Algoritmos varios con números naturales

$$\frac{1280}{345}=3+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{2+\cfrac{1}{4+\cfrac{1}{2}}}}}$$

Edición 2026

Colección Hojamat.es

PRESENTACIÓN

La utilidad mayor de una hoja de cálculo en el estudio de los números es la de implementación de algoritmos. Con ellos se logra automatismo y velocidad, liberando así nuestro tiempo para tareas más interesantes. En esta publicación se contienen algunos ejemplos, sin pretensiones de un estudio sistemático.

La programación de un algoritmo es una tarea altamente formativa y entretenida, aunque requiere algo de experiencia. En este documento usaremos el Basic para hojas de cálculo como codificación básica, pero podrán aparecer los algoritmos en pseudocódigo u organigramas, así como para otras herramientas.

Para dar más facilidades de comprensión, en algunos algoritmos incluiremos versiones para la calculadora WIRIS, el programa WxMaxima y el lenguaje PARI. En otras ocasiones usaremos nuestra calculadora StCalcu

http://hojamat.es/sindecimales/aritmetica/herramientas/herrarit.htm

En esta edición se han sustituido algunas subrutinas por funciones, para dar mayor generalidad.

Tabla de contenido

ALGORITMOS GENERALES

PERIODO DE LA FRACCIÓN 77/23

Sacar decimales

Las hojas de cálculo están orientadas a los números decimales y se comportan mal en algunos problemas que necesitan operaciones con números enteros. Así, en la cuestión de obtener el periodo de una fracción, aunque es un problema propio de números racionales, los cálculos se efectúan mediante la división entera tradicional. Así se efectuaba en las aulas cuando no existían las calculadoras.

No es difícil implementar una división entera para obtener los periodos largos que se producen con denominadores que contengan como factores números primos grandes. La idea es usar las funciones COCIENTE y RESIDUO

Por ejemplo, para obtener muchas cifras decimales del cociente 77/23, podemos proceder así: El cociente entre ambos sería COCIENTE(77;23) = 3, que sería la parte entera. El resto se hallaría mediante RESIDUO(77;23) = 8.

A continuación, podemos imitar la división que efectuábamos en el colegio (“sacar decimales”). Podemos multiplicar el resto 8 por 10 y volver a repetir la operación: COCIENTE(80;23) = 3, que sería la primera cifra decimal. Volvemos a hallar el resto: RESIDUO(80;23) = 11. Y así reiteramos cuantas veces deseemos.

En la imagen puedes estudiar la forma de ordenar estos cálculos

Núm. cifras	Restos	Cifras del cociente	
0	8	3	Parte entera del cociente
1	11	3	
2	18	4	
3	19	7	
4	6	8	
5	14	2	
6	2	6	
7	20	0	
8	16	8	
9	22	6	
10	13	9	Cifras decimales
11	15	5	
12	12	6	
13	5	5	
14	4	2	
15	17	1	
16	9	7	
17	21	3	
18	3	9	
19	7	1	

Puedes estudiar este algoritmo en el archivo de dirección

http:www.hojamat.es/aritmetica/teoria/hojas/granperiod.ods

Técnicas parecidas he usado para encontrar el periodo y anteperiodo en forma de cadena de caracteres. Se puede consultar en

http://hojamat.es/sindecimales/aritmetica/herramientas/herrarit.htm#periodo

Otras codificaciones

En PARI

Este algoritmo tiene fácil implementación en PARI:

M=77;N=23;T=100;for(i=1,T,Q=M\N;R=M%N;print1(Q);M=R*10)

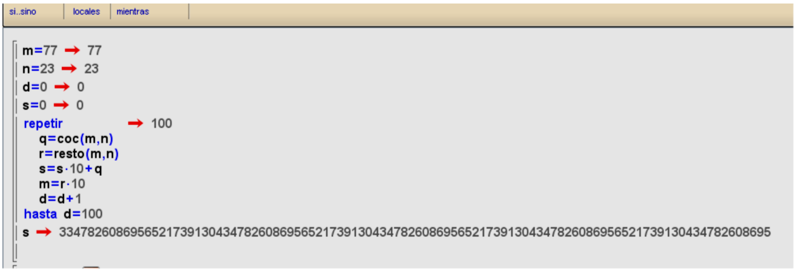
El código ha sido preparado para obtener 100 cifras del cociente 77/23

Su resultado en la página oficial de PARI (https://pari.math.u-bordeaux.fr/gpwasm.html) es:

```
? M=77;N=23;T=100;for(i=1,T,Q=M\N;R=M%N;print1(Q);M=R*10)
3347826086956521739130434782608695652173913043478260869565217391304347826086956521739130434782608695
```

En Wiris

Se pueden cambiar los valores 77, 23 y 100 para adaptarlos a otros datos.



En wxMaxima

```
m:77$
n:23$
s:0$
for i:1 thru 100 do
(
q:floor(m/n),
r:m-q*n,
m:r*10,
s:s*10+q
)$
s;
```

3347826086956521739130434782608695652173913043478260869565217391304347826086956521739130434782608695

También se pueden cambiar los datos 77, 23 y 100

Función en Stcalcu

Admite como datos dos números escritos entre comillas o en una celda con formato de texto, así como el número de decimales escrito con formato numérico:

```
Function stdecimales$(a$, b$, num)
Dim p$, m$, q$
Dim i

p$ = stdivi(a$, b$)
p$ = p$ + "."
m$ = residuo$
For i = 1 To num
m$ = m$ + "0"
Next i
q$ = stdivi(m$, b$)
p$ = p$ + q$
stdecimales = p$
End Function
```

Si lo aplicamos a 77, 23 y 100, nos devuelve una cifra más:

stdecimales("77";"23";100)=3.347826086956521739130
4347826086956521739130434782608695652173913043478260869565217391304347826086956

Es una curiosidad, pero como confeccionador de la calculadora, me satisface.

EL ALGORITMO DE MOESSNER

Se presenta en este apartado una curiosidad matemática a base de cribados: toma la lista de los primeros números naturales. Tacha después uno de cada cuatro, comenzando con el mismo 4:

1 2 3 5 6 7 9 10 11 13 14

Después escribe la lista de sus sumas parciales.

1 3 6 11 17 24 33 43 54 67 81

Y ahora tachas de tres en tres, sumando después de nuevo.

1 3 11 17 33 43 67 81

1 4 15 32 65 108 175 256

Después tachas de dos en dos

1 15 65 175

Y sumas

1 16 81 256

El resultado es la serie de las potencias cuartas de los naturales. Recuerda que hemos comenzando tachando de cuatro en cuatro. ¿Funcionará con el tres?

Lo escribimos sin explicaciones:

1 2 4 5 7 8 10 11 13 14 16 17

1 3 7 12 19 27 37 48 61 75 91 108

1 7 19 37 61 91

1 8 27 64 125 216

Resultan los cubos. Prueba de dos en dos y obtendrás los cuadrados. ¿Funcionará esto siempre? Este algoritmo lo propuso Alfred Moessner y fue demostrada su validez para cualquier valor natural por Oskar Perron en 1951 usando la inducción matemática.

El objetivo de hoy es reproducir este algoritmo con hoja de cálculo, que por cierto no es nada fácil. Contiene una verdadera trampa, que es la posible confusión entre valores y posiciones. Lo vemos:

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	A. Roldán 2011												
2													
3							Algoritmo de Moessner						
4													
5													
6													
7		1	2	3	4	5	6	7	8	9	10	11	12
8		0	0	0	1	1	1	2	2	2	3	3	3
9	4	1	2	3	5	6	7	9	10	11	13	14	15
10		1	3	6	11	17	24	33	43	54	67	81	96
11		0	0	1	1	2	2	3	3	4	4	5	5
12	3	1	3	11	17	33	43	67	81	113	131	171	193
13		1	4	15	32	65	108	175	256	369	500	671	864
14		0	1	2	3	4	5	6	7	8	9	10	11
15	2	1	15	65	175	369	671	1105	1695	2465	3439	4600	5800
16		1	16	81	256	625	1296	2401	4096	6561	10000	14600	20400
17													
18	1												
19													
20													
21													

En la celda A9 escribimos la amplitud de los saltos. En la imagen está preparado para que resulten las cuartas potencias. La hoja se encarga de ir restando una unidad hacia abajo y dejar de escribir cuando se llegue a 1. El modelo está preparado para llegar a 5, pero si lo descargas puedes ampliarlo a tu gusto.

La fila 7 contiene la serie de números naturales. Después se van repitiendo hacia abajo tres filas:

Primera: Es un artificio, pues la hoja debe buscar el elemento a tachar cada vez más lejos, y dependiendo del valor de A9. Esto lo hemos resuelto con la fórmula (usamos la contenida en C8)

=SI(ESNUMERO($A12);SI(RESIDUO(C$7-1;$A12)=0;B8+1;B8);"")

En primer lugar verifica si aún quedan saltos por dar con **ESNUMERO($A12**). Después encuentra el residuo del número de arriba respecto al salto y hace avanzar el contador (B8) si ese número es múltiplo del salto. Así medimos el alejamiento del elemento que debemos tachar. Observa que van aumentando los valores cada tres (representan los tres supervivientes después de tachar)

Segunda: Aquí se eligen los números entre los de arriba, saltando los que ocupan un lugar múltiplo de 3. Después, con la función DESREF se dirigen a la celda adecuada para copiar el número:

=SI(ESNUMERO($A12);DESREF(C9;-2;C8);"")

DESREF se dirige a dos filas más arriba (-2) y salta según indica el valor de arriba (C8). Como esta contiene los saltos adecuados, cada vez que cambie su valor se tacha un número. Es lo que queríamos. No es fácil de entender y cuesta encontrar el procedimiento.

Tercera: Se limita a acumular sumas, y al llegar al nivel 1 produce las potencias deseadas.

Aunque esto no pasa de una curiosidad, la construcción del algoritmo es apasionante. Este no usa macros, y lo puedes descargar en dos versiones desde

hojamat.es/blog/moessner.zip

EL PROBLEMA DE HAMMING

Reciben el nombre de números regulares o 5-lisos aquellos números naturales que son divisibles entre 2,3 y 5 y ningún otro factor primo. Presentan una factorización prima del tipo $2^n3^m5^p$. También puedes identificarlos como aquellos que son divisores de una potencia de 60 (¿por qué?)

Los tienes presentados en estas páginas

http://en.wikipedia.org/wiki/Regular_number

https://oeis.org/A051037

En esta última puedes consultar cuáles son

1, 2, 3, 4, 5, 6, 8, 9, 10, 12, 15, 16, 18, 20, 24, 25, 27, 30, 32, 36, 40, 45, 48, …

Si se les añade el número 1 como primer elemento, forman la llamada sucesión de Hamming.

El mayor interés que presentan estos números es el estudio de la formación ordenada de la sucesión, formándola **a partir de los elementos ya descubiertos**. Esto último es importante, pues si no, bastaría con ir recorriendo los números naturales para quedarnos sólo con los del tipo $2^n3^m5^p$.

Los posibles algoritmos, como el de Dijkstra, se estudian frecuentemente en programación funcional, como puedes ver en esta página de José_A_Alonso.

Como siempre en mi blog, optaré por un enfoque elemental, didáctico y ¡cómo no!, usando una hoja de cálculo.

Generación del siguiente número de Hamming

Una vez que tienes escritos los primeros números de la sucesión 1,2,3,4,5,6,8, … (veremos que en realidad sólo hay que escribir 1), para obtener el siguiente bastará multiplicar uno de ellos por 2,3 o 5, pero, ¿cuál? En la sucesión 1,2,3,4,5,6,8, … no me sirve multiplicar por 2 el número 3, porque me daría 6 que ya lo tengo. Sí me convendría multiplicar por 2 el 5, con lo que obtendría el 10, o al revés, multiplicar 5 por 2. Esto no tiene nada de sistemático, por lo que deberemos ordenarlo un poco:

Dada una sucesión de Hamming con varios elementos, para formar el siguiente nos basaremos en estos criterios:

(1) Multiplicamos por 2, 3 o 5 todos los elementos que ya tenemos, y nos quedamos con el resultado menor que aún no esté incorporado a la sucesión. En el caso del ejemplo, el 9.

(2) Para guiarnos en este proceso, escribimos todos los números del tipo 2H, 3H y 5H, (representando H los números de Hamming que ya tenemos) en tres columnas.

H	2H	3H	5H
1	2	3	5
2	4	6	10
3	6	9	15
4	8	12	20
5	10	15	25
6	12	18	30
8	16	24	40

(3) En cada columna señalamos aquel número que cumple que todos los de arriba no superan el número que ya tenemos (8) y él es el primero que sí lo sobrepasaría.

En la siguiente tabla los tienes señalados en este caso.

H	2H	3H	5H
1	2	3	5
2	4	6	**10**
3	6	**9**	15

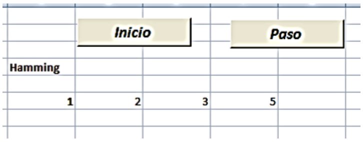
4	8	12	20
5	**10**	15	25
6	12	18	30
8	16	24	40

(4) Por último, de los tres candidatos elegimos el menor, 9, y ese será el siguiente elemento de la sucesión de Hamming.

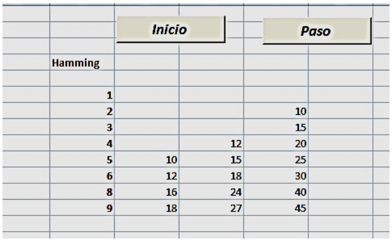
Reiteramos estas operaciones y los obtendremos todos de forma ordenada. Este procedimiento tiene la ventaja de que una vez elegido un número de la columna quedan desechados los anteriores, por lo que es posible mantener unos punteros que nos indiquen por dónde vamos.

El algoritmo con hoja de cálculo

Podemos traducirlo a hoja de cálculo. Lo hemos intentado sin usar macros, pero aparecían referencias circulares muy molestas, por lo que hemos acudido al uso de rutinas y botones. Se inicia el proceso con el botón Inicio, que escribe el primer término 1 y sus tres múltiplos 2,3 y 5.

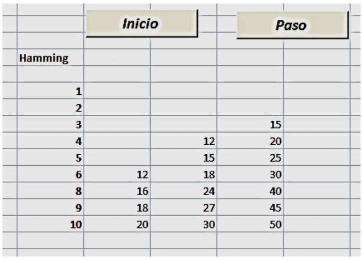


Después, cada vez que pulsemos sobre el botón Paso se irán eligiendo los múltiplos adecuados desechando los anteriores y los iguales. En la imagen puedes ver el estado del proceso después de obtener el 9:



Hamming			
1			
2			10
3			15
4		12	20
5	10	15	25
6	12	18	30
8	16	24	40
9	18	27	45

Se han dejado en blanco los múltiplos usados. La hoja elige después el mínimo (sería 10), elimina sus iguales, lo incorpora a la lista, crea sus múltiplos y borra los innecesarios.



Hamming			
1			
2			
3			15
4		12	20
5		15	25
6	12	18	30
8	16	24	40
9	18	27	45
10	20	30	50

El cómo lo consigue lo podrás estudiar descargando la hoja en Excel desde

http://hojamat.es/blog/hamming.xlsm

Estudio mediante funciones

Para ver si un número es regular o 5-liso bastaría con esta definición de función:

```
Public Function es_regular(n) As Boolean
Dim nn

nn = n
While nn = 2 * Int(nn / 2): nn = nn / 2: Wend
While nn = 3 * Int(nn / 3): nn = nn / 3: Wend
While nn = 5 * Int(nn / 5): nn = nn / 5: Wend
If nn = 1 Then es_regular = True Else es_regular = False
End Function
```

Observa cómo lo detecta: mientras el número sea par, lo va dividiendo entre 2, con lo que al final deja de serlo. Mientras sea múltiplo de 3 y de 5 también va dividiendo. Si el número es regular se agotarán todos los factores y quedará sólo un 1 y el valor de la función será VERDADERO. Si no es regular es porque o no se puede dividir entre 2,3 o 5, o al final del proceso queda un factor mayor que 1, y la función devuelve FALSO.

Con esta función puedes iniciar la sucesión de Hamming en el punto que desees. Basta ir recorriendo números y eligiendo los que sean regulares. También es muy sencillo usar la función ***proximo_regular***:

```
Public Function proximo_regular(n)
Dim p

p = n + 1
While Not es_regular(p): p = p + 1: Wend
proximo_regular = p
End Function
```

Con esta función puedes descubrir, por ejemplo, que el primer regular de siete cifras es 1012500.

ALGORITMO 196

En el blog “Espejo lúdico”, con fecha 9 de diciembre de 2008, se ha publicado esta propuesta:

Si a un número se le suma su reverso (por ejemplo 75 + 57) y se hace lo mismo con el resultado, llega un momento en que el resultado es capicúa.

Por ejemplo

75 + 57 =132; 132 +231 = 363

Se llama así el algoritmo porque para el número 196, en el momento de escribir este texto, aún no se sabe si el proceso para en un capicúa concreto o se prolonga indefinidamente. Se han llegado a organizar búsquedas que han durado años.

El 196 es el primero de los llamados números de Lychrel, de los que aún no se sabe si producen una parada en el algoritmo. Los siguientes son 295, 394, 493, 592, … (http://oeis.org/A023108). Todos están cerca de un número terminado en cero.

Para un cierto número de dos cifras es necesario repetir este proceso más de 10 veces. ¿Cuál es ese número?

El algoritmo propuesto tiene fácil ejecución para quien sepa sumar, pero no es tan simple para una hoja de cálculo. Hay que tener en cuenta que los números se almacenan en formato binario y la hoja "no sabe" la cifras que tiene un número en el sistema de numeración decimal. Por tanto, si no definimos nuevas funciones, no podrá invertir las cifras de un número ni tampoco averiguar si es capicúa o no. Así que necesitamos:

Función INVERTIR_CIFRAS: Debe de actuar sobre un número e invertir el orden de todas sus cifras, devolviéndonos el resultado.

Public Function cifrainver(n)
Dim l, i
Dim c
Dim auxi$, auxi2$, ci$
 ' invierte el orden de las cifras para dar otro número

```
auxi = Right(nn$, Len(nn$) - 1)
auxi2$ = ""
l = Len(auxi)
For i = l To 1 Step -1
ci$ = Mid(auxi, i, 1)
auxi2 = auxi2 + ci$
Next i
c = Val(auxi2$)
cifrainver = c
End Function
```

Función ESCAPICUA: Debe averiguar si un número es capicúa o no y devolver VERDADERO o FALSO

```
Public Function escapicua(n) As Boolean
Dim l, i, k
Dim c As Boolean
Dim auxi$
auxi = Right(nn$, Len(nn$) - 1)
l = Len(auxi)
If l < 2 Then
escapicua = False
Else
c = True
i = 1
k = Int(l / 2)
While i <= k And c
```

```
  If Mid(auxi, i, 1) <> Mid(auxi, l - i + 1, 1) Then c = False
  i = i + 1
  Wend
End If
escapicua = c
End Function
```

Invito a usar el algoritmo para averiguar (para números no muy grandes) cuántos pasos son necesarios hasta que un número desemboque en un capicúa sumando de forma reiterada su reverso.

Jugando un poco con este algoritmo se pueden descubrir hechos interesantes.

Llamaré meta al capicúa en el que termina un algoritmo aplicado a un número (semilla) y ruta al conjunto de números que se recorren hasta llegar desde la semilla hasta la meta.

Metas capicúas de dos cifras: Es evidente que los números semilla que desembocan en el mismo capicúa tienen todos la misma suma de cifras y esta es menor que 10. Por ejemplo, 70, 61, 52, 43, 34, 25, 16, 7, desembocan en 77=11*(a+b) con a+b<10

Llegan a 121 los de dos cifras que sumen 10 u 11 y alguno más de tres cifras ¿cuál? Y a 363 los de suma 12 y alguno más de tres cifras, con expresión N=11*(a+b) en el que se cumple a+b>10=11(10m+n) = 110m+11n

Metas de tres cifras: Se puede demostrar que sólo son metas los capicúas en los que la cifra del centro es par, como 343, 929, 787, … y por tanto no lo son 232, 878 o 171. Intenta demostrarlo, que no es complicado.

Metas de cuatro cifras: Han de ser múltiplos de 11. ¿Por qué?

Números ilustres: Los números 495 y 1089, están ambos en la misma ruta que desemboca en el 79497. Además, tienen como meta el 1089 los múltiplos de 198 de tres cifras.

Otra curiosidad: Los 10 primeros múltiplos de 1089 llegan todos hasta el 79497.

Si no recuerdas el porqué de que les llame “ilustres” al 495 y al 1089, consulta esta dirección:

http://www.hojamat.es/sindecimales/aritmetica/propuestas/proparit.htm

Ahí te enterarás que 495 es la constante de Kaprekar para tres cifras. Si elegimos como semilla la constante para cuatro cifras 6174, también tiene como meta 79497, lo que nos confirma que ambos algoritmos están relacionados. Es curioso que 6174+4716 = 10890 = 1089+9801.

Ver solución

El número 98 necesita 24 iteraciones para desembocar en capicúa.

El número 110 desemboca en el 121

Es fácil razonar que la segunda cifra de un capicúa de tres cifras que sea meta ha de ser par.

En efecto, si proviene de un número de dos cifras ha de ser múltiplo de 11, y los únicos capicúas son 121, 242, 363 y 484, todos con las decenas pares.

Si proviene de un número de tres cifras 100a+10b+c, tendrá la forma 101(a+c)+20b, con a+c<=9 y b<6, con lo que sus cifras serán a+c, 2b, a+c, y por tanto la cifra de las decenas será par.

Las metas de cuatro cifras han de ser múltiplos de 11.
Es evidente por ser capicúas
N=1000a+100b+100b+a=1001a+110b

CONJUNTOS IDÉNTICOS

En algunas cuestiones resulta útil decidir de forma automática si dos conjuntos son idénticos o no. Por ejemplo, en las tablas de multiplicar de los cuerpos finitos, como Z/Z7, es interesante descubrir si

(a) No existen elementos repetidos en ninguna fila o columna

(b) Los elementos de las distintas filas son los mismos.

Si escribimos los dos conjuntos en una hoja de cálculo, en filas paralelas, deberemos comprobar cuatro hechos para decidir si los conjuntos son idénticos o no:

1. No existen elementos repetidos en el primer conjunto
2. Tampoco se repiten los del segundo
3. Todo elemento del primero ha de pertenecer al segundo
4. Todo elemento del segundo ha de pertenecer al primero.

Las cuatro cuestiones las resuelve la función CONTAR.SI. Recorremos todo el primer conjunto y mediante esta función contamos las veces que figuran en el segundo. Si esos valores son mayores que 1, es que existen repetidos en el segundo conjunto, y si es 0, es que falta alguno. Lo deseable, pues, es que todos los contadores presenten el valor 1.

Procedemos de la misma forma, contando las veces que los elementos del segundo conjunto figuran en el primero, y también han de valer 1. Para evitar problemas en las siguientes operaciones que explicaremos, a las celdas vacías también se le debe asignar un 1.

¿Cómo resumimos la situación? Multiplicamos todos los contadores del primer conjunto, y nos ha de resultar la unidad. Ocurrirá lo mismo con el producto de los del segundo, por lo que si multiplicamos ambos productos, obtendremos un criterio para decidir si los dos conjuntos son idénticos: el que el producto final tenga el valor de 1.

ALGORITMOS AYUDADOS

En un comentario publicado en mi blog,

Claudio (http://simplementenumeros.blogspot.com/) proponía lo siguiente:

Te envío un problema que mandó Rodolfo Kurchan a la lista de Snark:

Este simpático acertijo me lo envió Michael Reid de EEUU: Colocar los dígitos 0, 1, ..., 9, sin repetir en la expresión a^b + c^d + e^f + g^h + i^j para obtener el año actual.

Pensé en aplicar esta idea al 2010. Como estábamos en días navideños, no me apetecía pensarlo mucho. Por otra parte, si intentaba un algoritmo sin preparación, me podía encontrar con 10^10 pasos, lo que era demasiado para cualquier hoja de cálculo.

Así que pensé en ayudar un poco al algoritmo, para ver si entre la máquina y yo lo encontrábamos sin gran esfuerzo. La primera idea fue la de un algoritmo voraz, pero también había que diseñar bastante, y mi cabeza estaba con los villancicos.

Después de cavilar se me ocurrió elaborar una lista de potencias desde 0^0 hasta 9^9 eliminando las mayores de 2010 y casi todas las triviales: 1^3, 1^4, ...0^8…La lista estaba compuesta por 1296,1024, 729, 625, 512, … 5, 4, 3, 2, 1, 0 (29 en total)

De esta forma, el algoritmo podía traducirse a "Descomponer 2010 en sumandos tomados de una lista". Aún así, los cálculos tardaban demasiado (29^5 pasos) y los compromisos sociales me esperaban. Lo dejé para otro día.

Reanudada la tarea, impuse la condición de que los sumandos fueran no crecientes, lo que simplificaba la búsqueda, y como 2010/5 = 402 debería haber algún sumando superior a esa cantidad. De esta forma puede seguir recortando pasos.

Al final obtuve una lista de sumandos que comenzaba con

1296	625	81	8	0
1296	625	81	7	1
1296	625	81	6	2

y terminaba con

625	625	625	128	7
625	625	512	243	5
625	625	512	216	32

Ya tenía algo con lo que trabajar. Añadí a cada sumando los dígitos que lo formaban como una potencia:

1296	6 y 4	625	5 y 4	81	9 y 2 3 y 4	8	8 y 1 2 y 3	0
1296	6 y 4	625	5 y 4	81	9 y 2 3 y 4	7	7 y 1	1
1296	6 y 4	625	5 y 4	81	9 y 2 3 y 4	6	6 y 1	2

Ya sólo quedaba elegir las posibilidades en las que no se repitieran los dígitos. Encontré cuatro:

4^5+2^8+9^3+1^7+0^6=2010

4^5+2^8+9^3+1^6+0^7=2010

4^5+2^8+3^6+1^8+0^7=2010

4^5+2^8+3^6+1^7+0^9=2010

De esta forma la hoja de cálculo aportó una base para soslayar mi pereza mental, y yo le ayudé con mi sentido común.

SUMA DE TRES NÚMEROS TRIANGULARES

En 1796, Gauss descubrió que todo entero positivo puede representarse como la suma de tres números triangulares (o menos). Estos pueden ser iguales, y si consideráramos el 0 como triangular, podríamos afirmar que todo número natural es suma exactamente de tres triangulares.

Un ejercicio interesante es el de descubrir un algoritmo que los encuentre. Para hacerlo más formativo podemos basarnos en dos funciones, una para descubrir si un número es triangular y otra que nos devuelve el mayor triangular mayor o igual que un número dado.

Función estriangular

Si un número N es triangular verificará igualdad N=x(x+1)/2, con x y N ambos enteros, lo que obliga a que 8*N+1 sea cuadrado perfecto ¿por qué? Esto nos lleva a este código:

Public function estriangular(n) as boolean
dim a

a = Int(sqr(8*n+1))

if a*a=8*n+1 then estriangular = true else estriangular = false

end function

Para encontrar el mayor triangular contenido en un número N bastará resolver la ecuación N=x(x+1)/2 truncando el resultado a un número entero. Así que quedará:

Public Function mayortriang(n)
dim a
a = Int((sqr(8*n+1)-1)/2)
mayortriang=a*(a+1)/2
end function

Con esto ya tenemos preparado un algoritmo para hoja de cálculo que encuentre todas las descomposiciones en tres triangulares (incluido el cero):

Function sumatriangulares(i)
Dim j, k
Dim a, b
Dim s$
s = ""
a = mayortriang(i)
For j = 0 To a '(se recorren los valores posibles del primer sumando)

If estriangular(j) Then
b = mayortriang(i - j)
For k = j To b '(se recorren los valores posibles del segundo sumando)
If estriangular(k) Then
If estriangular(i - j - k) And i - j - k >= k Then '(el tercer sumando ha de ser triangular y el trío creciente)
s = s + " # " + Str$(j) + ", " + Str$(k) + ", " + Str$(i - j - k)
End If
End If
Next k
End If
Next j
sumatriangulares = s
End Function

Pues ánimo y a implementarlo. Puedes añadir una variable que cuente todas las formas de descomposición que tiene un número. Por ejemplo, entre los de tres cifras hay uno que admite 24 sumas distintas de triangulares ¿Cuál?

Otras codificaciones

En PARI

isinteger(n)=(n==truncate(n))

isquare(n)= { local(f,m,p=0); if(n==1,p=1,f=factor(n); m=gcd(f[, 2]); if(isinteger(m/2),p=1));return(p) }

istriang(n)=isquare(8*n+1)

{n=95;for(i=0,n,if(istriang(i),for(j=i,n,if(istriang(j),for(k=j,n,if(istriang(k)&&i+j+k==n,print(i," ",j," ",k)))))))}

En primer lugar, se definen ***isinteger***, para ver si un número es entero, ***isquare***, para cuadrado e ***istriang*** para triangular.

Después se recorren ciclos para encontrar tres sumandos triangulares crecientes en sentido amplio. Si se cambia el valor 95 por otro número, dispondremos de un algoritmo que resuelve el problema para ese número.

En WxMAXIMA

m:92$
esent(x):=if x=floor(x) then true else false;
escuad(x):=(y:sqrt(x),if esent(y) then true else false);
estriang(x):=if escuad(8*x+1) then true else false;

```
for i:0 thru m do
(
if estriang(i) then for j:i thru m do
      (if estriang(j) then for k:j thru m do
         (if estriang(k) and i+j+k=m then display(i,j,k)
          )
      )
)$
```

Como en el anterior, se definen las funciones previamente y después se construyen tres bucles. El número se escribe en la primera línea.

Solución

La solución es el número 952, que admite estas 24 descomposiciones en triangulares:

0	6	946
0	91	861
1	171	780
1	210	741
3	3	946
6	351	595
10	276	666
21	28	903
21	190	741

21	435	496
36	55	861
36	136	780
55	231	666
66	66	820
66	325	561
78	171	703
78	378	496
91	120	741
91	231	630
91	300	561
105	351	496
136	351	465
171	253	528
276	325	351

FACTORIZACIÓN DE FERMAT

Algoritmo a paso de tortuga

La factorización de Fermat siempre se ha presentado como una técnica para representar un número impar como producto de dos de sus factores **sin usar la lista de números primos**. No es el único algoritmo de factorización con esa propiedad. Si extraemos progresivamente el factor más pequeño (mayor que 1) de N aseguraremos que hemos encontrado un número primo sin tener que memorizar la lista de primos

En efecto, la factorización de Fermat no se basa en los factores primos, sino en representar un número impar N como una diferencia de dos cuadrados y después expresar la misma como el producto de una suma por una diferencia, con lo que se logra la factorización:

$N=y^2-x^2=(x+y)(y-x), y>x$

En el caso impar esta operación siempre es posible, porque $N=(N+1)^2/4-(N-1)^2/4$, que da lugar a la factorización N=N.1

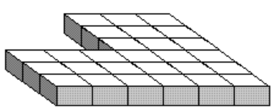
Desde el punto de vista algorítmico, la búsqueda de los valores de x e y adecuados presenta también otra originalidad, y es que se puede organizar de forma bastante eficiente sólo con sumas y comparaciones. En realidad, todos los algoritmos se pueden organizar así, y en caso último mediante la máquina de Turing, pero tampoco hay que llegar tan lejos, y con tantas operaciones se lentifica mucho el proceso. En el caso de la factorización de Fermat se logran resultados bastante aceptables. Es una tortuga, pero algo veloz.

La creación del algoritmo se basa en estos hechos que te pedimos intentes justificar:

(1) El valor de **y**, y por tanto el de **x**, están acotados, y su cota no está excesivamente alejada de n ¿Cuál es y cómo se demuestra?

(2) Para demostrar que un número es cuadrado perfecto, no es necesario dividir ni extraer la raíz cuadrada ¿Cómo se hace?

(3) En realidad, la factorización de Fermat consiste en darle al número impar la figura de escuadra simétrica (gnomon) de varias formas posibles. Observando la imagen puedes resolver las dos primeras cuestiones.



(4) Podemos encontrar la raíz cuadrada entera de un número haciendo crecer de forma simultánea dos

sucesiones, una linealmente y otra de forma cuadrática. Esto parece obvio, pero ¿cómo se organiza?

Podemos representar el algoritmo como una carrera, liebre y tortuga, entre $a=x^2$ y $b=y^2$, en la que la tortuga sale con ventaja porque se suma al número N, ya ha de ser $y^2=x^2+N$. Todo esto suena a metáfora, pero funciona.

Daremos a continuación un desarrollo en el que se ocultarán algunos detalles para hacer pensar un poco a quien lo siga.

Obtención de un primer valor de $a=y^2$

Tomamos tres variables, y, a, i

Las iniciamos a

y=1; a=1; i=1

Mientras a no sobrepase a N hacemos crecer estas variables así:

y=y+1; i=i+2; a=a+i

con lo que lograremos el primer cuadrado que es mayor o igual que N

Sólo hemos usado sumas y una comparación. Razona por qué se da este resultado.

Obtención de valores adecuados de $a=y^2$ y de $b=x^2$

Una vez obtenido y^2 iniciamos el crecimiento de la misma forma para x^2

$x=1; b=1; j=1$

Mientras $N+b$ no sobrepase a y^2, hacemos crecer las variables:

$x=x+1; j=j+2; b=b+j$

para formar el cuadrado x^2

Si ocurre que $N+b=a$ hemos conseguido una factorización, pues $N=y^2-x^2$

Obtenemos todas las coincidencias

Ya sólo queda hacer crecer **a** y **b** de la misma forma y comprobar si se da la igualdad N+b=a en más ocasiones, y así hasta la cota.

No he querido usar el lenguaje algorítmico, y se han ocultado algunos detalles, como qué hacer si N es un cuadrado perfecto. Lo importante de lo explicado es que sólo he sumado y comparado. No se ha recurrido a multiplicar, dividir o extraer la raíz cuadrada.

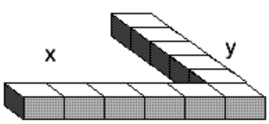
Puedes estudiar más a fondo el algoritmo en el Anexo. Si recorres el código de la macro usada, sólo verás operaciones de sumar. El proceso no es un prodigio de velocidad, pero esto es un divertimiento. La idea de hoy no era correr, sino demostrar una posibilidad. Tampoco corrió Fermat y hay que ver lo que logró.

Factorización de Fermat SOLUCION

(1) La variable **y** ha de estar acotada, porque tanto **x+y** como **y-x** son menores que N (al menos uno de ellos), luego su suma cumplirá **2y<2N** y por tanto N es una cota para y.

(2) Para saber si un número es un cuadrado sin usar la raíz cuadrada basta ir sumando los números impares consecutivos. Si una de las sumas coincide con ese número, será cuadrado perfecto. Si las sumas lo sobrepasan sin igualarse a él, no lo será.

(3) Si representamos N como un gnomon cuyo lado mayor es **y** mientras el lado de su hueco es **x**, vemos que en el caso extremo en el que **y-x** valga 1, **y** ha de ser menor o igual que (N+1)/2, lo que nos da una mejor cota para **y**. Lo podemos ver en la imagen, que corresponde a N=11.



Complemento

Factorización de Fermat en VBASIC

Entrada: Un número entero n

Operación: Descompone n en dos factores según la descomposición factorial sugerida por Fermat

Código en Basic

Sub factorfermat(n)

dim y,a,m,x,j,s,b,p as long

'Se hace crecer la variable "a" hasta que sobrepase a "n"

a=1:y=1:i=1
while a<n
i=i+2:a=a+i:y=y+1'la variable es suma de impares consecutivos y por tanto cuadrado perfecto
wend
' Aquí se resuelve el caso en el "n" sea cuadrado perfecto
if a=n then
msgbox(y)
msgbox(y)
i=i+2:a=a+i:y=y+1
end if
' Se hacen crecer "x" e "y"

```
x=1:j=1:b=1
while y+y<=n+1
while n+b<a
j=j+2
b=b+j
x=x+1
wend
'Se ha descubierto una coincidencia
if a=n+b then
m=x+y
p=y-x
msgbox(m)
msgbox(p)
end if
i=i+2:a=a+i:y=y+1
wend
End Sub
```

REPRODUCIR RESULTADOS

Somos muchos en el mundo. Estudiamos en una Facultad de Matemáticas, llevamos años y años enseñándolas, seguimos estudiando distintos temas y leyendo libros de divulgación, entretenimientos o curiosidades, pero nunca hemos publicado un resultado matemático apreciable. Sólo nos queda disfrutar con desarrollos ajenos, resolver placenteramente problemas de más o menos dificultad y… reproducir resultados.

Lo de obtener lo que ya han descubierto otros puede ser formativo y entretenido si lo intentamos con herramientas distintas a las del primero que lo logró. En mi blog uso las hojas de cálculo, lo que nos exige la construcción de tablas en las que se llevan al límite las posibilidades de las funciones que traen implementadas, o bien, que es una tarea más apasionante, implementar algoritmos adecuados mediante macros.

Propongo una reproducción:

He leído por ahí, en la Wikipedia, Wolfram Mathworld o una página similar, que el número 2011 es estrictamente no palindrómico. Se llaman así los números N que no son palindrómicos (capicúas) para bases comprendidas entre 2 y N-2. No se consideran bases mayores porque todos los números se expresan en ellas como capicúas (se admite que lo son los de una cifra) para bases mayores que N-2. ¿Sabrías razonarlo?

Alguien se ha tomado la molestia de ir probando el 2011, imagino que de forma automática, para todas las bases comprendidas entre 2 y 2010.

¿Puedes reproducir ese resultado con hoja de cálculo?

Para averiguar si un número es estrictamente no palindrómico necesitaremos una función que nos diga si es palindrómico o no en una base dada, y después recorrer todas las bases entre 2 y N-2 para descubrir si hay o no resultados negativos.

Diseñaremos la función ESCAPICUA(n,b), donde n será el número a probar y b la base del sistema de numeración. Esta función nos devolverá un 1 si el número es palindrómico y 0 si no lo es. Usamos 1 y 0 porque son más cómodos que True y False.

Necesitaremos organizar dos fases de cálculo

a) Extracción de las cifras de n en base b y almacenamiento de las mismas en una matriz c

b) Emparejamiento de las cifras de forma simétrica para averiguar si son todas iguales por parejas (caso palindrómico) o bien existe una que no es igual a su simétrica.

Primera fase: extracción de las cifras

Usaremos un algoritmo voraz, en el que **n** va disminuyendo de valor, con lo que la velocidad se acelera. Dividimos en cada paso n entre b, quedando el cociente como nuevo valor de n y el resto como cifra nueva. Cuando el cociente sea cero, paramos.

Puedes estudiarlo en Basic

En el listado he copiado n en m para preservar su valor

' extraer cifras
nopara=true Esta variable determina si se para o no el proceso
nc=0 Contador de cifras
while nopara
q=int(m/b):r=m-q*b Se halla el cociente y el resto de m entre la base
if q=0 then nopara=false Si el cociente es cero, se para
nc=nc+1:c(nc)=r:m=q Se incrementa el contador de cifras y se almacena la nueva
wend

Segunda fase: Comparación entre cifras

Una vez almacenadas las cifras, si sólo hay una, se declara el número como palindrómico. En caso contrario, si se detecta una desigualdad entre cifras simétricas, se declara como no palindrómico.

En Basic

esca=1 Admitimos que es capicúa
if nc>1 then Si hay más de una cifra, analizamos
for q=1 to int(nc/2)

if c(q)<>c(nc-q+1) then esca=0 En caso de desigualdad, no es capicúa

next q

escapicua=esca

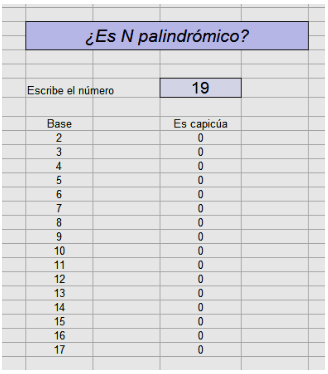
end if

Si deseas implementar esta función en tu hoja de cálculo, copia el código completo:

```
Function escapicua(n,b)
dim c(50)
dim m,q,r,nc,esca
dim nopara as boolean
m=n
' extraer cifras
nopara=true
nc=0
while nopara
q=int(m/b):r=m-q*b
if q=0 then nopara=false
nc=nc+1:c(nc)=r:m=q
wend
esca=1
if nc>1 then
for q=1 to int(nc/2)
if c(q)<>c(nc-q+1) then esca=0
```

next q
escapicua=esca
end if
end function

Con esta función se puede rellenar una columna que actúe sobre las bases comprendidas entre 2 y N-2. Por ejemplo, en la imagen puedes comprobar que el número 19 es estrictamente no palindrómico:



¿Es N palindrómico?

Escribe el número: 19

Base	Es capicúa
2	0
3	0
4	0
5	0
6	0
7	0
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0
16	0
17	0

Los primeros números estrictamente no palindrómicos son:

1, 2, 3, 4, 6, 11, 19, 47, 53, 79, 103… (http://oeis.org/A016038) En esta página de OEIS descubrirás que **son todos primos a partir del 6**. La justificación de esto proviene de que a*b=a(b-1)+a siendo a<b, lo que lo convierte en palindrómico en base b-1 (se expresa como 11). En el caso del cuadrado a*a=(a-1)^2+2(a-1)+1, lo que lo convierte en palindrómico en base a-1, expresado como 121. Luego los compuestos serán paindrómicos en ciertas bases. Puedes leer más detalles en la dirección citada.

He aplicado la prueba a 2011 y, efectivamente, no es palindrómico para ninguna base comprendida entre 2 y 2010.

Con ello he reproducido un resultado, con la consiguiente diversión e incremento de nuestra confianza en la comunidad matemática.

Si te atreves, codifica una función ESTRICTCAP, que decida si un número es estrictamente no palindrómico. Bastará programar en Basic lo que en la imagen he efectuado con columnas.

APRENDER COMPROBANDO

Tanto Internet como los libros de divulgación matemática están llenos de listas de números que se caracterizan por ser los únicos que cumplen algún requisito.

La página http://oeis.org/A084687 nos presenta la siguiente lista como la de los números enteros positivos que son múltiplos de los números formados por sus mismas cifras ordenadas en orden creciente:

9513, 81816, 93513, 94143, 95193, 816816, 888216, 933513, 934143, 935193, 941493, 951993, 2491578, 8166816, 8868216, 9333513, 9334143, 9335193, 9341493, 9351993, 9414993, 9519993, 24915798, 49827156, 81666816, 87127446, 88668216, 93333513

Este requisito ha de cumplirse en sentido estricto:

- No pueden contener cifras nulas.
- No pueden poseer ellos mismos las cifras ya ordenadas.

El primer ejemplo de la lista es el número 9513, que no contiene cifras nulas y es múltiplo de 1359, formado por las cifras 9, 5, 1 y 3 ordenadas de forma creciente.

Los cocientes que se forman son "casi todos" iguales a 7. Investiga este hecho si quieres.

Un ejercicio muy formativo es el de obtener esa misma lista con nuestros propios instrumentos, que aquí será la hoja de cálculo. Para ello debemos organizar muy bien el proceso, y en esta tarea aprenderemos de Matemáticas y de programación mucho más de lo que nos creemos.

A continuación, veremos una organización del proceso de obtención de la lista presentada, aunque sería deseable que nuestros lectores no siguieran leyendo y pasaran a su propia organización. Así también ellos, como nosotros, aprenderían probando.

Un posible esquema sería el siguiente:

Obtención de la lista de números

- *Se recorren todos los números A desde un inicio hasta un número final.*
- *Para cada uno se realizan estas operaciones:*
- *Calcular el número de cifras de A*
- *Extraer todas las cifras de A. Si alguna es cero se rechaza el número.*
- *Ordenar las cifras*
- *Formar con esas cifras un nuevo número B*
- *Si A=B se rechaza el número.*
- *Si A es múltiplo de B se incorpora A a la lista.*
- *Se pasa al siguiente número*

Si te interesa la programación en Basic, puedes estudiar el siguiente código comentado para OpenOffice.org Calc y Excel:

Funciones auxiliares

Para saber si m es múltiplo de n. Devuelve 1 si lo es, y 0 si no lo es

```
function esmultiplo(m,n)
if m=int(m/n)*n then esmultiplo=1 else esmultiplo=0
end function
```

Para contar el número de cifras

```
function numcifras(n)
numcifras=int(log(n)/log(10))+1
end function
```

Extrae la cifra de orden n de un número m

```
function cifra(m,n)
dim a,b
a=10^(n-1)
b=int(m/a)-10*int(m/a/10)
cifra=b
end function
```

Función de búsqueda

En la edición de 2026 se ha convertido una subrutina en función, para que puede usarse en varias hojas de cálculo.

```
Function busqueda_especial$(i)
Dim j, k, l, a, b, p, q
Dim s$
Dim ci(12)
```

'Extrae cifras y las ordena

```
s = ""
k = numcifras(i)
```

```
For l = 1 To k
ci(l) = cifra(i, l): If ci(l) = 0 Then Exit Function 'no se admiten cifras nulas
Next l
```

'Las ordena

```
If k >= 1 Then
For j = 1 To k - 1
For p = 2 To k
If ci(p - 1) < ci(p) Then b = ci(p - 1): ci(p - 1) = ci(p): ci(p) = b
Next p
Next j
End If
```

'Construye el número con cifras ordenadas

```
q = 0
For j = 1 To k
q = q + ci(j) * potencia(10, j - 1)
Next j
```

'Si es múltiplo, lo añade a la solución

```
If esmultiplo(i, q) And i <> q Then
s = s + "&& " + Str$(i) + " /" + Str$(q) + "=" + Str$(i / q)
```

End If
If s = "" Then s = "NO"
busqueda_especial = s
End Function

Por ejemplo, búsqueda_especial(93513)=&& 93513 / 13359= 7. Si no cumple lo exigido, devuelve un “NO”.

Ánimo y a estudiarla, que contiene bastante información valiosa.

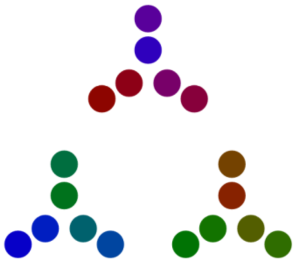
UNA HUMILDE IMITACIÓN

Desde hace unas semanas circula por la red un atractivo vídeo en el que se visualiza la factorización de los números como conjuntos de puntos en forma de árboles poligonales circulares muy cercanos a los conjuntos fractales

http://miscuriosidadesmatematicas.blogspot.com.es/2012/11/diagrama-animado-sobre-factorizacion-de.html

http://www.datapointed.net/visualizations/math/factorization/animated-diagrams/

http://mathlesstraveled.com/2012/10/05/factorization-diagrams/



En la imagen se puede ver el desarrollo para el número 18, en el que los niveles están definidos por los factores 3, 3 y 2.

Como en mi blog no se sale de los números y de la hoja de cálculo nos planteamos un reto:

¿Hasta dónde podíamos imitar estos esquemas usando tan sólo la programación de una hoja de cálculo?

Es evidente que el resultado obtenido ha de ser muy inferior al original, y que el interés de esta tarea está en la adaptación a una herramienta menos potente. Por tanto, quien no esté interesado en esta programación es mejor que disfrute del vídeo original sin embarcarse en el proceso que aquí he seguido.

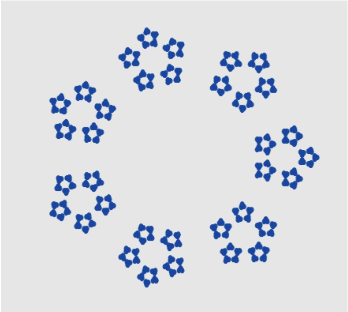
El resultado lo tienes en
http://hojamat.es/blog/arbofactor.xlsm

Ideas previas

Para imitar lejanamente el vídeo necesitaremos concretar aspectos del problema en sí mismo (representar cada factor como un polígono) y después superar las carencias de la hoja de cálculo (en esta ocasión, dado el distinto comportamiento de las mismas en los gráficos, lo he desarrollado sólo en Excel)

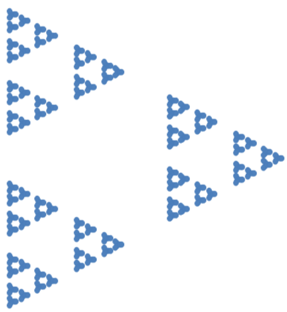
El gráfico

La única posibilidad que nos ofrece Excel para estas representaciones es el gráfico de dispersión. Tiene la ventaja de no depender del orden de los datos y adaptarse muy bien al uso de coordenadas cartesianas (X,Y). También permite dejar la zona en blanco y ocultar los ejes. Por contra, presenta gran dificultad en cambiar el tamaño de los distintos puntos según el número de factores. Todos los esquemas, pues, presentarán el mismo tamaño en los puntos



Aquí puedes ver el esquema para 1050. Se puede observar que los últimos triángulos de puntos parecen confundirse, por no haber controlado el tamaño.

Salvo este inconveniente, las figuras resultan atractivas. Las he orientado circularmente en lugar de buscar siempre la vertical. En la siguiente imagen puedes observar la estructura casi fractal que presentan los números que como el 486 contienen potencias altas de primos.



Lo único que hay que explicar del gráfico es que su área de datos está formada por las columnas B y C en las que volcaremos las coordenadas adecuadas.

El algoritmo

Sacar los primos

Necesitamos, en primer lugar, la lista de factores primos del número, con repetición y en orden decreciente. He aludido bastante en mi blog a estas técnicas, por lo que a nuestros seguidores les resultará familiar. Normalmente se comienzan a buscar los factores pequeños, pero en este trabajo, por razones estéticas, se comienza por los mayores. Por eso al final de la rutina se invierte el orden.

```
' se ordenan al revés

nprimos = indi
For indi = 1 To nprimos
xx(indi) = ff(nprimos - indi + 1)
Next indi Sub sacaprimos(n)
Dim f, a, indi

a = n
f = 2: nprimos = 0
indi = 0

While f * f <= a
While a / f = Int(a / f)
  indi = indi + 1
 ff(indi) = f  'La variable ff va recogiendo los primos
a = a / f
Wend
```

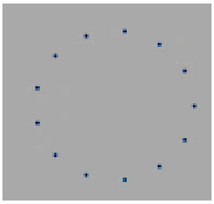
```
If f = 2 Then f = 3 Else f = f + 2
Wend
If a > 1 Then  'Último factor
indi = indi + 1
ff(indi) = a
End If


For indi = 1 To nprimos
ff(indi) = xx(indi)
Next indi
End Sub
```

Después de esta rutina tendremos el número de factores primos en la variable **nprimos** y sus valores en **ff(i)**. En este caso no nos interesan los exponentes.

Recorrido en cada factor

Una vez obtenidos los factores primos necesitamos convertirlos en polígonos. Para cada uno harán falta las coordenadas del centro (xx(i),yy(i)), su radio rr(i) y un contador de puntos ii(i) que nos evite el problema de los decimales al final de cada polígono. En cada paso incrementaremos el ángulo de giro necesario para que se forme el polígono y el contador de lados



En la imagen he comenzado con ff(1)=13, por lo que los elementos se incrementarán así:

ii(i) = ii(i) + 1
aa(i) = aa(i) + dospi / 13

Marcado el ángulo que va girando el punto pasamos de coordenadas polares a cartesianas y volcamos el punto en las columnas B y C, donde lo capturará el gráfico y aparecerá un punto nuevo.

fila = fila + 1
px = xx(i) + rr(i) * Cos(aa(i))
py = yy(i) + rr(i) * Sin(aa(i))
ActiveWorkbook.Sheets(1).Cells(fila, 2).Value = px
ActiveWorkbook.Sheets(1).Cells(fila, 3).Value = py

Esto se repite tantas veces como indique el factor y se formará el polígono básico

Algoritmo de cambio de nivel

Aquí reside el nudo de esta programación. Es un caso claro de procedimiento recursivo, pero como hay que gestionar bastantes parámetros lo he dejado para una posible extensión y se ha usado en su lugar un esquema muy sencillo que ya se ha publicado en mi blog: la subida y bajada de nivel.

Procedimientos iniciales: definir primer radio, sacar los primos…

Mientras haya un nivel activo (nivel>0)

Incrementamos el ángulo y el contador para avanzar en el polígono

Si se llega al final del polígono

He terminado y bajamos de nivel (nivel=nivel-1)

En caso contrario pueden ocurrir dos cosas:

Si hemos llegado al último factor hay que imprimir el punto volcándolo en las columnas B y C

Si no hemos llegado hay que subir de nivel(nivel=nivel+1)

Esto significa que hay que determinar nuevos centro y radio

Fin del Mientras

Fin de la rutina

No incluimos el código y preferimos que las personas interesadas lo estudien en el archivo de Excel.

Animación

Para conseguir la animación basta con una estructura repetitiva tipo FOR-NEXT y la creación de pausas para conseguir el efecto de transición continua. El problema radica en que cualquier operación aparentemente sencilla puede borrar el gráfico y perderse su persistencia en nuestra retina. He tenido que acudir al recálculo para refrescarlo y que no desaparezca.

La pausa se consigue leyendo el reloj e introduciendo a la hoja en un bucle continuo hasta que transcurra la pausa. Queda así:

Call arbol “se construye el árbol de factores

t1 = Timer ‘leemos el reloj y tomamos nota en t1 y t2

t2 = t1

While t2 - t1 < pausa: t2 = Timer: Calculate: Wend ‘bucle continuo de recálculo

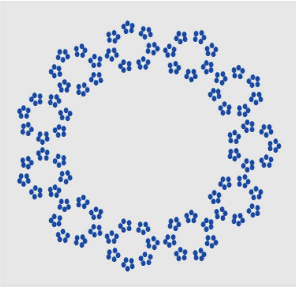
Call borrar ’terminada la pausa se borra el gráfico

Controles

Nos gusta tener todo el poder posible sobre la hoja de cálculo. Por eso se han añadido los controles de

Reducción, que se puede cambiar (no en la animación) para mejorar la estética del gráfico y **Pausa**, que ralentiza o acelera la animación.

A quienes hayan llegado hasta aquí les recomiendo el estudio de los detalles del código e intentar mejorarlo.



ALGORITMO DE EUCLIDES BINARIO

En mi blog motiva mucho el construir un esquema en hoja de cálculo que explique de la mejor forma posible el funcionamiento de un proceso o un algoritmo. Lo haremos hoy con la variante binaria del Algoritmo de Euclides.

Restar en lugar de dividir

Ya se sabe que el algoritmo de Euclides por divisiones se puede sustituir por otro efectuado a base de restar los dos números. Parece más lento, pero al ahorrar divisiones puede resultar más eficiente. Hace sesenta años lo usábamos los alumnos de Bachillerato (con once añitos) para comprobar si dos segmentos eran inconmensurables o no. Llevábamos uno sobre otro y nos quedábamos con la diferencia. Si al reiterar llegábamos al segmento nulo, es que tenían una medida común. Aquello era Geometría de Euclides pura.

Si el algoritmo clásico se organizaba a partir de la división euclídea, en esta variante se usa la resta. Así, para encontrar MCD(96,36) por divisiones sería: 96=2*36+24; 36=1*24+12; 24=2*12+0, luego el MCD(96,36) =12. Por restas formaríamos estas parejas: (96,36), (60,36), (36,24), (24,12), (12,12), (12,0), llegando al mismo resultado.

Eliminar el factor 2 siempre que se pueda.

Ya sabemos que en computación con sistema de numeración binario la división y la multiplicación por 2 se reducen a trasladar un lugar a la derecha o izquierda del dígito que se multiplica. Por eso, es interesante dar protagonismo al número 2 en los cálculos. Una primera idea es que si expresamos los dos números A y B de la forma $A=2^m*p$, $B=2^n*q$, con p y q los mayores divisores impares, el M.C.D se puede encontrar con dos cálculos por separado. Por una parte quedándonos con el menor exponente del 2 y por otra hallando MCD(p,q).

En el algoritmo que se está presentando, se elimina en primer lugar la potencia de 2 común a ambos números, y se toma nota de ella. Una vez eliminada, el factor 2 no va a influir en el resultado, y cuando aparezca en alguno de los dos números se podrá igualmente suprimir. En términos binarios suprimir un 2 es trasladar los dígitos una posición hacia la derecha.

En Wikipedia y otras páginas que he consultado expresan lo anterior en forma de tres reglas.

http://en.wikipedia.org/wiki/Binary_GCD_algorithm. No son difíciles de razonar.

(1) Si A y B son ambos pares se cumple MCD(A,B)=2*MCD(A/2,B/2)

Esto justifica que el primer paso que demos sea el de separar las potencias de 2.

(2) Si A es par y B impar se tiene
MCD(A;B)=MCD(A/2,B)

Igualmente se aplicaría si B es par y A impar. En virtud de esta regla eliminaremos todos los factores 2 una vez que se ha separado la potencia común.

(3) Si ambos A y B son impares y A<B,
MCD(A,B)=MCD(B-A,A). Si es B<A,
MCD(A,B)=MCD(A-B,B)

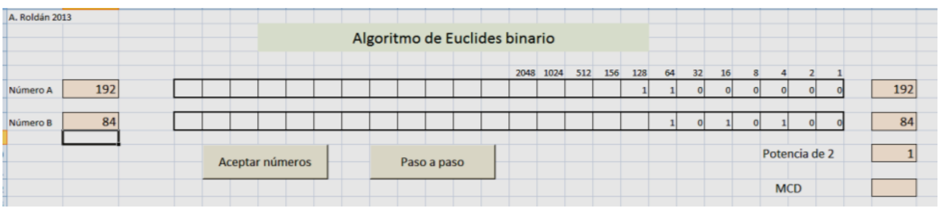
Esta es la esencia del algoritmo de Euclides, restar ambos números.

Eliminar la potencia de 2 común (Regla 1)

He creado una demo en hoja de cálculo para seguir visualmente los pasos del algoritmo binario. Lo puedes descargar desde la dirección

http://hojamat.es/sindecimales/divisibilidad/herramientas/herrdiv.htm#euclibin

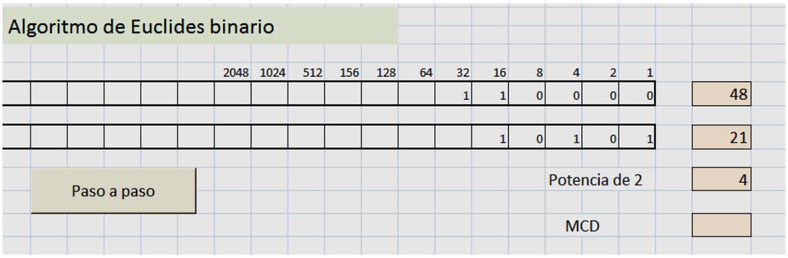
Como nuestro objetivo es visual, desarrollaremos ahora los pasos sugeridos pero en forma de esquema. En una hoja de cálculo se escribirán los dos números y con un botón iterativo se irán avanzando pasos hasta llegar al MCD. Esta primera fase de eliminar el factor común lo puedes ver en las imágenes siguientes:



En primer lugar hemos escrito los números 192 y 84. Al pulsar sobre el botón ***Aceptar números*** se han convertido ambos en binario y se han reconstruido a la derecha. La potencia de 2 común figura al principio con el valor 1.

Observamos que ambos números terminan en dos ceros (en binario), luego compartirán el factor 2^2=4.

Según estamos explicando, el primer paso del algoritmo binario es eliminar el factor 2 común. Si ahora usamos el botón ***Paso a paso*** dos veces veremos que los dígitos de ambos números se mueven dos posiciones a la derecha y que la potencia de 2 se convierte en 4.



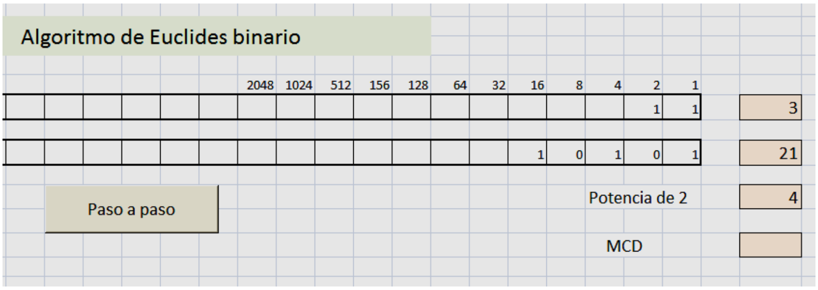
A la derecha figuran los números ya simplificados, 48 y 21, y la potencia de 2, que nos servirá al final.

Resto de pasos

Ahora la estrategia es triple:

(1) Si el primer número contiene aún potencias de 2, se eliminan (Regla 2)

Observa que en nuestro ejemplo el número 48 termina en cuatro ceros, luego al cabo de cuatro toques del botón ***Paso a paso*** desaparecerán.



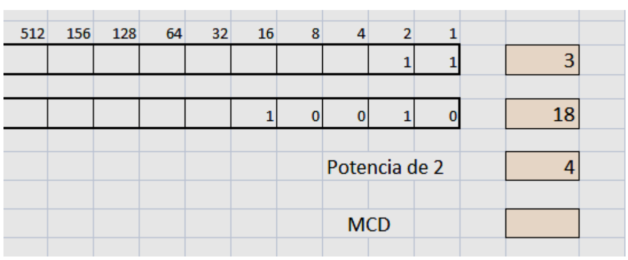
(2) Se opera igual con el segundo: se le elimina el factor 2 (Regla 2)

En nuestro ejemplo ya no quedan factores 2 (por ahora)

(3) Si ambos son impares, se sustituye el mayor de ellos por la diferencia entre ambos.

Este es el núcleo del algoritmo de Euclides por diferencias. En la práctica quizás haya que intercambiar los valores de A y B, pero no entraremos en esos detalles.

Al restar dos impares se **producirán nuevos factores 2**, por lo que en la siguiente pasada del algoritmo los eliminará. Lo ves en las siguientes dos imágenes:



512	156	128	64	32	16	8	4	2	1	
								1	1	3
						1	0	0	1	9
						Potencia de 2				4
						MCD				

Reiteramos estas tres reglas hasta que lleguemos al valor 0. En ese momento el algoritmo construye el M.C.D. multiplicando el resultado por la potencia de 2 que tenía almacenada.

Aquí lo tienes:

512	156	128	64	32	16	8	4	2	1	
								1	1	3
										0
						Potencia de 2				4
						MCD				12

Visto así, en hoja de cálculo, no parece ser nada del otro mundo, pero todas las operaciones que realiza son altamente eficientes en el sistema de numeración binario. Por algo lo introdujo el programador israelí Stein en 1967. Aquí sólo se nos queda como un tema de cultura matemática, pero es divertido implementarlo.

Versión recursiva

Lo que he desarrollado, con un botón que en cada paso reacciona según lo que se encuentra, nos permite sospechar que todo esto se resuelve también mediante una función recursiva. Es cierto, y está publicado. Lo que haremos aquí es adaptarlo al Basic de Excel y Calc:

Function mcdbin(m, n)
Dim mb

'Si son iguales, he llegado al MCD
If m = n Then
mb = m

'Si ambos son divisibles entre 2, se saca ese factor
ElseIf m / 2 = m \ 2 And n / 2 = n \ 2 Then
mb = 2 * mcdbin(m / 2, n / 2)

'El primero contiene un 2. Se elimina
ElseIf m / 2 = m \ 2 Then
mb = mcdbin(m / 2, n)

'Operamos de igual forma con el segundo
ElseIf n / 2 = n \ 2 Then
mb = mcdbin(m, n / 2)

'Ambos son impares. Se restan

Else
If m > n Then mb = mcdbin(m - n, n)
If n > m Then mb = mcdbin(m, n - m)
End If
'La función recoge el valor de mb
mcdbin = mb
End Function

Tiene toda la elegancia de las funciones recursivas

(ver http://hojaynumeros.blogspot.com.es/2012/03/funciones-recursivas-en-las-hojas-de.html)

En este caso resulta un poco complicada, pero funciona muy bien. Si te apetece, estudia la versión clásica y recursiva:

Public Function mcdeuclid(m, n)
Dim mb

'Si uno es múltiplo de otro, obtenemos el MCD
If m / n = m \ n Then
mb = n
ElseIf n / m = n \ m Then
mb = m
Else
'En caso contrario, se restan

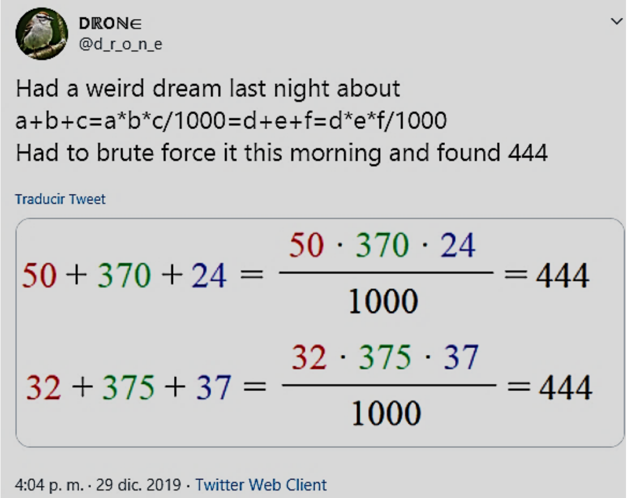
```
If m > n Then mb = mcdeuclid(m - n, n)
If n > m Then mb = mcdeuclid(m, n - m)
End If
'La función recoge el valor de mb
mcdeuclid = mb
End Function
```

Ambas están implementadas en la herramienta que ofrezco.

	Versión recursiva	
Variante binaria	Función MCDBIN	22
Variante clásica	Función MCDEUCLID	22

USO DE LA FUERZA BRUTA

El día 29/12/19 descubrí este tweet de @d_r_o_n_e:



En él podemos ver un ejemplo que puede reproducirse mediante el uso de la "fuerza bruta. Consiste en recorrer todas las variantes de un problema sin usar razonamientos ni condiciones complementarias. Es una buena estrategia comenzar con esta forma de buscar para después ir afinando resultados, explicarlos y, si es posible, justificarlos.

Uso de la herramienta Cartesius

Nuestra herramienta Cartesius también ayuda a combinar variables de todas las formas posibles, pero se hace lenta cuando nos acercamos al rango de los números de cuatro cifras. La puedes descargar desde http://www.hojamat.es/sindecimales/combinatoria/herramientas/herrcomb.htm#cartesius

Con ella he intentado este planteo:

XTOTAL=3
XT=1..200
ES X1*X2*X3/1000=X1+X2+X3
CRECIENTE

Es fácil entender su significado: combinaremos tres variables, todas entre 1 y 200, de forma que se cumpla la condición pedida y después nos quedamos con las crecientes. Al cabo de más de media hora se obtuvo esta tabla, completada en sus dos últimas columnas con las dos expresiones que deben coincidir.

X1	X2	X3	X4	SUMA	PRODUCTO/1000
12	170	175		357	357
15	125	160		300	300
15	132	150		297	297
16	125	141		282	282
20	75	190		285	285
20	78	175		273	273
20	85	150		255	255
20	100	120		240	240
21	100	110		231	231
22	84	125		231	231
25	60	170		255	255
25	65	144		234	234
25	66	140		231	231
25	80	105		210	210
25	90	92		207	207
27	64	125		216	216
28	50	195		273	273
30	46	200		276	276
30	50	160		240	240
30	65	100		195	195

Observamos, y lo confirmaremos un poco más abajo, que aparecen repetidas algunas soluciones menores que 444, como son 231 o 240. Con el planteo propuesto se encontraron 32 soluciones, de las que solo hemos reproducido las primeras. Aparecen en orden inverso al natural porque cuando unos factores tienen igual suma, su producto crece cuando sus diferencias son menores. Con este intento descubrimos ya que la técnica de la "fuerza bruta" es muy lenta en producir resultados.

Analizando la búsqueda descubrimos que Cartesius ha tenido que analizar 200*200*200 =8*10^6 números, y en cada uno calcular si una igualdad se verifica o no. Eso es mucho para un portátil normal. Ahí es donde falla la fuerza bruta, en la multiplicación de casos que produce la Combinatoria.

Algoritmos

La "fuerza bruta" se caracteriza casi siempre por el uso de bucles del tipo FOR_NEXT, WHILE o REPEAT, casi siempre anidados en tres o cuatro niveles.

Lo normal, en ejemplos similares al que nos ocupa, es disponer de tres bucles anidados, con la propiedad deseada en el interior de los tres. Comenzaremos exigiendo solo una condición de las propuestas por @d_r_o_n_e

En este caso podíamos comenzar por este código:

```
Sub fuerzabuta()
Dim i, j, k, a, b, fila

fila = 10   'La fila determina la construcción de una tabla en Excel
For i = 1 To 1000  'Bucle triple
For j = 1 To i
For k = 1 To j
a = i + j + k  'Cálculos previos
b = i * j * k / 1000
If a = b Then  'Condición pedida
fila = fila + 1: 'Construcción de la tabla
ActiveWorkbook.ActiveSheet.Cells(fila, 2).Value = b
ActiveWorkbook.ActiveSheet.Cells(fila, 3).Value = i
ActiveWorkbook.ActiveSheet.Cells(fila, 4).Value = j
ActiveWorkbook.ActiveSheet.Cells(fila, 5).Value = k
End If
Next k
Next j
Next i
End Sub
```

He ejecutado esta macro de Excel y nos han resultado muchas soluciones. Lo que nos interesa es que salgan repetidas. Por la forma de plantear el problema, aparecerán desordenadas. Las primeras han sido:

Número	i	j	k
165	60	55	50
168	70	50	48
171	76	50	45
186	80	75	31
189	84	75	30
180	90	50	40
207	92	90	25
189	100	54	35
192	100	60	32
195	100	65	30
210	105	80	25
231	110	100	21
204	120	50	34

Coinciden con las obtenidas en Cartesius. Observamos su falta de orden y la existencia de un repetido, el 189. Según la tabla:

189=84+75+30=84*75*30/1000

189=100+54+35=100*54*35/1000

Es una solución también más pequeña que la propuesta de 444.

Para verlas todas ordenaremos la columna y así se verán mejor los repetidos. Con este método hemos descubierto los siguientes: 189, 207, 231, 240, 255, 273, 297, 420, 444, 480, 504, 741, 759, 768, 810, 891, … De ellos presentan soluciones triples 231, 504 y 891.

Más fuerza bruta (o menos)

Podíamos intentar descubrir tan solo los números en los que se da más de una solución. El problema es que para esto se necesitaría un bucle más, con el consiguiente aumento de tiempo de proceso. Es el coste de utilización de bucles múltiples sin apenas condicionamientos. Hay una forma de evitar un nuevo bucle, y es considerar que en el algoritmo anterior hemos hecho variar el valor de **k** cuando en realidad está condicionado por la igualdad que se pide ***a=i+j+k.*** Considerándolo así, lo único que ha de cumplir k es que su valor sea ***a-i-j*** y, por cuestión de unicidad, que no sea mayor que **j**. Esto es lo que hemos implementado en PARI:

```
for(n=1,500,m=0;b=0;for(i=1,n,for(j=1,i,c=i+j;if(c<=n&&n-c<=j,b=i*j*(n-c)/1000;if(b==n,m+=1))));if(m>1,print1(n,", ")))
```

Mantenemos los bucles con las variables i y j. Eliminamos el bucle de k sustituyéndolo por la expresión ***b=i*j*(n-c)/1000*** y concretamos las condiciones que ha de cumplir. Con la variable **m** exigimos que haya repetición de casos. Hemos añadido un nuevo bucle, pero con los cambios apenas se resiente la velocidad del proceso. De todas formas, para rangos de números de

1000 más o menos, puede tardar muchos minutos o incluso más de una hora. Cosas de la fuerza bruta.

Los primeros resultados son:

189, 207, 231, 240, 255, 273, 297, 420, 444, 480, 504, 741, 759, 768, 810, 891, 1221, 1320, 2418,…

Con esto damos por terminada la búsqueda, porque la fuerza bruta cansa y no se aprende mucho con ella.

Rebajamos pretensiones

Podíamos exigir productos similares, pero con solo dos variables, es decir, N=i+j=i*j/100. Esto simplifica el problema, y solo lo incluimos como repaso de las técnicas empleadas anteriormente. Las explicaciones para el caso anterior valen también para este.

Con Cartesius

Plantearíamos, por ejemplo:

xtotal=2
xt=1..700
es x1+x2=x1*x2/100
creciente

Obtendríamos:

X1	X2	X3	X4	X5
120	600		720	720
125	500		625	625
140	350		490	490
150	300		450	450
180	225		405	405
200	200		400	400

Las primeras columnas corresponden a los valores de i, j y las siguientes el valor repetido de N.

Así, 120+600=120*600/100=720

125+500=125*500/100=625

En principio, no parece que existan soluciones dobles.

Con una función de Excel:

Function esmultiple2(n)
Dim i, j, k, a, b, m

m = 0
For i = 1 To n
For j = 1 To i
a = i + j
b = i * j / 100

```
If a = b And a = n Then m = m + 1
Next j
Next i
esmultiple2 = m
End Function
```

Esta función cuenta las veces en las que se da la igualdad ***i+j=i*j/100***. Organizando una búsqueda nos resulta:

400
405
450
490
625
720
841
1210
1458

Tampoco se aprecian repetidos

Con PARI

Traducimos la función anterior a PARI y la integramos en un bucle de búsqueda:

```
for(n=1,10000,m=0;b=0;for(i=1,n/2,b=i*(n-i)/100;if(b==n,m+=1));if(m>0,print1(n,", ")))
```

Volvemos a obtener los mismos resultados:

400, 405, 450, 490, 625, 720, 841, 1210, 1458, 2205, 2704, 5202, …

Tampoco aquí se detectan repetidos. Lo dejamos como complemento.

GENERACIÓN POR SUMAS ALTERNAS

El contenido de este estudio no tendrá valor teórico destacable. Su objetivo es analizar el procedimiento para encontrar sumas de números alternados que generen uno dado N. Por alternados entenderemos consecutivos de cierto tipo en los que se suprime uno de cada dos. También podríamos afirmar que "se salta" un elemento y se elige el siguiente.

Por ejemplo, 84=1+9+25+49, y esa suma es alternada de cuadrados, porque hemos elegido una suma de consecutivos y le hemos ido suprimiendo un término de cada dos. Los representamos entre paréntesis:

84=1+(4)+9+(16)+25+(36)+49

Otro ejemplo es el publicado en

https://oeis.org/A300395,

que contiene todos los números que se forman con una suma de nueve primos alternados. Por ejemplo:

521 = 23+31+41+47+59+67+73+83+97

En este conjunto se han ido suprimiendo los primos 29, 37, 43, 53, …

Queda claro, pues, lo que entenderemos por alternados.

Un número N puede ser equivalente a varias sumas de este tipo. Por eso, los algoritmos que busquen dichas sumas necesitarán dos bucles, pues primero hay que analizar el inicio de la suma y luego su longitud. Para cada inicio existirán varias posibles sumas, lo que exige los dos bucles.

En mi blog he usado la función PROXIMO en la entrada "En el punto medio de dispares"

(https://hojaynumeros.blogspot.com/2023/06/en-el-punto-medio-de-dispares.html) para ir encontrando sumandos del tipo dado si conocemos el primero.

El siguiente listado es el usado para encontrar el próximo primo, ya que usa la función ESPRIMO, pero se adapta fácilmente a la búsqueda de otros tipos de números, usando las funciones ESTRIANGULAR, ESCUBO, ESOBLONGO, … todas usadas en mi blog y accesibles con el comando Buscar.

```
Function proximo(a) As Long
Dim p, prim As Long
Dim sale As Boolean
p = a + 1: sale = False: prim = 0
While Not sale 'Reitera hasta que aparezca el próximo
If escuad(p) Then prim = p: sale = True
p = p + 1
Wend
```

proximo = prim
End Function

La función PROXIMO puede resultar lenta, por lo que si buscamos una suma alternada para un número N, es muy conveniente guardar en un vector todos los números comprendidos entre 1 y N y que sean del tipo dado. También tiene el peligro de no parar nunca, pero en los casos habituales esto no sucede.

Estructura del algoritmo

Según las consideraciones anteriores, para encontrar las sumas deseadas deberemos recorrer tres etapas:

1) Hay que encontrar todos los posibles sumandos del tipo dado entre 1 y N. Lo explicamos con un ejemplo, el de encontrar una suma de triangulares consecutivos alternados cuyo resultado sea 200.

En primer lugar, usando la función PROXIMO, creamos un vector que contenga todos los triangulares comprendidos entre 1 y 200. Si llamamos **s** al vector, tendríamos que s(1)=1, s(2)=3, s(3)=6, …s(18)=171 y s(19)=190. Tomamos nota de que existen 19 triangulares que pueden ser sumandos.

El uso de un vector nos facilita el “saltar” un elemento, pues se pasará de s(k) a s(k+2) en cada paso.

2) Se establece un bucle de búsqueda del primer sumando, que comenzará en el 1 y se avanzará hasta 190. En el ejemplo tendríamos que llegar, como veremos, hasta un inicio de 10.

3) Para cada posible inicio de la suma se construirá un segundo bucle que vaya añadiendo un sumando alternado en cada paso. Para ello se irá incrementado el índice en dos unidades cada vez. Si en uno de ellos se alcanza el resultado de N, se habrá conseguido el objetivo. Para el número 200 la solución sería

10+21+36+55+78=200. Habríamos saltado los triangulares 15, 28, 45, y 66.

Estas fases se pueden implementar en una subrutina para VBasic con objeto de usarla en Excel o Calc.

Rutina en VBasic

El listado que sigue ha sido diseñado para Excel, pero con un pequeño cambio se adapta a LibreOffice Calc.

fila = 14 ‘O cualquier otro valor

‘Se lee el valor de N y el tipo de sumando

n = ActiveWorkbook.Sheets(1).Cells(9, 4).Value

tipo = ActiveWorkbook.Sheets(1).Cells(29, 8).Value

‘Con el uso de la función PROXIMO se carga un vector con candidatos a sumandos, para acelerar el proceso

i = 0
k = 0
While a <= n
a = proximo(a, tipo)
If a <= n Then i = i + 1: s(i) = a: k = i
Wend

‘Se ha construido el vector **s**

For i = 1 To k ‘Bucle para elegir el inicio de la suma
j = i
a = s(i)
suma = a
st$ = Str$(a)
While suma <= n And j <= k – 2 ‘Bucle para construir una suma
j = j + 2 ‘Es importante añadir un 2, para alternar
a = s(j)
suma = suma + a ‘Se construye la suma
st$ = st$ + ", " + Str$(a) ‘Modo texto para la suma
If suma = n Then
st = st + " # " + Str$((j - i) / 2 + 1)
‘Se presenta la solución en modo texto

ActiveWorkbook.Sheets(1).Cells(fila, 10).Value = st$
ActiveWorkbook.Sheets(1).Cells(fila, 12).Value = rotulo(tipo)
End If
Wend
Next i
End Sub

En la imagen vemos la forma en la que aparecería la solución en Excel:

10, 21, 36, 55, 78 # 5		Triangulares

Se leerían los sumandos, el número de ellos, 5 en este caso, y el tipo, “Triangulares”. Lo del este último rótulo no lo hemos explicado aquí.

Este algoritmo se podría haber construido como una función, pero esta vez he optado por una rutina, pues uso una similar en una de mis herramientas.

Uno de los últimos usos, por si deseas reproducirlo, es comprobar que el número 5623 equivale a la suma de nueve números primos alternados y que, por tanto, pertenece a la sucesión https://oeis.org/A300395

5623=577+593+601+613+619+641+647+659+673

Si se dispone de varios tipos de números (en mis trabajos uso 17), se podría construir una gran rutina que recorriera varios tipos para buscar sumas. En la siguiente imagen lo he aplicado al número 200 con un resultado sorprendente por el número de soluciones:

97, 103 # 2	Primos
36, 64, 100 # 3	Cuadrados
10, 21, 36, 55, 78 # 5	Triangulares
46, 49, 51, 54 # 4	Compuesto
11, 13, 15, 17, 19, 21, 23, 25, 27, 29 # 10	Entero
18, 20, 22, 24, 26, 28, 30, 32 # 8	Entero
36, 38, 40, 42, 44 # 5	Entero
47, 49, 51, 53 # 4	Entero
99, 101 # 2	Entero
94, 106 # 2	Semiprimos

Entre ellos figura el que he usado de ejemplo con triangulares.

Con esta idea finalizo el estudio, ya que el objetivo era diseñar y explicar un algoritmo.

LAS OLVIDADAS FRACCIONES CONTINUAS

PRESENTACIÓN

¿Sabes qué significa este desarrollo y cómo obtenerlo?

$$\frac{1280}{345}=3+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{2+\cfrac{1}{4+\cfrac{1}{2}}}}}$$

Si no conoces la teoría inténtalo según tus conocimientos.

Puedes comenzar así:

1280/345 = 3 + 245/345 = 3 + 1/(345/245) =3 + 1/(1+100/245) =

3 + 1/(1 + 1/(245/100)) =…

DESARROLLO

Llamamos fracción continua a la expresada de esta forma:

$$a+\cfrac{1}{b+\cfrac{1}{c+\cfrac{1}{d\ldots}}}$$

donde **a** es entero y **b, c**…son enteros positivos llamados **cocientes**. Toda fracción ordinaria se puede expresar de esta forma, y todo número irracional admite aproximaciones mediante desarrollos de este tipo. Las fracciones continuas se usan cuando se desea manejar una representación de los números reales independiente del sistema de numeración (salvo en la expresión de los cocientes).

No es este libro el espacio más adecuado para estudiar todo su desarrollo teórico. Nuestro interés aquí será la implementación de los algoritmos necesarios en hoja de cálculo para desarrollar un número en fracciones continuas y las aplicaciones que derivan de ello.

Dos líneas podemos seguir en la obtención de los cocientes. Una está esbozada en la página anterior y la desarrollaremos más adelante. La otra se basa en el algoritmo de Euclides.

Método del algoritmo de Euclides

Si consultas la teoría descubrirás que los cocientes a, b, c,… son los que aparecen en el algoritmo de Euclides para el cálculo del m.c.d. de dos números. Así, por ejemplo, para encontrar el m.c.d. de 345 y 1280 en el algoritmo se obtienen los siguientes cocientes:

	3	1	2	2	4	2	0	0	0
1280	345	245	100	45	10	5	0	0	0
245	100	45	10	5	0	0	0	0	0

En el desarrollo mediante fracciones continuas de 1280/345 vuelven a aparecer los mismos cocientes 3, 1, 2, 2, … ¡porque se trata del mismo algoritmo orientado de forma diferente! En la siguiente imagen, capturada de una hoja de cálculo

Numerador	1280							
		Fracción continua:	3	1	2	2	4	2
Denominador	345		3	4	11	26	115	256
			1	1	3	7	31	69

puedes comprobar la evidente igualdad de la serie de cocientes. Comprueba que, efectivamente, es válido este desarrollo:

$$\frac{1280}{345}=3+\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{2+\cfrac{1}{4+\cfrac{1}{2}}}}}$$

El algoritmo de Euclides lo puedes encontrar implementado en varios sitios, por ejemplo, en nuestra página *Hojamat.es*

http://hojamat.es/sindecimales/divisibilidad/herramientas/herrdiv.htm

Como el algoritmo de Euclides tiene siempre un final en un número finito de pasos, las fracciones continuas de cualquier número racional poseerán también un número finito de cocientes.

Método de la parte entera

En la primera página de este tema comenzamos a desarrollar otro método: consiste en ir encontrando la parte entera de las fracciones e invirtiendo el resto R = 1/1/R. Este método es recomendable para desarrollar un número expresado en el sistema decimal, pero con las hojas de cálculo **no es exacto**, pues se van acumulando errores de redondeo y truncamiento.

¿Cómo se pueden organizar los datos?

Escribe aquí el número deseado								
3,71014493		1,41	2,45	2,22	4,5	2	0	0
	Fracción continua:	3	1	2	2	4	2	0

En la imagen vemos que en la fila inferior se van situando las partes enteras del número propuesto y de los que van apareciendo en la fila superior. Estos números, que en la imagen son 3, 1, 2, 2, …son los cocientes del desarrollo.

Los números de la fila superior son los inversos de los restos que se producen al restar las partes enteras. No es difícil de organizar en una hoja de cálculo, porque basta ir copiando las dos fórmulas de parte entera del cociente e inverso del resto para conseguir el desarrollo, sujeto, como hemos dicho a errores de truncamiento y redondeo.

REDUCIDAS

Para construir una pieza, un tornero ha de ajustar unos engranajes de forma que mientras uno gire 2009 vueltas, el otro sólo recorra 2000. En este caprichoso encargo, los números son primos entre sí, por lo que no se pueden simplificar, y el tornero carece de engranajes de 2000 ó de 2009 dientes.

Le pide consejo al oficial. Éste hace unos cálculos y le ofrece la solución: "Usa un engranaje de 222 dientes y otro de 223, que nadie lo va a notar"

¿Qué operación hizo el oficial? ¿Por qué estaba seguro de que la pieza saldría bien fabricada?

Hemos desarrollado 2009/2000 en forma de fracción continua, formada por los cocientes [1, 222, 4,2]

7	Escribe aquí la fracción deseada											
8												
9	Numerador	2009										
10	Denominador	2000		Fracción continua:				1	222	4	2	0
11					Algoritmo de Euclides		2009	2000	9	2	1	0
12							9	2	1	0	0	0
13	Equivalente decimal				Reducidas	0	1	1	223	893	2009	
14	1,00450000					1	0	1	222	889	2000	
15												
16								1,00000000	1,00450450	1,00449944	1,00450000	
17												

Puedes reproducir los resultados con las hojas de cálculo **fraccont.xls** y **fraccont.ods** contenidas en la dirección

http://hojamat.es/sindecimales/aritmetica/herramientas/herrarit.htm

Debajo de los cocientes aparecen una serie de fracciones, llamadas **reducidas o convergentes**, 1/1, 223/222, 893/889, que se van aproximando a 2009/200. Entre ellas figura 223/222, la solución propuesta por el oficial. Puedes ver esta aproximación en los desarrollos decimales que figuran debajo.

Estas reducidas se forman calculando fracciones parciales de izquierda a derecha:

1=1; 1+1/222=223/222; 1+1/(222+1/4) = 1+1/(889/4) =

1+4/889 = 893/889

La hoja de cálculo **fraccont.ods** (en su hoja dedicada a números fraccionarios) logra estas reducidas mediante un algoritmo clásico llamado de “los cumulantes”. Consiste en construir dos sucesiones recurrentes del tipo

$P_n = P_{n-1}*a_n+p_{n-2}$

siendo a_n la sucesión de cocientes de la fracción continua, precedidos en la primera fila por 0 y 1 y en la segunda por 1 y 0. Como ejemplo, si se aplican los cumulantes a la sucesión 1, 1, 1, 1, 1. Resulta la sucesión de Fibonacci 1, 1, 2, 3, 5,8…

Puedes seguir estos cumulantes en las filas que contienen los numeradores y denominadores de las reducidas en el caso de la fracción que usamos al principio, 1280/345. Cada número de las filas de reducidas se ha formado multiplicando el anterior por el cociente de la primera fila y sumando el penúltimo elemento

		3	1	2	2	4	2
0	1	3	4	11	26	115	256
1	0	1	1	3	7	31	69

11 = 2*4+2, 115=26*4+11, 31=7*4+3, y así con todos. No es difícil organizar esto con hoja de cálculo.

Las reducidas permiten la aproximación a una fracción con numerador y denominador grandes mediante otras que están construidas con números más pequeños. Esta utilidad la usaban los torneros cuando carecían de ruedas de determinado número de dientes y debían sustituirlas, con un pequeño error, por otras ruedas más pequeñas.

Por ejemplo, 2009/2000 se puede sustituir por 223/222 con un error inferior a 0,000005.

La reducidas son alternativamente mayores y menores que la fracción dada, y se acercan a ella, pues la diferencia entre dos reducidas es siempre igual a la unidad dividida entre el producto de sus denominadores.

ECUACIONES DIOFÁNTICAS

¿Cómo se pueden repartir 5957 objetos en lotes de 161 y de 182 objetos respectivamente, sin que sobre ni falte ninguno?

Puedes usar la fuerza bruta de las hojas de cálculo (tabla de doble entrada, multiplicaciones y sumas hasta ver el total 5957).

También dispones de la herramienta Solver. Aquí tienes una imagen de su planteamiento:

Con estas herramientas obtendrías las soluciones X=11 Y=23

¿Cómo lo resolverías sin ordenador?

Otra aplicación importante de las fracciones continuas y sus reducidas es la de resolver ecuaciones diofánticas lineales del tipo Ax+By=C, en las que C es múltiplo del MCD de A y B (que son las únicas que poseen solución). Quiere esto decir que A, B y C se pueden simplificar hasta conseguir que MCD(A,B)=1. En lo que sigue supondremos que esto se cumple.

Efectivamente, en un apartado anterior se vio que la diferencia entre dos reducidas consecutivas equivalía a una fracción de numerador la unidad y de denominador el producto de sus denominadores. Esta propiedad también se cumple entre la última reducida y la fracción dada.

Vemos cómo se aprovecha esta propiedad para resolver la ecuación.

Sea, por ejemplo, la ecuación 244X+108Y=112.

Simplificamos: 61X+27Y=28, con MCD(61,27)=1

Buscamos las reducidas de la fracción 61/27 y elegimos la última 9/4

Y

Fracción continua:				2	3	1	6
	Algoritmo de Euclides		61	27	7	6	1
			7	6	1	0	0
	Reducidas	0	1	2	7	9	61
		1	0	1	3	4	27

se cumplirá, según la propiedad citada, que 61*4-27*9=1, luego 4 y -9 serán las soluciones de 61X+27Y=1. Bastará multiplicar por el término independiente 28 para obtener una solución: X=4*28 = 112 e Y=-9*28 = -252

Las demás soluciones se obtienen mediante las paramétricas.

X=112-27t

Y=-252+61t

Si se desean soluciones positivas deberemos ajustar el parámetro t

En el ejemplo propuesto anteriormente hay que resolver la ecuación diofántica 182X+161Y= 5957. Si usas la hoja de cálculo **diofant1.ods** (OpenOffice) o la **diofant1.xls** (Excel) contenidas en la dirección

http://www.hojamat.es/sindecimales/aritmetica/herramientas/herrarit.htm

verás que las soluciones dadas en primer lugar son X=6808 Y=-7659. Para conseguir dos soluciones positivas hay que jugar bastante con el parámetro T. Intenta conseguirlo, que no es inmediato. Una solución sería X=23 Y=11.

APROXIMACIÓN DIOFÁNTICA

*¿Sabías que la fracción **3650401/2107560** es una muy buena aproximación de la raíz cuadrada de 3 (coinciden en los trece primeros decimales)? ¿Cómo se ha obtenido?*

Cualquier número expresado en forma decimal puede representarse mediante una fracción continua. Si el número es racional, ésta será finita, pero si es irracional no podrá serlo, y tendríamos que prolongar el desarrollo de la fracción continua hasta el infinito. En los siguientes párrafos veremos cómo.

Un caso muy interesante es el de los irracionales cuadráticos, que, como demostró Lagrange, presentan desarrollos periódicos.

¿Cómo desarrollar un decimal cualquiera en fracción continua exacta (caso racional) o aproximada (si es irracional)?

La idea es: separamos la parte entera y la parte decimal del número N=e+d, y la decimal la expresamos así: N=e+1/(1/d). Volvemos a separar parte entera y decimal de 1/d y reiteramos, con lo que irán apareciendo los cocientes enteros de una fracción continua.

Probamos con la raíz cuadrada de 3. Los cálculos serían:

1,73205080757 = 1+ (1/(1/1,73205080757) =

1+ 1/1,36602540378 = 1+ 1/(1+1/2,73205080760) =

1+1/(1+1/(2+1/1,36602540378) =…

Al salir este último número se descubre la periodicidad, luego 1,73205080757 equivale a la fracción continua

[1, 1, 2, 1, 2, 1, 2, …] que es periódica por tratarse de un irracional cuadrático.

Para evitar los errores de truncamiento y redondeo deberás organizar los cálculos sin acudir a su expresión decimal, manteniendo el radical cuadrático en todos ellos. En este caso habrá que acudir en cada paso a la racionalización de los denominadores mediante el producto por el radical conjugado.

Incluimos a continuación el desarrollo para √8, en el que se van destacando los cocientes obtenidos:

$\sqrt{8} = 2 + \frac{1}{1/\left(\sqrt{8}-2\right)} = 2 + \frac{1}{\left(\sqrt{8}+2\right)/4}$ luego $a_1 = 2$

$\frac{\sqrt{8}+2}{4} = 1 + \frac{1}{1/\left(\sqrt{8}-2/4\right)} = 1 + \frac{1}{\sqrt{8}+2}$ luego $a_2=1$

$\frac{\sqrt{8}+2}{1} = 4 + \frac{1}{1/\left(\sqrt{8}-2\right)} = 4 + \frac{1}{\left(\sqrt{8}+2/4\right)}$ luego $a_3 = 4$

Se observa que llegamos a la misma expresión que cuando obtuvimos $a_1 = 2$, luego hemos llegado a la periodicidad, y

√8 = [2, 1, 4, 1, 4, …]

A continuación, destacamos algunos desarrollos importantes:

$\sqrt{2} = [1,2,2,2,2...]$

Números metálicos

Número de oro

$$\phi = \frac{\sqrt{5}+1}{2} = [1,1,1,1,1\ldots]$$

Número de plata

$$\varphi = 1+\sqrt{2} = [2,2,2,2\ldots]$$

Número de bronce

$$\psi = \frac{\sqrt{13}+3}{2} = [3,3,3,3\ldots]$$

Las hojas de cálculo

http://www.hojamat.es/sindecimales/aritmetica/herramientas/hoja/fraccont.xls

http://www.hojamat.es/sindecimales/aritmetica/herramientas/hoja/fraccont.ods

automatizan el proceso. Si las descargas recuerda que 3están compuestas de dos hojas, una para números fraccionarios y otra para racionales. Si en la primera escribes =RAIZ(3) obtendrás las aproximaciones (leyendo las reducidas) a la raíz cuadrada de 3.

El carácter aproximado de los cálculos produce que se rompan las posibles periodicidades después de muchos pasos.

ECUACIÓN DE PELL

Se llama ecuación de Pell (por error, porque Pell no la estudió) a la ecuación diofántica cuadrática $X^2 - DY^2 = 1$, con X e Y variables enteras y D número entero positivo no cuadrado perfecto. Existe una variante con el segundo miembro -1 que se resuelve de forma similar, con algunas restricciones, y también se consideran los casos en los que se trate de cualquier número entero.

En su resolución hay que distinguir dos problemas:

Primera solución

Una primera solución no es difícil de encontrar en general.

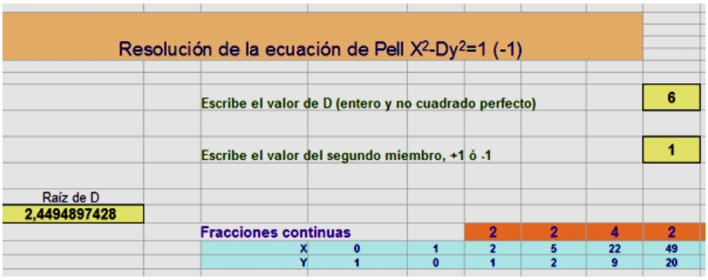
(a) Puedes acudir a un simple tanteo entre cuadrados perfectos. Por ejemplo, una solución de $X^2 - 6Y^2 = 1$ es $X_0=5$ $Y_0=2$. Con una hoja de cálculo no es tarea muy complicada.

(b) Las fracciones continuas también son útiles en la resolución de esta ecuación. Basta para ello desarrollar la raíz cuadrada de D mediante ellas y, según vimos en un apartado anterior, aprovechar la periodicidad del desarrollo. En el caso de la ecuación de Pell basta tomar las reducidas anteriores a la finalización del primer periodo.

2	2	4	2	4	2	4	2
2	5	22	49	218	485	2158	4801
1	2	9	20	89	198	881	1960
-2	1	-2	1	-2	1	-2	1
	Solución		Solución		Solución		Solución

En la imagen observarás que la solución $X_0=5$, $Y_0=2$ aparece antes del final del primer periodo [2,4] en el desarrollo por fracciones continuas. Después siguen otras: X=49, Y=20, X=485, Y=198, etc.

En nuestro modelo de hoja de cálculo que recomendamos más abajo basta escribir el valor de D y el segundo miembro +1 ó -1 y la hoja se encarga de desarrollar la raíz cuadrada de D mediante fracciones continuas:



Siguientes soluciones

Según la teoría del anillo Q(√D), que no podemos desarrollar aquí, las primeras soluciones, escritas como $X_0+Y_0\sqrt{D}$ constituyen una unidad del anillo, y también lo serán todas sus potencias, por lo que las siguientes soluciones provendrán de los desarrollos de las expresiones

$(X_0+Y_0\sqrt{D})^n$

agrupando después los términos que no contienen el radical como valor de Y y los que sí lo contienen como valor de X. Este método puede ser fatigoso, por lo que es mejor ir obteniendo las distintas soluciones por recurrencia. En efecto, de la anterior consideración se deduce que

$X_n+Y_n\sqrt{D} = (X_{n-1}+Y_{n-1}\sqrt{D})\ (X_0+Y_0\sqrt{D})$

O bien, separando términos:

$X_n = X_{n-1}X_0+Y_{n-1}Y_0D$

$Y_n = X_{n-1}Y_0+Y_{n-1}X_0$

Estas son las fórmulas que hemos usado en la hoja de cálculo.

Puedes consultar la búsqueda de la primera solución por fracciones continuas y la recurrencia para las siguientes en las hojas de cálculo

http://www.hojamat.es/sindecimales/aritmetica/herramientas/hoja/pell.ods

http://www.hojamat.es/sindecimales/aritmetica/herramientas/hoja/pell.xls

Por ejemplo, intenta resolver esta cuestión: ¿Qué cuadrado perfecto de diez cifras, al quitarle una unidad se puede descomponer en cinco cuadrados perfectos idénticos?

TÉCNICAS Y ALGORITMOS

PROLONGACIÓN DE UNA RECURRENCIA

En la confección de sucesiones, que es una de las tareas más frecuentes en mi blog, aparecen con cierta frecuencia algunas de las que se sabe o sospecha que pueden generarse mediante una fórmula de recurrencia respecto a sus primeros términos. Así ocurre, por ejemplo, con los números poligonales, que ocupan una buena parte de nuestros estudios, o con aquellas cuestiones que se resuelven con la ecuación de Pell o similares (ecuaciones Pell-like).

Las ecuaciones de recurrencia más frecuentes en estos temas son las lineales, en las que existe una relación de este tipo entre un elemento y varios de sus anteriores. Las llamaremos homogéneas si no intervienen términos independientes. Comenzaremos por ellas.

Sistema de ecuaciones de una recurrencia

Si cada elemento depende de los anteriores, pongamos por ejemplo, de cuatro, y de forma lineal, se dará la siguiente situación:

$a(n)=c_1a(n-1)+c_2a(n-2)+c_3a(n-3)+c_4a(n-4)$

Si elegimos los ocho primeros términos de la sucesión podremos plantear (en el caso homogéneo)

$a(8)=c_1a(7)+c_2a(6)+c_3a(5)+c_4a(4)$

$a(7)=c_1a(6)+c_2a(5)+c_3a(4)+c_4a(3)$

$a(6)=c_1a(5)+c_2a(4)+c_3a(3)+c_4a(2)$

$a(5)=c_1a(4)+c_2a(3)+c_3a(2)+c_4a(1)$

Esto constituye un sistema de cuatro ecuaciones con cuatro incógnitas c_1, c_2, c_3, c_4, que, al resolverse, nos descubre la ecuación de recurrencia. Con los instrumentos de cálculo disponibles en la actualidad es una tarea fácil de sobrellevar.

Pongamos un ejemplo. Los números hexagonales se generan con una ecuación de recurrencia de orden 3. Para encontrarla, ya lo habrás descubierto, necesitamos el doble de elementos, en este caso 6. Buscamos cualquier listado de ellos y seleccionamos 1 , 6 , 15 , 28 , 45 , 66. Por comodidad, llamamos a los coeficientes A, B, C, y queda

66=45A+28B+15C

45=28A+15B+6C

28=15A+6B+C

Resolvemos el sistema y obtenemos A=3, B=-3, C=1, luego los números hexagonales se generan mediante H(n)=3H(n-1)-3H(n-2)+H(n-3). Puedes comprobarlo en la entrada correspondiente en mi blog

(https://hojaynumeros.blogspot.com/2021/02/numeros-hexagonales-1.html)

Automatización del proceso

Desde hace años ofrezco en mi página web una calculadora matricial para Excel y Calc

(http://www.hojamat.es/sindecimales/aritmetica/herramientas/herrarit.htm#matrices)

Esta herramienta es de propósito general, con varias opciones y posibilidad de programar operaciones. Para no confundir con excesivo material, la he adaptado al problema que nos ocupa, y he situado esa versión en la carpeta propia de mi blog.

http://www.hojamat.es/blog/ecurrecurre.xlsm

En el caso de los hexagonales marcamos como orden 3, y escribimos en la fila correspondiente los primeros términos 1 , 6 , 15 , 28 , 45 , 66.

		Orden	3	Homogénea	No homogénea	
Sucesión	1	6	15	28	45	66
1,703125	-2,34375	0,890625				1
-2,34375	2,8125	-0,96875				-3

Después pulsamos en el botón “Homogénea” y se construirá el sistema de ecuaciones correspondiente:

		Orden	3	Homogénea	No homogénea	
Sucesión	1	6	15	28	45	66
1	6	15				28
6	15	28				45
15	28	45				66

Finalmente, pulsamos el botón “Resolver” y obtendremos los coeficientes:

No homogénea	Resolver	
45	66	91
	1	
	-3	
	3	

Como es una adaptación de otra herramienta, **se aconseja no tocar nada más de la hoja**. Si todo se viene abajo, volveremos a iniciar Excel.

Caso no homogéneo

Hay recurrencias lineales que poseen un término independiente. Estos mismos números hexagonales del ejemplo admiten otra recursión de tercer orden con término independiente 4.

$a(3)=K+c_1a(1)+c_2a(2)$

$a(4)=K+c_1a(2)+c_2a(3)$

$a(5)=K+c_1a(3)+c_2a(4)$

Resolvemos y nos resultan los coeficientes como en el caso homogéneo.

En nuestra hoja de cálculo basta pulsar sobre el botón “No homogéneo” y después sobre “Resolver”. En el caso de los hexagonales:

		Orden	3	Homogénea	No homogénea	
Sucesión	1	6	15	28	45	66
1	1	6				15
1	6	15				28
1	15	28				45

No se debe olvidar rellenar el Orden, en este caso, 3. La solución, después de resolver, queda:

4
-1
2

La interpretamos como a(n)=2a(n-1)-a(n-2)+4. En efecto:

15=2*6-1+4

28=2*15-6+4

45=2*28-15+4

Otros ejemplos

La recurrencia homogénea que hemos descubierto para los números hexagonales es una propiedad general de todos los poligonales, en los que P(n)=3P(n-1)-P(n-2)+P(n-3). Lo vemos en los octogonales: 1, 8, 21, 40, 65, 96, 133, 176, 225, 280, 341, 408, 481, 560, 645, 736, 833, 936,…

Aquí se inserta la captura de pantalla en la que comprobamos que los coeficientes:

		Orden	3	Homogénea	No homogénea	
Sucesión	1	8	21	40	65	96
1,087963	-1,481481	0,560185				1
-1,481481	1,740741	-0,592593				-3
0,560185	-0,592593	0,199074				3

Puedes probar con otros tipos de poligonales, como estos cuadrados centrados, 1, 5, 13, 25, 41, 61, 85, 113, 145,…y te resultarán los mismos coeficientes 3, -3 y 1.

Triangulares cuadrados

Los triangulares que también son cuadrados (los hemos estudiado en https://hojaynumeros.blogspot.com/2015/10/damos-vueltas-los-triangulares.html)

también admiten una recurrencia homogénea de tercer orden. Tomamos su listado y lo volcamos en nuestra hoja de cálculo: 1, 36, 1225, 41616, 1413721, 48024900, 1631432881, 55420693056, 1882672131025, 63955431761796,

2172602007770041, …y resulta:

		Orden	3	Homogénea	No homogénea	
Sucesión	1	36	1225	41616	1413721	48024900
1155,986	-1189,5	34,01389				1
-1189,5	1207	-34,5				-35
34,01389	-34,5	0,986111				35

Efectivamente, TC(n)=35TC(n-1)-35TC(n-2)+TC(n-3), tal como hemos comprobado con los siguientes términos.

Así podríamos recorrer más ejemplos. Como esto es una presentación de una herramienta, con lo explicado basta.

DETECCIÓN DE PROGRESIONES ARITMÉTICAS

Es bastante frecuente que tres elementos de una misma sucesión o del mismo tipo presenten una progresión aritmética, es decir, que tengan la forma (n-k, n, n+k). Un ejemplo de esto son las ternas de números primos en progresión:

3 5 7
3 7 11
5 11 17
7 13 19
11 17 23
7 19 31
17 23 29
17 29 41
19 31 43
31 37 43
29 41 53

En mis sucesiones http://oeis.org/A292313, http://oeis.org/A292309, http://oeis.org/A292314 y otras se hace referencia a este tipo de ternas, que son admitidas por casi todos los tipos de números más populares.

La detección de estas ternas no ofrece problemas de diseño de un algoritmo. Si fijamos la atención en el término central **n**, es claro que **n-k** pertenecerá al rango **(1, n-1)**. Por tanto, bastará recorrerlo y, para cada número del mismo tipo que n, buscar si **n+k** también coincide con ellos en la misma característica o propiedad, que en el caso anterior era la de ser primos.

Las ternas anteriores de primos se pueden detectar con la siguiente función de Excel y Calc:

function hayprog$(n) ‘Devuelve un string con la terna, o bien una cadena vacía
dim i,k
dim s$

if not esprimo(n) then hayprog="":exit function ‘Si el número no es primo, salimos.
s=""
k=0
for i=1 to n-1 ‘La variable i recorre el rango (1,n)
if esprimo(i) then ‘Si es primo, seguimos
k=n-i ‘Si n+k también es primo, hemos llegado a una terna
if esprimo(n+k) then s=s+” # “+str$(n-k)+" "+str$(n)+" "+str$(n+k)
end if
next i

hayprog=s
end function

Con PARI se puede usar este mismo algoritmo. Al ser un lenguaje más orientado a números, usaremos como respuesta de la función ESPROG el término central.

esprog(n)={my(s=0,k=0,i);if(isprime(n),for(i=1,n-1,if(isprime(i),k=n-i;if(isprime(n+k),s=1))));s}

for(i=1,100,if(esprog(i),print(i)))

Si lo ejecutas en https://pari.math.u-bordeaux.fr/gp.html, te devolverá los términos primos centrales:

```
5
7
11
13
17
19
23
29
31
37
41
```

El uso del bucle FOR-NEXT se justifica porque, en otros casos que veremos más adelante, pueden existir dos soluciones. Si no, usaríamos WHILE, que se puede detener antes.

Ternas de cuadrados

Las ideas anteriores las podemos aplicar a muchos tipos de números. Comenzamos por los cuadrados. Para ello, sustituimos la función ESPRIMO por ESCUAD en Excel y Calc, y en PARI, i*sprime* por *issquare*. Puedes buscar la función ESCUAD en mi blog. El resultado sería el siguiente:

1 25 49
4 100 196
49 169 289
9 225 441
49 289 529
16 400 784
25 625 1225 # 289 625 961
196 676 1156
1 841 1681
36 900 1764
196 1156 2116
49 1225 2401
529 1369 2209
441 1521 2601
64 1600 3136
961 1681 2401

Vemos que el 625 es centro de dos ternas distintas. Comprueba que todas las ternas presentan una progresión aritmética.

Los términos centrales están publicados en OEIS:

A198385 Second of a triple of squares in arithmetic progression.

25, 100, 169, 225, 289, 400, 625, 676, 625, 841, 900, 1156, 1369, 1225, 1681, 1521, 1600, 2500, 2025, 2704, 2601, 2500, 3721, 2809, 3025, 4225, 3364, 3600, 4225, 4225, 4225, 4624, 5625, 5476, 7225, 4900, 6724, 6084, 5329, 5625, 6400, 7225, 7225, 7225, 7921

Las primeras diferencias de estas progresiones son: 24, 96, 120, 216, 240, 384, 336, 480, 840, 864, 960, 1176 y 840, y todas son múltiplos de 24.

Las sumas de los tres componentes de la terna las publiqué en http://oeis.org/A292313. Coinciden, como es evidente con el triple del término central.

A292313 Numbers that are the sum of three squares in arithmetic progression.

75, 300, 507, 675, 867, 1200, 1875, 2028, 2523, 2700, 3468, 3675, 4107, 4563, 4800, 5043, 6075, 7500, 7803, 8112, 8427, 9075, 10092, 10800, 11163, 12675, 13872, 14700, 15987, 16428, 16875, 18252, 19200, 20172, 21675, 22707, 23763, 24300, 24843, 27075, 28227, 30000, 30603

Relación entre bases

Una búsqueda se debe enriquecer siempre con ideas teóricas o algebraicas. En este caso, no es difícil relacionar las bases.

Podemos representar la terna como $(n-r)^2$, n^2 y $(n+s)^2$. Con ello, la igualdad de diferencias será:

$$n^2-(n-r)^2=(n+s)^2-n^2$$

Simplificando:

$$s^2+2ns+r^2-2nr=0$$

Es una ecuación de segundo grado en la variable **s** en la que el discriminante es n^2-r^2+2nr y ha de ser cuadrado. Esto nos da una posibilidad de crear otra función en la que los resultados sean las bases de los cuadrados. Puede ser esta:

```
function hayprog2$(n)
dim i,k,q
dim s$

s=""
k=0
for i=1 to n-1
```

q=n^2-i^2+2*n*i ‘Se exige que el discriminante sea cuadrado
if escuad(q) then
k=sqr(q)-n ‘Las diferencias serán i y k
s=s+” # “+str$(n-i)+" "+str$(n)+" "+str$(n+k)
end if
next i
hayprog2=s
end function

Efectivamente, los números que producen resultados válidos en esta función son las raíces cuadradas de los que obtuvimos más arriba,

1 5 7
2 10 14
7 13 17
3 15 21
7 17 23
4 20 28
17 25 31 # 5 25 35
14 26 34
1 29 41
6 30 42
14 34 46
7 35 49
23 37 47

21 39 51
8 40 56

Protagonismo para las diferencias

Podemos despejar **n** en la expresión $s^2+2ns+r^2-2nr=0$:

n=(s²+r²)/(2(r-s))

Si **n** es entero, el par **r, s** será válido para cumplir la condición de progresión. Este punto de vista ha resultado muy eficaz, porque para cada valor de **r** (que será par por ser igual a una diferencia de cuadrados y mayor que s) se obtienen varios valores posibles de n. Son los mismos términos centrales en las ternas de bases, pero agrupados por conjuntos. Hemos usado la función:

```
function hayprog3$(n)
dim i,q
dim s$
s=""
for i=1 to n-1
q=(n^2+i^2)/(2*(n-i)) 'La variable q representa la base central
if q=int(q) then s=s+" "+str$(q) 'Se añaden todas las soluciones para una diferencia dada
next i
hayprog3=s
end function
```

Si se organiza una búsqueda con esta función, obtendremos los valores de la base central para cada diferencia:

6	5 13
8	10 25
10	17 41
12	10 15 26 61
14	37 85
16	20 50 113
18	15 39 65 145
20	13 25 34 82 181
22	101 221
24	17 20 30 52 75 122 265
26	145 313
28	29 35 74 170 365
30	25 29 51 65 123 197 421
32	40 100 226 481

Equivalencia entre dos sumas de impares

Si recordamos que un cuadrado es una suma de los primeros impares, la diferencia entre dos de ellos también será una suma, aunque no comenzará con el 1. Entonces, las ternas que estamos investigando se traducirán en equivalencia entre dos sumas de impares consecutivos. Así, la terna 1, 25, 49 se puede representar como

3+5+7+9=11+13

Ternas de triangulares en progresión

Si sustituimos la función ESCUAD por ESTRIANGULAR (está publicada en mi blog, y basta exigir que 8n+1 sea cuadrado), obtendremos ternas del mismo tipo, pero para triangulares. Aquí tenemos una pequeña dificultad si consultamos OEIS, y es que se considera el 0 como triangular, siguiendo la política de esa página. Así, en el siguiente listado publicado, no entrarían, por ejemplo, 3 y 105 si no admitiéramos 0 como triangular. En los cuadrados no existe ese problema, porque uno de ellos no puede ser el doble de otro, pero en los triangulares sí (ver http://oeis.org/A075528).

3 # 0 3 6

21 # 6 21 36

28 # 1 28 55

36 # 6 36 66

78 # 3 78 153 # 36 78 120

105 # 0 105 210

153 # 6 153 300

171 # 66 171 276

190 # 55 190 325

210 # 120 210 300

253 # 10 253 496

325 # 55 325 595
351 # 36 351 6
378 # 15 378 741

A292310

Triangular numbers that are equidistant from two other triangular numbers.

3, 21, 28, 36, 78, 105, 153, 171, 190, 210, 253, 325, 351, 378, 465, 528, 666, 703, 903, 946, 990, 1035, 1128, 1176, 1275, 1378, 1485, 1540, 1596, 1653, 1711, 1770, 1891, 1953, 2278, 2346, 2556, 2628, 2775, 2926, 3003, 3081, 3160, 3403, 3570, 3741, 3828, 4095, 4186, 4278, 4371, 4656

Relación entre órdenes

Repitiendo los pasos de los cuadrados, tendremos:

$\mathbf{(n+s)(n+s+1)+(n-r)(n-r+1)=2n(n+1)}$

$\mathbf{ns+n+ns+s^2+s-rn+n-rn+r^2-r=2n}$

$\mathbf{2ns+s^2+s-2rn+r^2-r=0}$

$\mathbf{s^2+(2n+1)s-(2n+1)r+r^2=0}$

El discriminante para s en esa ecuación será

$$(2n+1)^2+4(2n+1)r-4r^2$$

Si es un cuadrado se tendrá:

$$q^2=(2n+1+2r)^2-8r^2$$

$$s=(q-2n-1)/2$$

Por tanto, en *hayprog2* habrá que introducir este nuevo discriminante:

```
function hayprog2$(n)
dim i,k,q
dim s$

s=""
k=0
for i=0 to n-1
q=(2*n+1+2*i)^2-8*i^2
if escuad(q) then
k=(sqr(q)-2*n-1)/2
if k>0 then s=s+" # "+str$(n-i)+" "+str$(n)+" "+str$(n+k)
end if
next i
hayprog2=s
end function
```

El resultado de los primeros órdenes de los triángulos es:

6 # 3 6 8
7 # 1 7 10
8 # 3 8 11
12 # 8 12 15 # 2 12 17
17 # 3 17 24
18 # 11 18 23
19 # 10 19 25
20 # 15 20 24
22 # 4 22 31
25 # 10 25 34
26 # 8 26 36
27 # 5 27 38
30 # 24 30 35
32 .. # 23 32 39 # 17 32 42 # 11 32 44 # 6 32 45
36 # 3 36 51

Aparecen varios casos múltiples para un mismo orden central. Sin tenerlos en cuenta, con ellos se reconstruyen las ternas de más arriba:

N-r	n	n+s	T1	T2	T3
3	6	8	6	21	36
1	7	10	1	28	55
3	8	11	6	36	66
8	12	15	36	78	120
3	17	24	6	153	300
11	18	23	66	171	276
10	19	25	55	190	325
15	20	24	120	210	300
4	22	31	10	253	496
10	25	34	55	325	595
8	26	36	36	351	666
5	27	38	15	378	741
24	30	35	300	465	630
23	32	39	276	528	780

Equivalencia entre sumas de consecutivos

Al igual que nos ocurrió con los cuadrados, al ser los triangulares suma de números consecutivos, cada progresión aritmética de triangulares se traducirá en una equivalencia de sumas de ese tipo. Por ejemplo, la terna 6, 36, 66 se traduce en 4+5+6+7+8=9+10+11, ya que ambos suman 30, que es la diferencia.

Ternas de oblongos

Cada terna de triangulares se convierte en otra de oblongos, ya que estos son el doble de los primeros. Cada triangular n(n+1)/2 dará lugar al oblongo n(n+1). No tiene mayor interés.

Semiprimos

Todos los semiprimos, salvo 4 y 6, pueden ser términos centrales en ternas de semiprimos en progresión aritmética. Hay estudios sobre esto en Xianmeng Meng, On sums of three integers with a fixed number of prime factors
https://www.sciencedirect.com/science/article/pii/S0022314X0500106X?via%3Dihub

Aquí figuran las ternas que admiten los primeros semiprimos:

9 1 : # 4, 9, 14
10 1 : # 6, 10, 14
14 1 : # 6, 14, 22
15 2 : # 4, 15, 26 # 9, 15, 21
21 2 : # 4, 21, 38 # 9, 21, 33
22 3 : # 6, 22, 38 # 9, 22, 35 # 10, 22, 34
25 2 : # 4, 25, 46 # 15, 25, 35
26 2 : # 6, 26, 46 # 14, 26, 38
33 3 : # 4, 33, 62 # 9, 33, 57 # 15, 33, 51
34 4 : # 6, 34, 62 # 10, 34, 58 # 22, 34, 46 # 33, 34, 35
35 2 : # 15, 35, 55 # 21, 35, 49
38 3 : # 14, 38, 62 # 21, 38, 55 # 25, 38, 51

39 3 : # 4, 39, 74 # 9, 39, 69 # 21, 39, 57
46 5 : # 6, 46, 86 # 10, 46, 82 # 15, 46, 77 # 34, 46, 58 # 35, 46, 57
49 3 : # 4, 49, 94 # 21, 49, 77 # 33, 49, 65

Vemos que faltan 4 y 6, y que el resto admite una o más ternas de las pedidas. Por ejemplo, 46 admite 5 soluciones.

Podemos modificar las funciones para que solo nos ofrezcan el número de soluciones. Si exigimos dos, además del 4 y 6, hay otros semiprimos que no admiten soluciones dobles. En el listado se identifican 9, 10 y 14, y no hemos encontrado más. Los siguientes admiten al menos dos soluciones.

Si exigimos que al menos presenten tres soluciones, las excepciones se extienden a esta lista:

4, 6, 9, 10, 14, 15, 21, 25, 26, 35 y 57

Para cuatro sigue el tope de 57 (sólo hemos investigado hasta el 500), pero se incorporan 22, 33, 38, 39, y 49. Con exigencia de cinco llegaríamos a 87. Lo dejamos ahí sin formular ninguna conjetura. El autor, a punto de cumplir 80 años, no se ve con ánimo para estudiar documentos como el reseñado de Meng.

Con estas consideraciones, se pueden emprender otras búsquedas, que dejamos a quienes nos siguen.

ALTERNATIVA A FAULHABER

En muchas ocasiones puede interesar sumar las primeras potencias de los números naturales. Están publicados todos los casos populares, como sumas de cuadrados o de cubos, y existe una fórmula, atribuida a Faulhaber, que nos da el resultado para cualquier exponente. En el siguiente recorte de Wikipedia puedes estudiarla.

> En Matemáticas, la **fórmula de Faulhaber**, en honor de Johann Faulhaber, expresa la suma de las potencias de los primeros *n* números naturales
>
> $$\sum_{k=1}^{n} k^p = 1^p + 2^p + 3^p + \cdots + n^p$$
>
> como un polinomio en *n* de grado $(p+1)$ cuyos coeficientes se construyen a partir de los números de Bernoulli: B_j.
>
> La fórmula es la siguiente:
>
> $$\sum_{k=1}^{n} k^p = \frac{1}{p+1} \sum_{j=0}^{p} \binom{p+1}{j} B_j n^{p+1-j} \qquad \left(\text{con } B_1 = +\frac{1}{2} \text{ en vez de } -\frac{1}{2}\right)$$
>
> Faulhaber no conoció nunca esta fórmula general; lo que sí conoció fueron al menos los primeros 17 casos y el hecho de que, si el exponente es impar, entonces la suma es una función polinomial de la suma en el caso especial en el que el exponente sea 1. También hizo algunas generalizaciones (véase Knuth).

https://es.wikipedia.org/wiki/F%C3%B3rmula_de_Faulhaber

El problema que tiene esta fórmula para un uso elemental es que requiere conocer los números de Bernouilli.

El procedimiento que explicaremos a continuación es una alternativa para encontrar el valor de la suma de una sucesión de potencias o expresiones polinómicas con Excel o Calc. Se debe tomar como un simple entretenimiento, aunque en algunas situaciones puede resultar útil.

Un ejemplo: Cuadrados de oblongos:

Explicamos el procedimiento con un ejemplo, como sería encontrar una fórmula para la suma de los cuadrados de los primeros oblongos, es decir de la expresión $n^2(n+1)^2$.

A) Creamos la sucesión:

Con las hojas de cálculo es muy fácil encontrar las primeras sumas de cualquier sucesión. En este caso hemos ido creando columnas para N(N+1), que son los oblongos, sus cuadrados $N^2(N+1)^2$ y sus sumas sucesivas.

N	N(N+1)	N^2(N+1)^2	SUMAS
1	2	4	4
2	6	36	40
3	12	144	184
4	20	400	584
5	30	900	1484
6	42	1764	3248
7	56	3136	6384
8	72	5184	11568
9	90	8100	19668
10	110	12100	31768
11	132	17424	49192
12	156	24336	73528
13	182	33124	106652

Con esto ya sabemos que la sucesión 4, 40, 184, 584, 1484, 3248, 6384,…es la que requiere una fórmula similar a las de Faulhaber. Para continuar debemos basarnos en dos conjeturas:

1) La fórmula buscada creemos que será de tipo polinómico.

2) Intuiremos de alguna forma qué grado puede tener ese polinomio. Esta segunda no es tan importante, pero nos ayudará en el siguiente paso.

B) Aplicamos la interpolación de Newton:

La búsqueda de una fórmula polinómica que resuma un conjunto de valores es una interpolación. Disponemos de una hoja de cálculo que encuentra esa fórmula para los valores 1, 2, 3, 4, …mediante la interpolación de Newton:

http://www.hojamat.es/sindecimales/aritmetica/herramientas/herrarit.htm#newton

(Ver https://es.wikipedia.org/wiki/Interpolaci%C3%B3n_polin%C3%B3mica_de_Newton)

Este método se adapta bien a la estructura de Excel y Calc. En nuestra hoja basta escribir los primeros términos y observar las diferencias que se producen. Conviene usar todos los que entren en el esquema (máximo 7). Si se conoce el grado del polinomio, se pueden usar menos. Esta sería la situación para el caso de cuadrados de oblongos:

Valor natural inicial	1	2	3	4	5	6	7
Valores función	4	40	184	584	1484	3248	6384
Dif1		36	144	400	900	1764	3136
Dif2			108	256	500	864	1372
Dif3				148	244	364	508
Dif4					96	120	144
Dif5						24	24
Dif6							0
Coeficientes (en forma de fracción)	0	24	96	148	108	36	4
	720	120	24	6	2	1	1

Observamos que la diferencia sexta ya es 0, con lo que el grado del polinomio será cinco, como se ve en los coeficientes de abajo, que son seis.

Estos coeficientes actúan sobre los polinomios 1, (x-1), (x-1)(x-2), (x-1)(x-2)(x-3),…y esa es la mayor dificultad de esta interpolación, porque el resultado en este caso sería

4+36*(x-1)+54*(x-1)*(x-2)+74/3*(x-1)*(x-2)*(x-3)+4*(x-1)*(x-2)*(x-3)*(x-4)+1/5*(x-1)*(x-2)*(x-3)*(x-4)*(x-5)

(Ya se han simplificado los coeficientes)

C) Desarrollamos el polinomio interpolador:

Este resultado podría ser descorazonador, pero para su simplificación contamos con los programas CAS ((Computer Algebra System) En mi blog se suele usar la Calculadora Wiris, gratuita y extendida en la enseñanza.

https://calcme.com/a

Copiamos nuestra monstruosa fórmula en ella y pulsamos sobre el signo =

$$4+36*(x-1)+54*(x-1)*(x-2)+74/3*(x-1)*(x-2)*(x-3)+4*(x-1)*(x-2)*(x-3)*(x-4)+1/5*(x-1)*(x-2)*(x-3)*(x-4)*(x-5) =$$

$$\frac{1}{5}\cdot x^5+x^4+\frac{5}{3}\cdot x^3+x^2+\frac{2}{15}\cdot x$$ Calc

Ya hemos conseguido el objetivo: la suma de los cuadrados de los oblongos sigue la fórmula $x^5/5+x^4+5/3x^3+x^2+2/15x$

Sustituimos el polinomio en la tabla para comprobar

N	N(N+1)	N^2(N+1)^2	SUMAS	Polinomio
1	2	4	4	4
2	6	36	40	40
3	12	144	184	184
4	20	400	584	584
5	30	900	1484	1484
6	42	1764	3248	3248
7	56	3136	6384	6384
8	72	5184	11568	11568
9	90	8100	19668	19668
10	110	12100	31768	31768
11	132	17424	49192	49192
12	156	24336	73528	73528
13	182	33124	106652	106652

Luego la fórmula queda:

$$S(n) = \frac{n^5}{5} + n^4 + \frac{5n^3}{3} + n^2 + \frac{2n}{15}$$

La podemos factorizar con Wiris:

$$\frac{1}{5} \cdot n \cdot (n+1) \cdot (n+2) \cdot \left(n + \frac{\sqrt{6}}{3} + 1\right) \cdot \left(n - \frac{\sqrt{6}}{3} + 1\right) \quad \text{Calc}$$

Estas son las etapas del proceso. Nos basamos en Excel y Calc, en nuestra hoja de interpolación y en un CAS. En la mayoría de los casos obtendremos el polinomio adecuado. Puede que el grado requerido sea mayor, con lo que habría que ampliar el esquema de cálculo, pero ese trabajo es algo complejo.

Repetimos el trabajo con oblongos, Lo dejamos con redacción escueta:

Sumas de oblongos

N	N(N+1)	SUMAS
1	2	2
2	6	8
3	12	20
4	20	40
5	30	70
6	42	112
7	56	168
8	72	240
9	90	330
10	110	440
11	132	572
12	156	728
13	182	910

Debemos buscar una fórmula para 2, 8, 20, 40, 70, 112, …

Interpolamos

Valor natural inicial	1	2	3	4	5	6	7
Valores función	2	8	20	40	70	112	
Dif1		6	12	20	30	42	
Dif2			6	8	10	12	
Dif3				2	2	2	
Dif4					0	0	
Dif5						0	
Dif6							
ntes (en fracción)		0	0	2	6	6	2
	720	120	24	6	2	1	1

Observamos que son nulas las diferencias a partir de la cuarta, luego obtendremos un polinomio de tercer grado. Sería este

2+6*(x-1)+3*(x-1)*(x-2)+1/3*(x-1)*(x-2)*(x-3)

Con wiris

$$\frac{1}{3}\cdot x^3+x^2+\frac{2}{3}\cdot x \quad \text{Calc}$$

Factorizando con la misma calculadora:

$$\text{factorizar}\left(\frac{x^3}{3}+x^2+\frac{2\cdot x}{3}\right) \quad = \quad \frac{1}{3}\cdot x\cdot(x+1)\cdot(x+2) \quad \text{Calc}$$

Luego es el doble del combinatorio C(x+2,3), como puede verse en http://oeis.org/A007290

En este caso nos hemos limitado a comprobar, porque esta suma ya está resuelta.

Como un ejemplo del uso de esta fórmula puedes distraerte con mi entrada

http://hojaynumeros.blogspot.com/2018/09/suma-de-numeros-oblongos-consecutivos.html

Suma de potencias cuartas

Por último, reproduciremos una de las fórmulas más conocidas de Faulhaber, la que suma potencias cuartas.

$$1^4 + 2^4 + 3^4 + \cdots + n^4 = \frac{6n^5 + 15n^4 + 10n^3 - n}{30}$$

(Fuente: Wikipedia)

Seguimos los pasos sugeridos.

Construimos la sucesión:

N	N^4	SUMAS
1	1	1
2	16	17
3	81	98
4	256	354
5	625	979
6	1296	2275
7	2401	4676
8	4096	8772
9	6561	15333
10	10000	25333
11	14641	39974
12	20736	60710
13	28561	89271

Como sabemos que el grado de la fórmula de Faulhaber es 5, interpolaremos con al menos seis elementos.

Interpolación

Valor natural	1	2	3	4	5	6	7
Valores función	1	17	98	354	979	2275	4676
Dif1		16	81	256	625	1296	2401
Dif2			65	175	369	671	1105
Dif3				110	194	302	434
Dif4					84	108	132
Dif5						24	24
Dif6							0
ientes (en	0	24	84	110	65	16	1
e fracción)	720	120	24	6	2	1	1

A partir de los coeficientes de abajo construimos el polinomio:

1+16*(x-1)+65/2*(x-1)*(x-2)+55/3*(x-1)*(x-2)*(x-3)+7/2*(x-1)*(x-2)*(x-3)*(x-4)+1/5*(x-1)*(x-2)*(x-3)*(x-4)*(x-5)

Simplificamos con Wiris

$$\frac{1}{5} \cdot x^5 + \frac{1}{2} \cdot x^4 + \frac{1}{3} \cdot x^3 - \frac{1}{30} \cdot x \quad \text{Calc}$$

Si reducimos todo a denominador 30, coincidirá con la fórmula de Faulhaber correspondiente.

ALGORITMOS CODICIOSOS PARA SUMAS

Este tipo de algoritmo recibe muchos nombres distintos: voraz, goloso, devorador, codicioso, greedy, …Se basa en la estrategia de elegir en cada paso la solución óptima del problema, esperando que, al reiterar esta estrategia, se construya una solución óptima global. Para conseguirlo, no da marcha atrás en las decisiones si no se llega a un óptimo. Esto lo hace más rápido, pero más inseguro, porque no hay garantía de consecución de objetivos.

Ha de ser posible la elección óptima en cada paso, como en el caso de descomposición en suma de factoriales, que siempre se puede conocer el mayor factorial menor o igual que el número dado.

Aquí estudiaremos varios ejemplos de descomposición de un número entero positivo en sumandos de un cierto tipo. Un ejemplo claro es el de encontrar una suma de factoriales cuyo resultado sea un número dado. Este algoritmo lo llevamos usando años en nuestras publicaciones, porque es rápido, aunque en ocasiones produce soluciones con un número alto de sumandos.

Lo vemos con un ejemplo elegido al azar, el número 1329, al que queremos convertir en una suma de factoriales lo menos extensa posible.

Es claro que si no deseamos muchos sumandos, vayamos eligiendo los factoriales mayores posibles para construir la suma. Podríamos efectuarlo así:

1329-720=609
609-120=489
489-120=369
369-120=249
249-120=129
129-120=9
9-6=3

3-2=1
1-1=0

Con ello, hemos descompuesto 1329 en suma de factoriales:

1329=720+120+120+120+120+120+6+2+1

Basta analizar la solución para darnos cuenta de que no podemos esperar demasiado de este algoritmo, pero al contar con el 1 entre los factoriales, siempre dará solución.

Un caso similar es el de descomponer una fracción en fracciones egipcias (del tipo 1/p). Bastará ir buscando en el numerador el mayor divisor posible del denominador (suponemos fracción irreducible). Por ejemplo, la fracción 17/24 se puede descomponer de esta forma, eligiendo en el numerador sucesivamente los mayores divisores de 24:

17/24=(12+4+1)/24=1/2+1/6+1/24

Los dos ejemplos los hemos desarrollado de forma manual, pero es claro que se podrá automatizar fácilmente.

Para planificar los algoritmos, distinguiremos dos variantes:

La primera requerirá que los sumandos pertenezcan a un mismo tipo, como los factoriales de más arriba, o, por ejemplo el problema de descomponer un número en sumandos triangulares.

La segunda, muy parecida a la primera, deberá elegir los sumandos dentro de una lista, como en el problema de las monedas, en el que una cantidad hay que traducirla a un conjunto de monedas con distintos valores.

En realidad, el ejemplo de los factoriales se puede considerar de este segundo tipo, por su escasez.

Sumandos del mismo tipo

Si el sumando óptimo lo debemos elegir dentro de un tipo, como primos, cuadrados o pentagonales, podremos usar un sencillo algoritmo que el autor usa con frecuencia. Tendría esta estructura, en la que, para un número **n** y un tipo dado, devuelve la suma más adecuada que un algoritmo codicioso puede dar

Function codicioso$(n)
Dim c$
Dim i, r, s,suma
Dim vale As Boolean

suma = 0 'Irá sumando progresivamente
i = n ‘Recorrerá los posibles sumandos
r = n ‘Posible sumando válido
c = "" ’Recogerá soluciones
While i > 0 ‘Recorremos de mayor a menor, para optimizar
vale = False ‘Aún no hay solución
if CONDICION then vale=true ‘La CONDICIÓN dependerá del tipo de sumandos
If vale Then
c$ = c$ + Str$(i) + " "
suma=suma+i
‘Si existe solución, se resta y se incorpora a la suma
r = r - i: i = r
Else
‘Si no existe solución, se sigue buscando el óptimo, descendiendo
i = i – 1
end if
Wend
If suma=n then codicioso = c else codicioso=”NO”
End Function

Por ejemplo, si la condición es la de ser primo, usaríamos la función ESPRIMO, fácilmente localizable en mi blog. En la tabla siguiente se recogen las sumas en números primos, si es que son posibles, para los primeros números de tres cifras:

No primos	Suma de primos
100	97 3
105	103 2
106	103 3
111	109 2
112	109 3
115	113 2
116	113 3
118	113 5
120	113 7
122	113 7 2
123	113 7 3
124	113 11
126	113 13
129	127 2
130	127 3
133	131 2
134	131 3
136	131 5

Este ejemplo es muy interesante, porque ilustra las limitaciones de los algoritmos codiciosos. En primer lugar, sólo figuran los números en los que, al finalizar el algoritmo, la suma coincide con ellos, sin dejar ningún residuo no primo. Por ejemplo, el 15 daría como resultado 7+7, y sobraría 1, porque no es primo. En segundo lugar, al no explorar todas las posibilidades, ignora las sumas de Golbach para números pares, porque se detiene antes de llegar a los dos primos que predice la conjetura. Por ejemplo, el número 104, que no figura en la tabla, presenta varias descomposiciones en sumas de dos primos, y el algoritmo codicioso las ignora:

X1	X2
3	101
7	97
31	73
37	67
43	61

No obstante, en sumados de algunos tipos, el algoritmo funciona relativamente bien. Por ejemplo, cuando el 1 pertenece a ese tipo, como ocurriría en los cuadrados, triangulares, palindrómicos o de Fibonacci, la descomposición siempre será exitosa, aunque, quizás, algo larga. También suele funcionar bien con primos y cuadrados de primos, y con oblongos si el número es par. El siguiente ejemplo contiene resultados de cuadrados de primos entre 375 y 400:

Número	Sumandos cuadrados de primos
378	361 9 4 4
379	361 9 9
383	361 9 9 4
386	361 25
390	361 25 4
394	361 25 4 4
395	361 25 9
399	361 25 9 4

En este caso están condicionados por el cuadrado de 19, 361. En general suelen aparecer los ejemplos con una cierta rapidez.

Un ejemplo clásico, y que publicamos a veces, es el de la representación de Zeckendorf, en la que todo número es suma de elementos de la sucesión de Fibonacci (ver, por ejemplo, nuestra entrada

https://hojaynumeros.blogspot.com/2020/09/representacion-de-zeckendorf.html).

En ella el algoritmo indicado es el codicioso. Hemos aplicado la función que presentamos más arriba a un conjunto de números consecutivos, y observamos que todos están representados así:

Número	Sumandos números de Fibonacci
25520	17711 6765 987 55 2
25521	17711 6765 987 55 3
25522	17711 6765 987 55 3 1
25523	17711 6765 987 55 5
25524	17711 6765 987 55 5 1
25525	17711 6765 987 55 5 2
25526	17711 6765 987 55 8
25527	17711 6765 987 55 8 1
25528	17711 6765 987 55 8 2
25529	17711 6765 987 55 8 3
25530	17711 6765 987 55 8 3 1

Puedes decubrir aspectos muy interesantes sobre esta descomposición en

https://en.wikipedia.org/wiki/Zeckendorf%27s_theorem

Sumandos dentro de una lista

Este otro tipo de descomposición en sumandos que se adapta bien a los algoritmos codiciosos, es el de expresar un número como suma de números ya fijados en una lista. De este tipo es el problema de las monedas, en el que hay que encontrar la forma de pagar una cantidad si se dispone de un número suficiente de monedas de varias clases.

En mi blog he tratado este tema y otros similares, como el de los números McNugget y la representación de un número respecto a una lista:

https://hojaynumeros.blogspot.com/2010/02/frobenius-y-los-mcnuggets.html

https://hojaynumeros.blogspot.com/2012/11/descomposicion-de-un-numero-segun-una.html

Por ejemplo, si nos piden representar el número 80 mediante los números de la lista 2, 4, 14, 32, procederíamos como en los problemas anteriores, restando de 80 el número mayor posible cada vez, y usando repeticiones:

48=80-32; 16=38-32; 16=14+2

Luego 80=32+32+14+2

No hemos necesitado el sumando 4.

El procedimiento usado para sumandos del mismo tipo es válido aquí, pero deberemos iniciar el algoritmo declarando los elementos de la lista según un vector.

Algo similar usa las herramientas que ofrecemos en hojamat.es:

http://www.hojamat.es/sindecimales/aritmetica/herramientas/herrarit.htm#reprenum

http://www.hojamat.es/sindecimales/aritmetica/herramientas/herrarit.htm#mcnugget

Aquí seguiremos usando una función como en el primer caso, pero declarando los sumandos. Los introduciremos como componentes de un vector. Por ejemplo, en el siguiente listado se pretende encontrar qué monedas y billetes de euro podemos usar en una compra ordinaria, sin céntimos, y con un tope del billete de 50€:

Function codicioso3$(n)
Dim c$
Dim i, j, r, s, suma, tope
Dim v(20) ‘Usamos un vector de 20 componentes

v(1) = 1: v(2) = 2: v(3) = 5: v(4) = 10: v(5) = 20: v(6) = 50
tope = 6 ‘Determinamos seis monedas y billetes

‘El resto del listado es muy similar al del anterior tipo
suma = 0 'Irá sumando progresivamente
i = n 'Recorrerá los posibles sumandos
r = n 'Posible sumando válido

c = "" 'Recogerá soluciones
j = tope
While i > 0 And j > 0 'Recorremos de mayor a menor, para optimizar
If i >= v(j) Then
c$ = c$ + Str$(v(j)) + " "'Aquí sumamos componentes del vector
suma = suma + v(j)
i = i - v(j)
Else
'Si no existe solución, se sigue buscando el óptimo, descendiendo
j = j - 1
End If
Wend
If suma = n Then codicioso3 = c Else codicioso3 = "NO"
End Function

Como ejemplo, insertamos una tabla con algunas cantidades en euros y su descomposición en monedas y billetes:

Cantidad	Monedas y billetes
37	20 10 5 2
56	50 5 1
82	50 20 10 2
107	50 50 5 2
176	50 50 50 20 5 1
228	50 50 50 50 20 5 2 1

Hay que volver a advertir que estas soluciones no han de ser ni únicas ni óptimas.

SUMAS DE POTENCIAS CONSECUTIVAS

Existen fórmulas para sumar las primeras potencias de números naturales. Son populares las de la suma de potencias con los primeros exponentes. En esta captura de Excel figuran algunas:

$$1+2+3+\cdots+n=\frac{n(n+1)}{2}=\frac{n^2+n}{2}$$

$$1^2+2^2+3^2+\cdots+n^2=\frac{n(n+1)(2n+1)}{6}=\frac{2n^3+3n^2+n}{6}$$

$$1^3+2^3+3^3+\cdots+n^3=\left(\frac{n^2+n}{2}\right)^2=\frac{n^4+2n^3+n^2}{4}$$

$$1^4+2^4+3^4+\cdots+n^4=\frac{6n^5+15n^4+10n^3-n}{30}$$

$$1^5+2^5+3^5+\cdots+n^5=\frac{2n^6+6n^5+5n^4-n^2}{12}$$

$$1^6+2^6+3^6+\cdots+n^6=\frac{6n^7+21n^6+21n^5-7n^3+n}{42}$$

Aquí deseamos usar potencias consecutivas, pero que no comiencen necesariamente por la unidad. Si se usa la fórmula correspondiente, el problema se resuelve restando. Por ejemplo, una suma de potencias entre $\mathbf{a}^{\mathbf{k}}$ y $\mathbf{b}^{\mathbf{k}}$ se encontraría restando la fórmula correspondiente a **b** y la de **a-1**, para que se incluya también **a**. No será ese el camino que se tome aquí, porque deseamos encontrar la suma con un algoritmo que sirva para todos los exponentes. No obstante, dejamos abierta la posibilidad de comprobar algún cálculo.

Nuestro objetivo es más ambicioso, y es encontrar una suma de potencias consecutivas que sea equivalente a otra potencia dada, como en los ejemplos que siguen:

47^3=22^2+23^2+24^2+…+67^2+68^2

6^3=3^3+4^3+5^3

20^3=11^3+12^3+13^3+14^3

Deberemos fijar dos parámetros, el exponente de la potencia resultado de la suma, sea, por ejemplo **k**, y el de los sumandos, que es igual para todos, y llamaremos **h**. De esa forma también abarcamos la posibilidad de que el resultado no sea una potencia, salvo la trivial de exponente unidad:

294 =7^2+8^2+9^2+10^2 (k=1)

A la inversa, deberemos poder descomponer una potencia en suma de números naturales consecutivos, como potencias triviales:

12^5=82943+82944+82945 (h=1)

En la siguiente función usamos tres variables distintas, e integramos los resultados en modo texto:

n: base del total de la suma

k: exponente de n

h: exponente de los sumandos

m: total de sumandos en cada solución. Esta es útil para búsquedas.

Function sumapoteconsec$(n, k, h)
Dim p, s, i, j,m
Dim t$

t = "" ‘Texto vacío para incluir soluciones
p = n ^ k ‘Resultado deseado para la suma
For i = 1 To (n – 2)^(k/h) ‘Tope de búsqueda de sumandos
s = i ^ h ’Primer sumando potencia
j = i
While s <= p
If s = p And j > i Then m=j-i+1:t = t + " #" + ajusta(m)+”: “+Str$(i) + " a " + Str$(j)’Solución con número de sumandos, inicio y final
j = j + 1
s = s + j ^ h ‘Se acumula la suma
Wend
Next i
If t = "" Then t = "NO"

sumapoteconsec = t
End Function

Vemos algunos ejemplos obtenidos con esta función:

SUMAPOTECONSEC(30;1;1)= #5: 4 a 8 #4: 6 a 9 #3: 9 a 11

Significa que el número 30 (elevado a la unidad) es suma de números consecutivos de tres formas diferentes:

#5: 4 a 8 : 30=4+5+6+7+8

#4: 6 a 9 : 30=6+7+8+9

#3: 9 a 11 : 30=9+10+11

SUMAPOTECONSEC(990;1;2)= #5: 12 a 16

El número 990 es igual a la suma de cinco cuadrados:

$990=12^2+13^2+14^2+15^2+16^2$

Comprobamos el primer ejemplo de este texto:

SUMAPOTECONSEC(47;3;2)= #47: 22 a 68

Equivale a lo que ya sabíamos:

47^3=22^2+23^2+24^2+…+67^2+68^2

El cubo de 47 es suma de 47 cuadrados consecutivos.

Podríamos comprobarlo en una hoja de cálculo como indicamos en los primeros párrafos, restando la fórmula de la suma de cuadrados en 68 y en 21:

(2*68^3+3*68^2+68)/6-(2*21^3+3*21^2+21)/6=103823=47^3

Versión en PARI

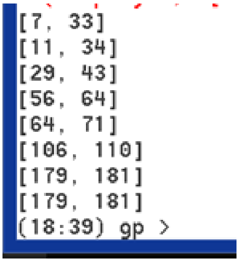
Para quienes deseen llegar a números grandes, se ofrece aquí una alternativa en PARI:

```
smpc(n,k,h)={my(v=[0,0],p=n^k,s,i,j);for(i=1,(n-2)^(k/h),s=i^h;j=i;while(s<=p,if(s==p&&j>i,v=[i,j];print(v));j+=1;s=s+j^h));v}
```

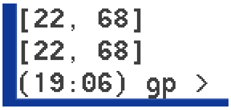
```
print(smpc(540,1,1))
```

Imprime las soluciones parciales y aparece repetida la final. Se podría corregir este detalle, pero al algoritmo va rápido y no merece la pena suprimirlo.

Solución para smpc(540,1,1)



Solución para smpc(47,3,2), que fue nuestro primer ejemplo:



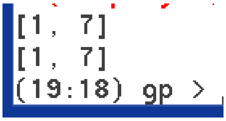
Confirma que el cubo de 47 es la suma de los cuadrados que van del 22^2 a 68^2.

Un ejemplo para confirmar:

smpc(29008,1,5)

¿Por qué ese número?

Vemos la solución:



Resulta que es la suma de las primeras siete potencias quintas. Es así porque el número 29008 lo hemos obtenido aplicando la fórmula presentada al principio:

S=(2*7^6+6*7^5+5*7^4-7^2)/12=29008.

Ejemplos concretos

Cubos que son suma de cubos

Acudimos a PARI, que es más rápido, para comprobar que 1155 posee esa propiedad:

k=1155;print(smpc(k,3,3))

Nos da que 1155^3 es igual a la suma de todos los cubos comprendidos entre 291^3 y 339^3

```
[291, 339]
[291, 339]
(18:43) gp >
```

(Consultar https://oeis.org/A097811)

Este resultado no se podría haber descubierto razonablemente con cálculo manual. Lo hemos comprobado con hoja de cálculo.

Cuadrados que son suma de cubos

Al efectuar una búsqueda de todos los casos, aparecen, entre otros, los números triangulares, por la conocida fórmula

$$1^3+2^3+3^3+\cdots+n^3=\left(\frac{n(n+1)}{2}\right)^2=(1+2+3+\cdots n)^2$$

Si buscamos los ejemplos de esta propiedad, encontraremos todos los números triangulares ordenados por su orden. En la siguiente imagen descubrimos que aparecen otros, como el 204, que no son triangulares. Son aquellos en los que la suma de cubos no comienza en 1^3:

Número N	¿Es triangular?	Orden del triangular
3	VERDADERO	2
6	VERDADERO	3
10	VERDADERO	4
15	VERDADERO	5
21	VERDADERO	6
28	VERDADERO	7
36	VERDADERO	8
45	VERDADERO	9
55	VERDADERO	10
66	VERDADERO	11
78	VERDADERO	12
91	VERDADERO	13
105	VERDADERO	14
120	VERDADERO	15
136	VERDADERO	16
153	VERDADERO	17
171	VERDADERO	18
190	VERDADERO	19
204	FALSO	0
210	VERDADERO	20
231	VERDADERO	21
253	VERDADERO	22
276	VERDADERO	23

Podemos crear un listado con los números cuyo cuadrado es suma de cubos pero que no son triangulares:

Número N	¿Es triangular?	Suma de cubos
204	FALSO	#3: 23 a 25
312	FALSO	#12: 14 a 25
315	FALSO	#5: 25 a 29
323	FALSO	#17: 9 a 25
504	FALSO	#8: 28 a 35
588	FALSO	#21: 14 a 34
720	FALSO	#15: 25 a 39
2079	FALSO	#33: 33 a 65
2170	FALSO	#5: 96 a 100
2940	FALSO	#5: 118 a 122

Como era de esperar, ninguna suma de cubos comienza con 1^3

Cubos que son suma de cuadrados

Este ejemplo es bastante conocido, pero con nuestras funciones podemos encontrar otros.

47^3=22^2+23^2+…+68^2

13156^3=2277044900416 es la suma de todos los cuadrados comprendidos entre 17354^2 y 22930^2

```
[17354, 22930]
[17354, 22930]
(19:24) gp >
```

Podemos usar la fórmula para sumar cuadrados, como comprobación:

22930*22931*(22930*2+1)/6-
17353*17354*(17353*2+1)/6

```
? a=22930*22931*(22930*2+1)/6-17353*17354*(17353*2+1)/6
print(a)
%11 = 2277044900416
2277044900416
```

Otras igualdades

Podemos usar lo aprendido para encontrar más igualdades en las que una potencia sea igual a la suma de varias otras potencias consecutivas. Esta es una muestra de lo encontrado con los primeros números como bases.

6^3=3^3+4^3+5^3

6^4=1^3+2^3+3^3+4^3+5^3+6^3+7^3+8^3

13^4=119^2+120^2

20^3=11^3+12^3+13^3+14^3

TRES POTENCIAS ENTERAS

En el desarrollo de mis cálculos sobre el año 2025 llegué a esta identidad:

$2025^2=36^4+18^5+9^6$

Al releerla se me ocurrió averiguar qué números admiten una descomposición en tres potencias enteras no negativas (admitiendo el 1 y el 0) y cuáles no. Por ejemplo, 72 se puede descomponer de seis formas:

$72=2^5+2^5+2^3=6^2+3^3+3^2=6^2+2^5+2^2=6^2+6^2+0^1=2^6+2^2+2^2=2^6+2^3+0^1$

Es evidente que el cero figura para poder considerar también las sumas de dos potencias, como $72=2^6+2^3$, o de una para los números que sean potencias perfectas.

Otros números, como 7 y 23 no admiten esta descomposición. En unas primeras búsquedas se puede sospechar que estos números son más escasos. Lo iremos viendo.

Búsqueda de soluciones

Esta búsqueda no supone ninguna complicación. Se resuelve con dos bucles, uno para la primera potencia y otro para la segunda, porque la tercera se encuentra restando.

Hemos usado una función para Excel, siguiendo el espíritu de este blog, pero usa una función propia, ESPOTENCIA

(ver https://hojaynumeros.blogspot.com/2022/04/numeros-consecutivos-con-una-suma-del.html).

También sustituye la función STR$ por la función AJUSTA, que funciona mejor. No obstante se incluirá su código aquí, porque a continuación se traducirá a PARI, con lo que todos los lectores podrán experimentar con ella.

```
Function trespotencias$(n)
Dim i, j, k, p1, p2, p3, m
Dim s$

s = "" 'Contenedor de la solución
m = 0 'Contador de soluciones
For i = 0 To n 'Bucle para la primera potencia
p1 = espotencia(i) 'Devuelve el exponente
If p1 > 0 Then
For j = 0 To i 'Bucle para la segunda potencia
p2 = espotencia(j)
If p2 > 0 Then
k = n - i – j 'Tercera potencia
```

```
p3 = espotencia(k)
If k <= j And p3 > 0 And k >= 0 Then
 'Los tres sumandos son potencias. Lo que sigue construye la solución
m = m + 1
s = s + " ## " + ajusta(Int(i ^ (1 / p1) + 0.000001)) + "^" + ajusta(p1) + "+"
s = s + ajusta(Int(j ^ (1 / p2) + 0.000001)) + "^" + ajusta(p2) + "+"
s = s + ajusta(Int(k ^ (1 / p3) + 0.000001)) + "^" + ajusta(p3)
End If
End If
Next j
End If
Next i
If s = "" Then s = "NO" Else s = ajusta(m) + " : " + s
trespotencias = s
End Function
```

Con esta función podemos descomponer los primeros números:

Número N	Soluciones
1	1 : ## 1^1+0^1+0^1
2	1 : ## 1^1+1^1+0^1
3	1 : ## 1^1+1^1+1^1
4	1 : ## 2^2+0^1+0^1
5	1 : ## 2^2+1^1+0^1
6	1 : ## 2^2+1^1+1^1
7	NO
8	2 : ## 2^2+2^2+0^1 ## 2^3+0^1+0^1
9	3 : ## 2^2+2^2+1^1 ## 2^3+1^1+0^1 ## 3^2+0^1+0^1
10	2 : ## 2^3+1^1+1^1 ## 3^2+1^1+0^1

Observamos que el 7 no admite esta descomposición.

Yal como el autor esperaba, al llegar a números mayores se incrementa el número de soluciones:

Número N	Soluciones
80	5 : ## 2^5+2^5+2^4 ## 6^2+6^2+2^3 ## 7^2+3^3+2^2 ## 2^6+2^3+2^3 ## 2^6+2^4+0^1
81	7 : ## 3^3+3^3+3^3 ## 6^2+6^2+3^2 ## 7^2+2^4+2^4 ## 7^2+2^5+0^1 ## 2^6+3^2+2^3 ## 2^6+2^4+1^1 ## 3^4+0^1+0^1
82	5 : ## 2^5+5^2+5^2 ## 7^2+5^2+2^3 ## 7^2+2^5+1^1 ## 2^6+3^2+3^2 ## 3^4+1^1+0^1
83	2 : ## 7^2+5^2+3^2 ## 3^4+1^1+1^1
84	4 : ## 2^5+3^3+5^2 ## 6^2+2^5+2^4 ## 7^2+3^3+2^3 ## 2^6+2^4+2^2
85	4 : ## 7^2+3^3+3^2 ## 7^2+2^5+2^2 ## 7^2+6^2+0^1 ## 3^4+2^2+0^1
86	4 : ## 2^5+3^3+3^3 ## 6^2+5^2+5^2 ## 7^2+6^2+1^1 ## 3^4+2^2+1^1
87	NO
88	3 : ## 6^2+3^3+5^2 ## 6^2+6^2+2^4 ## 2^6+2^4+2^3
89	7 : ## 2^5+2^5+5^2 ## 7^2+2^5+2^3 ## 7^2+6^2+2^2 ## 2^6+2^4+3^2 ## 2^6+5^2+0^1 ## 3^4+2^2+2^2 ## 3^4+2^3+0^1

Volvemos a encontrar un número que no presenta soluciones, el 87. De hecho, experimentalmente van desapareciendo estos números sin solución. Estos son los primeros:

Número N	
7	NO
15	NO
23	NO
87	NO
111	NO
119	NO
167	NO
335	NO

Están publicados en https://oeis.org/A113505 y al llegar al número 26375 se puede conjeturar que es posible que no aparezcan más. Suele ocurrir en casos similares.

A113505

Numbers not the sum of at most three perfect powers (A001597).

7, 15, 23, 87, 111, 119, 167, 335, 1391, 1455, 1607, 1679, 1991, 25887, 26375

a(16), if it exists, is larger than 10^8. - Giovanni Resta, May 07 2017

Para los lectores que deseen experimentar, se incluye nuestra versión en PARI.

```
u=100;for(i=0,u,if(ispower(i)||i<2,for(j=0,i,if((ispower(j)||j<2)&&(ispower(u-i-j)||u-i-j<2)&&j>u-i-j&&u-i-j>=0,print(i,", ",j,", ",u-i-j)))))
```

La variable **u** se rellena con el número a descomponer, en el ejemplo, 100

```
(17:55) gp > \r trespotencias.txt
64, 27, 9
64, 32, 4
64, 36, 0
(17:57) gp > _
```

Observamos que admite tres sumas con sumandos que son potencias.

La siguiente imagen recoge parte del resultado para el año 2025:

```
(18:03) gp > \r trespotencias.txt
784, 729, 512
841, 784, 400
841, 841, 343
900, 900, 225
1000, 625, 400
1000, 900, 125
1000, 961, 64
1000, 1000, 25
1024, 1000, 1
1089, 900, 36
1225, 784, 16
1296, 729, 0
1369, 400, 256
1369, 512, 144
1600, 256, 169
1600, 361, 64
1600, 400, 25
1681, 216, 128
1681, 343, 1
```

La abundancia de resultados reafirma la sospecha de que los elementos sin solución serán limitados.

Estudio con nuestra herramienta Cartesius

Esta hoja de Excel, “Cartesius”, permite combinar muchas posibilidades, y resulta adecuada para esta cuestión. Se puede programar con estas condiciones:

Escribe a partir de la siguiente fila ⇓⇓⇓ (no dejes filas en blanco)
xtotal=3
xt=0..2025
xt=filtro(potencia)
suma=2025
creciente

Se interpretan como que buscamos tríos de potencias con suma 2025. Se consigue el resultado siguiente, compuesto por 28 resultados:

X1	X2	X3
0	0	2025
0	729	1296
1	343	1681
1	1000	1024
8	81	1936
8	289	1728
16	784	1225
25	64	1936
25	400	1600
25	1000	1000
32	144	1849
36	225	1764
36	900	1089
64	361	1600
64	961	1000
81	216	1728
125	900	1000
128	169	1728
128	216	1681
144	512	1369
169	256	1600
225	900	900
256	400	1369
343	841	841
400	400	1225
400	625	1000
400	784	841
512	729	784

Tal como se sugirió al principio, esta búsqueda no es complicada, y produce un incremento tan grande de resultados que es razonable la conjetura de que a partir de cierto número, todos los enteros presentarán esta descomposición.

PALÍNDROMOS TRIPLES

El día 26 de marzo de 2021 publiqué esta igualdad de tipo palindrómico:

26321=16*5*61+16561+16*5*61

La triple repetición simétrica de las cifras 16561 es muy atractiva, e invita a descubrir casos similares. En principio se pueden considerar dos variantes de la cuestión. La primera seguiría el modelo del ejemplo, con un número impar de cifras y un capicúa central. Como ocurre en 125=1*2*1+121+1*2*1. La segunda consistiría en un esquema que prescinda de ese elemento central en cada sumando, como es el caso de 261=9*9+99+9*9.

La construcción de una lista con los primeros números que se siguen un esquema de este tipo no es difícil. La lista de los primeros del tipo

AB*C*BA+ABCBA+AB*C*BA es la siguiente:

Número N	Soluciones
13229	## 12*2*21+12221+12*2*21
13833	## 12*3*21+12321+12*3*21
14437	## 12*4*21+12421+12*4*21
14843	## 13*2*31+13231+13*2*31
15041	## 12*5*21+12521+12*5*21
15645	## 12*6*21+12621+12*6*21
15749	## 13*3*31+13331+13*3*31
16249	## 12*7*21+12721+12*7*21
16537	## 14*2*41+14241+14*2*41
16655	## 13*4*31+13431+13*4*31
16853	## 12*8*21+12821+12*8*21
17457	## 12*9*21+12921+12*9*21
17561	## 13*5*31+13531+13*5*31
17785	## 14*3*41+14341+14*3*41
18311	## 15*2*51+15251+15*2*51
18467	## 13*6*31+13631+13*6*31
19033	## 14*4*41+14441+14*4*41
19373	## 13*7*31+13731+13*7*31
19941	## 15*3*51+15351+15*3*51

Con paciencia y una calculadora se puede ir construyendo. Lo interesante es descubrir soluciones para otro rango distinto, en el que el orden natural no ayude mucho. La siguiente tabla, a partir del número 50000, demuestra que los resultados no son ya tan previsibles:

Número N	Soluciones
50293	## 35*4*53+35453+35*4*53
50488	## 42*4*24+42424+42*4*24
50836	## 29*4*92+29492+29*4*92
50899	## 38*2*83+38283+38*2*83
50998	## 41*8*14+41814+41*8*14
51000	## 27*6*72+27672+27*6*72
51113	## 19*9*91+19991+19*9*91
51205	## 50*2*5+5025+50*2*5
51542	## 28*5*82+28582+28*5*82
51805	## 50*3*5+5035+50*3*5
52106	## 43*3*34+43334+43*3*34
52187	## 34*6*43+34643+34*6*43
52246	## 41*9*14+41914+41*9*14
52405	## 50*4*5+5045+50*4*5
52604	## 42*5*24+42524+42*5*24
52654	## 26*8*62+26862+26*8*62

Observamos que el algoritmo no mantiene el cero a la izquierda en 52405, porque Excel lo suprime por defecto. Es preferible corregirlo manualmente a tenerlo previsto en la programación. Es el precio por usar hojas de cálculo.

Función “PALINTRI”

Para construir un algoritmo adaptado a nuestra búsqueda, parece conveniente fijar como datos de entrada un número **N**, que es el que se quiere representar así, y dos parámetros, **ka** y **kb**, que representarían el número de cifras de cada elemento del esquema. Si uno de ellos es nulo, produciría variantes, como las dos que se han presentado en el párrafo anterior. Si kb=0 se entenderá que no hay elemento central.

La fijación del número de cifras parece aconsejable, porque aparecen más ejemplos de los previstos. Por ejemplo, con cinco cifras, dos cifras para los elementos laterales y una para el central, aparecen 416 ejemplos, desde 13229=12*2*21+12221+12*2*21 hasta 99974=64*6*46+64646+64*6*46. Incluso existen dos números que presentan dos soluciones:

46006=41*4*14+41414+41*4*14=26*6*62+26662+26*6*62

97925=58*4*85+58485+58*4*85=39*8*93+39893+39*8*93

La organización de la búsqueda se basará en la variable **a1**, que representará a la parte lateral, con su simétrico **a2**, y su número de cifras **ka**. Así, en el ejemplo anterior a1=58, a2=85 y ka=2. Exigiremos que a1 sea distinto de a2 si tienen al menos dos cifras, para evitar trivialidades.

Procederemos de la misma forma con la parte central, llamando **b** a su valor, con la condición de que sea capicúa, y **kb** a su número de cifras. En el ejemplo b=4 y kb=1.

Con estas variables, y en sistema de numeración decimal, el valor de N quedaría:

N=2*a1*a2*b+a2+b*10^ka+a1*10^(ka+kb)

En el ejemplo:

97925=2*58*85*4+85+4*100+58*1000

En esta expresión nos basaremos para construir el algoritmo.

Para evitar búsquedas inútiles deberemos contar con que ***2ka+kb*** es una buena cota inferior para las cifras de N.

Si b=1 y kb=0 porque no se usa término central, la expresión se simplifica mucho:

N=(2*a1+1)*a2+a1*10^ka

Como ejemplo, 3804=23*32+2332+23*32 y se cumple:

3804=(2*23+1)*32+23*100=1504+2300=3804

Si integramos las dos expresiones en una misma función, su código será algo más largo de lo habitual, pero merece la pena.

Versión en Excel

Las ideas anteriores están plasmadas en la siguiente función. Se han integrado las dos versiones del problema. La división se basa en si kb es mayor que cero o no:

Function palintri$(n, ka, kb)’Parámetros N y número de cifras
Dim kn, a1, a2, b, m, c, b1, aa1
Dim s$, t$

s = ""’Contenedor de resultados
c = 0 ’Contador de resultados
If kb > 0 Then ‘Caso con elemento central
If kb = 1 Then b1 = 1 Else b1 = 10 ^ (kb - 1) + 1 ‘Origen de búsquedas
For b = b1 To 10 ^ kb – 1 ‘Cifras centrales
If escapicua(b) Or kb = 1 Then ‘La variable b ha de ser capicúa
If ka = 1 Then aa1 = 1 Else aa1 = 10 ^ (kb - 1) + 1’ Origen de a
For a1 = aa1 To 10 ^ ka – 1 ‘Búsqueda de a
a2 = cifrainver(a1) ‘Simétrico de a
If a1 <> a2 Or ka = 1 Then ‘Condición algebraica

```
m = 2 * a1 * a2 * b + a2 + b * 10 ^ ka + a1 * 10 ^ (ka + kb)
If m = n Then ‘Hay solución
c = c + 1 ‘Se comunica solución
t = ajusta(a1) + "*" + ajusta(b) + "*" + ajusta(a2)
s = s + " ## " + t + "+" + ajusta(a1) + ajusta(b) + ajusta(a2) + "+" + t
End If
End If
Next a1
End If
Next b
Else ‘ Segunda variante, sin elemento central
If ka = 1 Then aa1 = 1 Else aa1 = 10 ^ (ka - 1) + 1
For a1 = aa1 To 10 ^ ka – 1 ‘Proceso similar
a2 = cifrainver(a1)
If a1 <> a2 Or ka = 1 Then
m = (2 * a1 + 1) * a2 + a1* 10 ^ ka
If m = n Then
c = c + 1
t = ajusta(a1) + "*" + ajusta(a2)
s = s + " ## " + t + "+" + ajusta(a1) + ajusta(a2) + "+" + t
End If
End If
Next a1
```

End If
If s <> "" Then s = ajusta(c) + " : " + s Else s = "NO"
palintri = s
End Function

En un buscador jugaremos con los parámetros para conseguir los resultados adecuados. Por ejemplo, estos son los primeros números de 4 cifras del tipo A*A'+AA'+A*A'

Número N	Soluciones
1725	1 : ## 12*21+1221+12*21
2082	1 : ## 20*2+202+20*2
2137	1 : ## 13*31+1331+13*31
2589	1 : ## 14*41+1441+14*41
2616	1 : ## 21*12+2112+21*12
3081	1 : ## 15*51+1551+15*51
3183	1 : ## 30*3+303+30*3
3613	1 : ## 16*61+1661+16*61
3804	1 : ## 23*32+2332+23*32
3919	1 : ## 31*13+3113+31*13
4185	1 : ## 17*71+1771+17*71
4324	1 : ## 40*4+404+40*4
4458	1 : ## 24*42+2442+24*42
4695	1 : ## 32*23+3223+32*23
4797	1 : ## 18*81+1881+18*81

Persiste el problema del 0 inicial que suprime Excel, pero se entiende bien.

De igual forma, se detectan los números del tipo A*BB*A'+ABBA'+A*BB*A':

3388	1 : ## 2*77*2+2772+2*77*2
3586	1 : ## 2*88*2+2882+2*88*2
3619	1 : ## 3*22*3+3223+3*22*3
3784	1 : ## 2*99*2+2992+2*99*2
3927	1 : ## 3*33*3+3333+3*33*3
4235	1 : ## 3*44*3+3443+3*44*3
4466	1 : ## 4*11*4+4114+4*11*4
4543	1 : ## 3*55*3+3553+3*55*3
4851	1 : ## 3*66*3+3663+3*66*3
4928	1 : ## 4*22*4+4224+4*22*4
5159	1 : ## 3*77*3+3773+3*77*3
5390	1 : ## 4*33*4+4334+4*33*4
5467	1 : ## 3*88*3+3883+3*88*3

VERSION PARI (para la primera variante)

Si deseamos más rango de soluciones puede ser útil la versión en PARI. La hemos diseñado para que devuelva sólo **a1** y **b**, porque el resto se completa fácilmente. Su código es

```
ori(k)=my(m=1);if(k>1,m=10^(k-1)+1);m
simetrico(n)=eval(concat(Vecrev(Str(n))))
palind(n)=n==simetrico(n)
```

```
palintri1(n,ka,kb)=my(a1,a2,v=[ka,kb],b,m,es=0);f
or(b=ori(kb),10^kb-
1,if(palind(b)||kb==1,for(a1=ori(ka),10^ka-
1,a2=simetrico(a1);if(a1<>a2||ka==1,m=2*a1*a2*b
+a2+b*10^ka+a1*10^(ka+kb);if(m==n,v[1]=a1;v[2]
=b;es=1;print(n," es ",v[1],", ",v[2]))))))
for(i=10000,20000,if(palintri1(i,2,1),print(i)))
```

Basta cambiar la última línea según nuestras preferencias. Tal como está, repetiría la búsqueda efectuada con Excel. Este es un fragmento:

```
%4 = (n,ka,kb)->my(a1,a2
a),10^ka-1,a2=simetrico(
[2]=b;es=1;print(n," es
12625 es 12, 1
13229 es 12, 2
13833 es 12, 3
13937 es 13, 1
14437 es 12, 4
14843 es 13, 2
15041 es 12, 5
15289 es 14, 1
15645 es 12, 6
15749 es 13, 3
16249 es 12, 7
16537 es 14, 2
```

Por ejemplo, 16537=14*2*41+14241+14*2*41

VERSION PARI (para la segunda variante)

```
ori(k)=my(m=1);if(k>1,m=10^(k-1)+1);m
simetrico(n)=eval(concat(Vecrev(Str(n))))
palind(n)=n==simetrico(n)
palintri2(n,ka)=my(a1,a2,v=[ka,ka],m,es=0);for(a1
=ori(ka),10^ka-
1,a2=simetrico(a1);if(a1<>a2||ka==1,m=(2*a1+1)*a
2+a1*10^ka;if(m==n,v[1]=a1;v[2]=a2;es=1;print(n,
" es ",v[1],", ",v[2]))));es
for(i=1000,9999,if(palintri2(i,2),print1("")))
```

```
%4 = (n,ka)->my(a1,a2,v=[k
if(m==n,v[1]=a1;v[2]=a2;es
1725 es 12, 21
2082 es 20, 2
2137 es 13, 31
2589 es 14, 41
2616 es 21, 12
3081 es 15, 51
3183 es 30, 3
3613 es 16, 61
3804 es 23, 32
3919 es 31, 13
4185 es 17, 71
4324 es 40, 4
4458 es 24, 42
4695 es 32, 23
4797 es 18, 81
5152 es 25, 52
```

Coinciden los resultados con los de Excel. Por ejemplo, 3613=16*61+1661+16*61

Con más cifras

Finalizamos con ejemplos que presentan más cifras en su desarrollo. Todos ellos se podrían encontrar realizando bucles para sus componentes, pero aquí se prefieren funciones para poder elegir el rango de búsqueda de N, y no los de sus componentes.

Seis cifras del tipo AB*CC*BA+ABCCBA+AB*CC*BA:

```
153241 es 12, 55
153769 es 14, 11
159885 es 12, 66
159929 es 13, 33
166529 es 12, 77
167497 es 14, 22
167981 es 15, 11
169895 es 13, 44
173173 es 12, 88
179817 es 12, 99
179861 es 13, 55
181225 es 14, 33
182633 es 16, 11
185911 es 15, 22
189827 es 13, 66
```

Comprobamos uno:

173173=12*88*21+128821+12*88*21

Es curioso que el resultado sea una concatenación de 173 consigo mismo.

Con cinco cifras y tres centrales

Este es un rango de soluciones obtenidas con Excel:

27994	1 : ## 2*444*2+24442+2*444*2
28174	1 : ## 2*454*2+24542+2*454*2
28354	1 : ## 2*464*2+24642+2*464*2
28534	1 : ## 2*474*2+24742+2*474*2
28714	1 : ## 2*484*2+24842+2*484*2
28894	1 : ## 2*494*2+24942+2*494*2
29092	1 : ## 2*505*2+25052+2*505*2
29272	1 : ## 2*515*2+25152+2*515*2
29452	1 : ## 2*525*2+25252+2*525*2
29632	1 : ## 2*535*2+25352+2*535*2
29812	1 : ## 2*545*2+25452+2*545*2
29992	1 : ## 2*555*2+25552+2*555*2

Así podríamos seguir.

DIFERENCIA DE DOS CUBOS IGUAL A UNA SUMA

Caso general

En los cálculos que publico diariamente uso dos funciones para averiguar si un número es suma de cubos o bien diferencia. No se me había ocurrido simultanear ambas propiedades y lo hago ahora.

¿Qué números enteros positivos son suma de dos cubos y simultáneamente diferencia de otros dos, siendo en ambos casos cubos enteros positivos?

Un ejemplo es el 152, que por una parte equivale a 3^3+5^3=27+125, y por otra a 6^3-4^3=216-64.

Con las publicaciones precedentes tenemos acceso a dos funciones que encuentran estas descomposiciones: SUMCUBOS y DIFCUBOS. Las copio a continuación:

Function sumcubos(n)
Dim k, a, m, b
Dim s$

s = "" ‘Contenedor de resultados
m = 0 ‘Contador de soluciones
a = n ^ (1 / 3) *‘Máximo cubo posible*
For k = 1 To a
b = n - k ^ 3 ‘Segundo posible cubo
If escubo(b) And k <= b ^ (1 / 3) Then m = m + 1: s = s + " a=" + Str$(k) + " b=" + Str$(Int(b ^ (1 / 3) + 0.0001)) *‘Hay nueva solución*
Next k ‘Si no hay solución devuelve “NO”
End Function

Usa nuestra función ESCUBO:

Function escubo(n)
Dim a
a = Int(n ^ (1 / 3) + 10 ^ (-6))
If a * a * a = n Then escubo = True Else escubo = False
End Function

La segunda función es algo más complicada, porque la diferencia presenta otro condicionante, y es que la diferencia de cubos ha de ser divisor de N.

```
Function difcubos$(n)

Dim k, a, t, m, p
Dim s$

s = "" ‘Contenedor de soluciones
m = 0 ‘Contador de soluciones
For k = 1 To n / 2 ‘La diferencia de bases es divisor de N
If n / k = n \ k Then ‘Criterio de divisibilidad
t = Sqr(n / k / 3) ‘Máximo cubo con esa diferencia
For a = 1 To t
If (a + k) ^ 3 - a ^ 3 = n Then m = m + 1: s = s + " # a=" + Str$(a) + " b=" + Str$(a + k) ‘Existe solución
Next a
End If
Next k
If s = "" Then difcubos = "NO" Else difcubos = ajusta(m) + " " + s
End Function
```

El que esta segunda función use también el “NO” para cuando no existe solución nos permite simultanear las dos condiciones:

SUMCUBOS(N)<>"NO" AND DIFCUBOS(N)<>"NO"

Usamos esta condición en un buscador y nos devolverá la lista de números enteros positivos que son simultáneamente suma y diferencia de dos cubos (también enteros positivos)

Número N	Suma de cubos	Diferencia de cubos
91	1: a= 3 b= 4	1 # a= 5 b= 6
152	1: a= 3 b= 5	1 # a= 4 b= 6
189	1: a= 4 b= 5	1 # a= 3 b= 6
217	1: a= 1 b= 6	1 # a= 8 b= 9
513	1: a= 1 b= 8	1 # a= 6 b= 9
728	1: a= 6 b= 8	2 # a= 10 b= 12 # a= 1 b= 9
1027	1: a= 3 b= 10	1 # a= 18 b= 19
1216	1: a= 6 b= 10	1 # a= 8 b= 12
1512	1: a= 8 b= 10	1 # a= 6 b= 12
1736	1: a= 2 b= 12	1 # a= 16 b= 18
2457	1: a= 9 b= 12	1 # a= 15 b= 18
3087	1: a= 7 b= 14	1 # a= 17 b= 20
4104	2: a= 2 b= 16 a= 9	1 # a= 12 b= 18
4706	1: a= 11 b= 15	1 # a= 27 b= 29
4921	1: a= 2 b= 17	1 # a= 40 b= 41
4977	1: a= 4 b= 17	1 # a= 22 b= 25

Están publicados en https://oeis.org/A225908, y es subsecuencia de https://oeis.org/A051347 En esas páginas se llama la atención sobre que estas propiedades permiten descomponer algunos cubos en suma de otros tres. Ya he comentado esta propiedad anteriormente en otras ocasiones. Por ejemplo, 18^3=16^3+12^3+2^3. Esta igualdad se extrae de los resultados del número 4104.

Si se usa la función TRESCUBOS (no se explica aquí porque usa otras funciones que alargarían este estudio) con uno de los cubos mayores, se reproducirán algunos resultados. Por ejemplo, con 18^3, si lo descomponemos en suma de tres cubos, daría lugar a

18^3=15^3+6^3+9^3=16^3+6^3+2^3

Si pasamos restando algún sumando, obtendríamos algunos casos de los estudiados. Por ejemplo:

18^3-15^3=9^3+6^3

A continuación estudiaremos algunos casos y propiedades particulares.

Casos particulares de los cubos sumandos

Cubos consecutivos

Los dos cubos que se suman pueden ser consecutivos. Añadiendo algún parámetro a nuestra función se pueden encontrar esos casos particulares. No abundan. Estos son los inferiores a 10^5:

Número N	Suma de cubos consecutivos	Diferencia de cubos
91	1: a= 3 b= 4	1 # a= 5 b= 6
189	1: a= 4 b= 5	1 # a= 3 b= 6
12691	1: a= 18 b= 19	1 # a= 21 b= 28
68705	1: a= 32 b= 33	1 # a= 6 b= 41
97309	1: a= 36 b= 37	1 # a= 3 b= 46

En ellos se ha de cumplir que

$N=(a+1)^3+a^3=2a^3+3a^2+3a+1$, luego **a** será un divisor de N-1.

Por ejemplo. 68705-1 se descompone como $2^5*19*113$, y, efectivamente, $32=2^5$ es un divisor suyo, y es la base del cubo menor de la suma.

Cubos de N y N+2

Con el mismo procedimiento obtenemos los primeros casos en los que las bases de los cubos que se suman se diferencian en dos unidades.

Número N	Suma de cubos de N y N+2	Diferencia de cubos
152	1: a= 3 b= 5	1 # a= 4 b= 6
728	1: a= 6 b= 8	2 # a= 10 b= 12 # a= 1 b= 9
1512	1: a= 8 b= 10	1 # a= 6 b= 12
16120	1: a= 19 b= 21	1 # a= 18 b= 28

Es sencillo demostrar que aquí la base del cubo menor ha de ser divisor de N-8, como ocurre con 16120, en el que (16120-8)/19=848

Cubos de base prima

Para finalizar, se adjuntan los tres primeros resultados para el caso en el que las bases de los cubos que se suman sean números primos. Se acumulan las exigencias y es normal que resulten pocos resultados.

Número N	Suma de cubos de base prima	Diferencia de cubos
152	1: a= 3 b= 5	1 # a= 4 b= 6
4921	1: a= 2 b= 17	1 # a= 40 b= 41
5256	1: a= 7 b= 17	1 # a= 14 b= 20

Versión en PARI

Para quienes deseen practicar con este lenguaje, se inserta a continuación un código que devuelve las soluciones entre1 y 5000. Es interesante estudiar el uso de vectores para devolver las soluciones:

```
sumadoscubos(n)=my(i=1,m,v=[0,0]);while(i<=n^(1/3),m=n-i^3;if(ispower(m,3)&&m<=i^3,v=[i,m^(1/3)]);i+=1);v
```

```
difcubos(n)=my(k,t,a,v=[0,0]);for(k=1,n,if(n%k==0,t=sqrt(n/k/3);for(a=1,t,if((a+k)^3-a^3==n,v=[a+k,a]))));v

for(m=1,5000,u=difcubos(m);t=sumadoscubos(m);if(u<>[0,0]&&t<>[0,0],print(m," Suma ",t," Diferencia ",u)))
```

Resultado:

```
(18:58) gp > \r C:\Users\arold\Documents\P
%48 = (n)->my(i=1,m,v=[0,0]);while(i<=n^(1
%49 = (n)->my(k,t,a,v=[0,0]);for(k=1,n,if(
91 Suma [4, 3] Diferencia [6, 5]
152 Suma [5, 3] Diferencia [6, 4]
189 Suma [5, 4] Diferencia [6, 3]
217 Suma [6, 1] Diferencia [9, 8]
513 Suma [8, 1] Diferencia [9, 6]
728 Suma [8, 6] Diferencia [9, 1]
1027 Suma [10, 3] Diferencia [19, 18]
1216 Suma [10, 6] Diferencia [12, 8]
1512 Suma [10, 8] Diferencia [12, 6]
1736 Suma [12, 2] Diferencia [18, 16]
2457 Suma [12, 9] Diferencia [18, 15]
3087 Suma [14, 7] Diferencia [20, 17]
4104 Suma [16, 2] Diferencia [18, 12]
4706 Suma [15, 11] Diferencia [29, 27]
4921 Suma [17, 2] Diferencia [41, 40]
4977 Suma [17, 4] Diferencia [25, 22]
(18:59) gp > |
```