Tarea 5
Algoritmos y Complejidad
2021-1

Camilo Contreras
201873063-7

7 de junio de 2021

Concepto	**Tiempo [min]**
Investigación	4[h]
Desarrollo	3[h]
Informe	2[h]

Los antiguos egipcios representaban fracciones como sumas de fracciones con numerador 1. O sea, por ejemplo:

$$\frac{3}{5} = \frac{1}{2} + \frac{1}{10}$$

A tales representaciones se les llama *fracciones egipcias*.

Un algoritmo simple para obtener una representación es el de Fibonacci (un algoritmo voraz). Dada la fracción $f = t/b$, donde $t > 1$, obtenga la máxima fracción con denominador 1 menor que f, o sea $1/\lceil b/t \rceil$, réstela de f y siga con el proceso.

Por ejemplo:

$$\begin{aligned}
\frac{779}{1020} & & \left\lceil \frac{1020}{779} \right\rceil &= 2 \\
&= \frac{1}{2} + \frac{269}{1020} & \left\lceil \frac{1020}{269} \right\rceil &= 4 \\
\frac{269}{1020} &= \frac{1}{4} + \frac{7}{510} & \left\lceil \frac{510}{7} \right\rceil &= 73 \\
\frac{7}{510} &= \frac{1}{73} + \frac{1}{37230}
\end{aligned}$$

Vale decir:

$$\frac{779}{1020} = \frac{1}{2} + \frac{1}{4} + \frac{1}{73} + \frac{1}{37230}$$

1. Demuestre que los numeradores de las fracciones disminuyen, por lo que necesariamente terminan en 1; el proceso termina en un número finito de pasos. (30 puntos)

   Tenemos una fraccion $\frac{n}{m}$, cuando n = 1. logramos nuestro objetivo. Ahora asusmimos que por cada n ∈ (0,1,2.. k) y que el algoritmo terminara cuando la fraccion dada sea $\frac{n}{m}$.

   Tomamos n = k + 1: Ahora calculamos nuestra nueva fraccion restandole la fraccion unitaria a la fraccion original:

$$\frac{k+1}{m} - \frac{1}{\lceil \frac{m}{k+1} \rceil} = \frac{\lceil \frac{m}{k+1} \rceil (k+1) - m}{\lceil \frac{m}{k+1} \rceil * m}$$

   Por propiedades de la funicon techo y funcion suelo, llegamos a que el numerador de la nueva funcion = k

$$\lceil \frac{m}{k+1} \rceil (k+1) - m = k$$

   Por lo que queda demostrado por induccion que el algoritmo tiene pasos finitos y disminuye su valor del numerador.

2. Escriba una función Python que dada una fracción propia a/b con $a < b$ entregue la lista de denominadores de una fracción egipcia. Use cálculo con números enteros. La función `math.gcd(x, y)` entrega el máximo común divisor de x, y.

   Su programa debe leer n, luego n fracciones de la entrada estándar (como numerador, denominador); para cada fracción debe dar esta, luego la lista de denominadores. En nuestro ejemplo escribiría:

   ```
   779/1020: 2, 4, 73, 37230
   ```

   Respuesta : Importamos la librteria "fractions", la cual nos permite trabajar con fracciones. Creamos nuestra funcion recursiva llamada egipciosVoraces(), esta consta de un arreglo en el que se van agregando las fracciones unitarias encontradas. Nuestra condicion de termino de la recursion es si el nominador que entregamos a la funcion es 1, ya que si es asi, no es necesario seguir adelente y el denominador de esta fraccion se agrega directamente al arreglo. Finalmente retornamos el arreglo con los denominadores.

   ```
   def egipciosVoraces(frac):
   arr = []
   if frac.numerator == 1:
      arr.append(str(frac.denominator))
      return arr
   ```

   Finalmente jugamos con los datos para configurar el output.

   ```
   x = (input()).split(" ")
   n = int(x[0])
   d = int(x[1])
   frac = Fraction(n, d)
   print((str(n) +"/"+ str(d) + ": " +
            ",".join(egipciosVoraces(frac))))
   ```

   Un ejemplo de input y output:

   ```
   3
   779 1020
   779/1020: 2,4,73,37230
   123 456
   123/456: 4,51,7752
   21 44
   21/44: 3,7,924
   ```

   Cabe destacar que asumimos que nunca se introduciran fracciones impropias, es decir, fracciones cuyo numerador es mayor a su denominador.