(ms.getPos());`
	`0`
	`hola`
	`a`
	`6`

`MyScanner ms=new MyScanner("2 3 4");`
`int suma=0;`
`while(ms. hasNext()) suma+=ms.nextInt();`
`System.out.println(suma);`
	`9`

`MyScanner ms=new MyScanner("2 \t3 \n4\n");`
`int suma=0;`
`while(ms. hasNext()) suma+=ms.nextInt();`
`System.out.println(suma);`
	`9`

`MyScanner ms=new MyScanner("2 3 4\n\t");`
`int suma=0;`
`while(ms. hasNext()) suma+=ms.nextInt();`
`System.out.println(suma);`
	`9`

`MyScanner ms=new MyScanner("2 \nmi mama me mima");`
`System.out.println(ms.nextInt());`
`ms.nextLine();//limpiar enter`
`System.out.println(ms.nextLine());`
	`2`
	`mi mama me mima`

Respuesta: (sistema de penalización: 0 %)

```
1 class MyScanner {
```

Syntax Error(s)

```
__tester__.java:24: error: invalid method declaration; return type required
nextLine(). Devuelve un String que se corresponde con  la línea leida.  Recuerda la posibilidad de que nextLine()
devuelva el string vacío "" cuando se encuentra inmediatamente un \n.
^
__tester__.java:24: error: ';' expected
nextLine(). Devuelve un String que se corresponde con  la línea leida.  Recuerda la posibilidad de que nextLine()
devuelva el string vacío "" cuando se encuentra inmediatamente un \n.
          ^
__tester__.java:24: error: illegal character: '\'
nextLine(). Devuelve un String que se corresponde con  la línea leida.  Recuerda la posibilidad de que nextLine()
devuelva el string vacío "" cuando se encuentra inmediatamente un \n.

^
__tester__.java:26: error: ';' expected
nextInt(). Devuelve un int que se corresponde con el  token leido.
                          ^
__tester__.java:26: error: ';' expected
nextInt(). Devuelve un int que se corresponde con el  token leido.
                                       ^
__tester__.java:26: error: ';' expected
nextInt(). Devuelve un int que se corresponde con el  token leido.
                                                  ^
__tester__.java:26: error: ';' expected
nextInt(). Devuelve un int que se corresponde con el  token leido.
                                                                  ^
7 errors
```

Incorrecta

Puntos para este envío: 0,00/1,00.

◄ Dungeons & Dragons

Ir a...

Examen 1º ev dic 2022 COPIA PARA PROBAR ►