Primer trabajo práctico
Buscando algunos números "especiales"

(En honor a Srinivasa Ramanujan)

Taller de Álgebra 1 - Segundo cuatrimestre de 2021

¿Escucharon alguna vez hablar de Srinivasa Ramanujan? Ramanujan nació en la India en 1887 y fue una de las grandes mentes de la historia de la matemática. Sus variadas contribuciones fueron extraordinarias e incluyen, entre otras cosas, aportes a la teoría de números, al análisis matemático, a las series y a las fracciones continuas.

¿Saben lo que nos parece más llamativo y a la vez meritorio sobre él? Ramanujan, en sus comienzos, fue completamente autodidacta. No tuvo casi educación formal. Siguiendo su curiosidad incansable por los números, desarrolló sus propias investigaciones en forma totalmente aislada e independiente. Sus logros fueron prontamente percibidos por la comunidad matemática local y su extraordinaria capacidad se hizo conocida también en Europa (epicentro de la matemática en aquel entonces). Así, comenzó su relación con el famoso matemático británico Godfrey Harold Hardy, quien fue, en algún sentido, su mentor.

Durante su corta vida (murió a los 32 años) Ramanujan fue capaz de obtener alrededor de 4000 resultados independientes (mayoritariamente ecuaciones e identidades). Si bien algunos de sus resultados ya eran conocidos, logró otros catalogados como brillantes y sumamente influyentes para la época. En 1917, Ramanujan contrajo tuberculosis, pero su condición mejoró lo suficiente como para regresar a la India en 1919. Murió al año siguiente, desconocido para el mundo en general, pero reconocido por los matemáticos y la comunidad científica como un genio fenomenal.

¿Y por qué les hablamos tanto de Ramanujan? Porque lo involucra en una muy linda anécdota que pone de manifiesto su inmensa capacidad y que sirve de inspiración para este primer Trabajo Práctico. Un día, mientras transitaba su penosa enfermedad en un hospital de Putney (cerca de Londres), Ramanujan recibió la visita de fiel amigo Hardy. Al ver a su colega tan enfermo y no sabiendo de qué hablar, Hardy decidió contarle que el taxi que lo había traído hasta allí tenía en su patente un número poco interesante, tal vez aburrido: el 1729.

Ramanujan convaleciente, pero igualmente manteniendo la asombrosa lucidez que lo caracterizaba le contestó algo así: "No digas eso, no es para nada un número aburrido. El 1729 es muy especial: es el número más pequeño expresable como suma de dos cubos de dos maneras diferentes, ya que $1729 = 1^3 + 12^3$ y también $1729 = 9^3 + 10^3$."

Figura 1: Imagen ficticia del taxi con patente 1729 que tomó Hardy para visitar a su colega y amigo Ramanujan.

Bueno, después de esta historia, estamos en condiciones de comenzar con las consignas del Trabajo Práctico. A la gran mayoría de los mortales, a excepción del gran matemático Ramanujan y tal vez algunos otros pocos genios, le resulta casi imposible determinar en solo una mirada si un número dado es suma de dos cubos (o si tiene alguna otra propiedad que pueda resultar interesante). En este trabajo escribiremos una serie de funciones que pueden ayudarnos en esta tarea. Elaborar la función que sigue es el primer paso.

Ejercicio 1

Escribir la función:

```
esSumaDeDosCubos :: Integer -> Bool
```

que recibe un número natural `n` y devuelve `True` si el número `n` es suma de dos cubos y `False` en caso contrario.

```
*Main> esSumaDeDosCubos 88
False
*Main> esSumaDeDosCubos 2
True
*Main> esSumaDeDosCubos 79625
True
*Main> esSumaDeDosCubos 97187584969377
True
*Main> esSumaDeDosCubos 97187584969378
False
```

La función `esSumadeDosCubos` nos dirá si un número es o no suma de dos cubos. En

el caso de que sí lo sea, es posible que querramos conocer cómo es una posible suma.

Ejercicio 2

Escribir la función:

```
descomposicionCubos :: Integer -> (Integer,Integer)
```

que dado un natural `n` que se escribe como suma de dos cubos devuelve un par (a, b) de números naturales tales que $n = a^3 + b^3$

```
*Main> descomposicionCubos 9
(1,2)
*Main> descomposicionCubos 1729
(1,12)
*Main> descomposicionCubos 4104
(2,16)
*Main> descomposicionCubos 20683
(10,27)
```

Como ya habrán notado, un número dado puede llegar a escribirse como suma de dos cubos de más de una forma. Por ejemplo, el número 20683 se escribe exactamente de dos formas distintas: $20683 = 10^3 + 27^3$ y también $20683 = 19^3 + 24^3$. Aquí adoptamos el mismo criterio que tuvo Ramanujan con el 1729, por lo que no hacemos ninguna distinción entre $10^3 + 27^3$ y $27^3 + 10^3$ ni tampoco entre $19^3 + 24^3$ y $24^3 + 19^3$. Nuestro siguiente objetivo será intentar conocer de *cuántas* formas distintas podemos escribir a un número dado de esta manera particular.

Ejercicio 3

Escribir la función

```
cantidadDeFormas :: Integer -> Integer
```

que recibe un número natural `n` y devuelve la cantidad de formas en la que `n` se escribe como suma de dos cubos.

```
*Main> cantidadDeFormas 88
0
*Main> cantidadDeFormas 4104
2
*Main> cantidadDeFormas 6963472309248
4
*Main> cantidadDeFormas 9
1
```

Obviamente necesitamos una computadora para encontrar números que puedan escribirse como suma de dos cubos de al menos dos formas distintas. Un número para nosotros será *"especial"* si cumple efectivamente esta propiedad.

• Definición: Un número $n \in \mathbb{N}$ es *"especial"* si existen $\{a, b\}, \{c, d\} \subset \mathbb{N}$ dos subconjuntos distintos tales que $a^3 + b^3 = c^3 + d^3 = n$.

La función que aparece a continuación permite conocer el próximo número *"especial"* mayor o igual a uno dado.

Ejercicio 4

Escribir la función

```
especialDesde :: Integer -> Integer
```

que recibe un número natural `n` y devuelve el menor número *"especial"* mayor o igual que `n`.

```
*Main> especialDesde 5
1729
*Main> especialDesde 1729
1729
*Main> especialDesde 1730
4104
*Main> especialDesde 8000
13832
*Main> especialDesde 87539299
87539319
```

Si quisiéramos encontrar todos los números *"especiales"* sin dejar pasar ninguno, o por lo menos todos los que queramos (porque, como ya sospecharán para este momento, hay infinitos), sería buena idea encontrarlos en orden. Sabemos que el primer número *"especial"* es 1729 pero... ¿cuál es el segundo? ¿Y el tercero? ¿Cuál es el quinceavo *"especial"*?

Ejercicio 5

Escribir la función

```
especialNumero :: Integer -> Integer
```

que recibe un número natural `n` y devuelve el n-ésimo número *"especial"*.

```
*Main> especialNumero 1
1729
*Main> especialNumero 4
20683
*Main> especialNumero 12
110808
```

Observemos que es fácil construir números *"especiales"* a partir de otro *"especial"* dado: simplemente multiplicando a este último por un cubo mayor a uno. Por ejemplo, $2^3 \cdot 20683 = 165464$ es *"especial"* ya que admite dos descomposiciones que se obtienen a partir de las del número 20683:

$$2^3(10^3 + 27^3) = 20^3 + 54^3$$
$$2^3(19^3 + 24^3) = 38^3 + 48^3.$$

Diremos que un número es *"muy especial"* si es *"especial"* pero no es producto de un número *"especial"* con un cubo.

• Definición: Un número $n \in \mathbb{N}$ es *"muy especial"* si es *"especial"* y no existen $m, k \in \mathbb{N}$, siendo m *"especial"* y $k > 1$, tales que $n = k^3 \cdot m$.

Pues bien, para finalizar, estamos interesados en poder determinar si un número dado es *"muy especial"*.

Ejercicio 6

Escribir la función

```
esMuyEspecial :: Integer -> Bool
```

que recibe un número natural `n` y devuelve `True` si el número `n` es *"muy especial"* y `False` en caso contrario.

```
*Main> esMuyEspecial 165464
False
*Main> esMuyEspecial 1729
True
*Main> esMuyEspecial 3
False
*Main> esMuyEspecial 20683
True
*Main> esMuyEspecial 805688
True
```

Ayuda

Para resolver el Trabajo Práctico se recomienda utilizar una función `esUnCubo`, que determina si un número dado es o no un cubo. Por cuestión de eficiencia en el cómputo, les proponemos la siguiente implementación para esta función:

```
esUnCubo :: Integer -> Bool
esUnCubo x = (round (fromIntegral x ** (1/3))) ^ 3 == x
```

Conclusión final

Esperamos que después de leer este enunciado, al igual que Hardy, abandonen la errónea idea de que hay números *"interesantes"* y otros *"aburridos"*. Así como todas las personas del mundo tienen algo que las convierte en importantes, peculiares o atractivas, lo mismo ocurre con los números. ¿Saben por qué? Se los vamos a demostrar... Razonemos por el absurdo: si hubiese un conjunto no vacío de números naturales *"aburridos"*, existiría el número más chico *"aburrido"*. ¿Y entonces? Simple, este número *"aburrido"* es, en sí mismo, claramente *"interesante"* por ser el número más pequeño de los *"aburridos"*, lo que resulta en una contradicción, fin de la demostración (y del enunciado).

Condiciones de entrega

El Trabajo Práctico se debe realizar en grupo de 3 alumnos (ni más ni menos). Deberán previamente completar el formulario *"Inscripción del grupo"* que se encuentra en el campus virtual. Aquellas personas que no hayan podido formar grupo deberán completar el formulario *"No tengo grupo"* y se les asignará un equipo de trabajo. La fecha límite para completar estos formularios es el viernes 17 de septiembre a las 23:59 hs. Pasada esta fecha, se considerará que los alumnos que no integran un grupo y no completaron el formulario para informar que no tienen grupo, no van a entregar el Trabajo Práctico en esta instancia y deberán hacerlo en fecha de recuperatorio.

Está terminantemente prohibido subir el código que implementen a repositorios públicos, así como también usar total o parcialmente soluciones implementadas por otros grupos. No está permitida la interacción con otros grupos para discutir las soluciones de los ejercicios propuestos.

No evaluaremos la eficiencia de los algoritmos que propongan para resolver los ejercicios (y por ende, el tiempo en que tardan en ejecutarse las funciones) pero sí la correctitud del código. Algunas funciones que implementen pueden demorar minutos en arrojar el resultado, no se preocupen por el tiempo sino porque devuelvan el valor correcto.

La entrega consiste en un único archivo `.hs` con las funciones de los ejercicios implementadas, junto con todas las funciones auxiliares que sean necesarias para ejecutarlas. Las funciones deben respetar la signatura (nombres y parámetros) especificados en cada ejercicio, dado que serán testeadas automáticamente. El archivo que entregan tiene que poder compilarse solo, y sin llamar a otro módulo. Les pedimos que utilicen como base para escribir el código el archivo tp1.hs que se encuentra en el campus virtual.

El archivo entregado debe tener la forma "`apellido1-apellido2-apellido3.hs`", respetando el orden alfabético de los apellidos. Por ejemplo, si el grupo está formado por los estudiantes María López, Ariel Gómez, y Juan Pérez que cursan en el turno de los viernes debe entregar un archivo llamado "`Gomez-Lopez-Perez.hs`".

La entrega se debe llevar a cabo a través del campus virtual, subiendo el archivo con el código con el mecanismo disponible dentro del espacio del taller en el campus virtual. Un solo integrante será el encargado de subir el Trabajo Práctico al campus en representación de todo el grupo.

Se deberá utilizar exclusivamente los conceptos vistos hasta ahora (semana 5 del cronograma de la materia), inclusive. Se evaluará la corrección de las funciones implementadas, la declaratividad y claridad del código, y que las funciones auxiliares (si las hay)

tengan nombres apropiados.

Si tienen dudas o consultas respecto del trabajo práctico, pueden enviar un mail a la lista de docentes `algebra1-doc (arroba) dc.uba.ar`. No hacer consultas a través del campus virtual ni mandando mails a la lista de alumnos.

Además del código, deberán rendir un coloquio grupal, en el que se conversará **individualmente con cada integrante del grupo** sobre el trabajo realizado, debiendo responder diversas preguntas sobre lo que entregaron. El día y horario del coloquio lo acordarán por email con el docente que haya corregido su trabajo.

Fecha de entrega: hasta el viernes 1 de octubre a las 23:59 hs.