

Primes and Perfects Numbers

Universidad de Cuenca

Optativa 6 - Criptología

Dr. Diego Ponce

Capítulo 2

Freddy L. Abad L.
freddy.abadl@ucuenca.edu.ec

(2.1) Prove that there are infinitely many primes which are one less than a multiple for 4. Hint show that any integer of the form $4n-1$ must be divisible by a prime of the same form.

- Si hay k números primos \Rightarrow El conjunto de números primos es

$$S = \{ p_1, p_2, \dots, p_k \}$$

- Definimos N como

$$N = 4(p_1, p_2, \dots, p_k) - 1$$

$$p_1 = 3 \quad p_2 = 7 \quad p_3 = 11 \quad (\dots)$$

- Si N es primo es de la forma $4n-1$ pero $N \notin S$, lo que significa que no es el conjunto completo de primos de la forma $4n-1$

$\therefore N$ es un # compuesto

- suponiendo que todos los factores primos de N son de la forma $4n-1$

Entonces del producto de enteros de la forma $4n-1$ se sigue que N es de la forma $4n-1$

$\therefore N$ debe tener al menos un factor primo "p" de la forma $4n+1$

* Observamos que $N+1 = 4(p_1, p_2, \dots, p_k)$ es divisible para p_1, p_2, \dots, p_k

$\therefore N$ no es divisible para ningún p_1, p_2, \dots, p_k que son primos de la forma $4n-1$

Si p_1, p_2, \dots, p_k contiene todos los primos de la forma $4n-1$, debe existir un "p" $\notin S$ que divide a N

Se da una contradicción con la suposición inicial

FREDDY
ABAD

Norma

La fibra de caña de azúcar es totalmente responsable con el medio ambiente.



2.2 Using Theorem 2.2, approximately how many hundred digit primes are there? How does this compare with the number of primes with at most a hundred digits?

$$P(n) = \frac{1}{\ln(n)}$$

$$P(10^{100}) - P(10^{99}) = \frac{10^{100}}{\ln(10^{100})} - \frac{10^{99}}{\ln(10^{99})}$$

$$P(10^{99}) - P(10^{98}) = \frac{10^{100} \ln(10^{99}) - 10^{99} \ln(10^{100})}{\ln(10^{100}) \ln(10^{99})}$$

$$P(10^{100}) - P(10^{99}) = \frac{99 \cdot 10^{99} \ln(10) - 10 \cdot 10^{99} \ln(10)}{100 \ln(10) \cdot 99 \ln(10)}$$

$$P(10^{100}) - P(10^{99}) = \frac{\ln(10)(99 \cdot 10^{99} - 100 \cdot 10^{99})}{100 \cdot 99 \ln(10) \cdot \ln(10)}$$

$$P(10^{100}) - P(10^{99}) = \frac{\ln(10)(99 \cdot 10^{99} - 100 \cdot 10^{99})}{100 \cdot 99 \ln(10) \ln(10)}$$

$$P(10^{100}) - P(10^{99}) = \frac{10^{99} \cdot 890 \cdot \ln(10)}{10^2 \cdot 99 \cdot \ln(10) \ln(10)}$$

$$P(10^{100}) - P(10^{99}) = \frac{890}{99 \ln(10)} \cdot 10^{97}$$

$$\therefore P(10^{100}) - P(10^{99}) = 3,904263 \times 10^{97}$$

(*) What is the asymptotic formula for the number of perfect squares less than or equal to n ?

$$P(10^{99}) = \frac{10^{99}}{\ln(10)}$$

$$= \frac{10^{99}}{99 \ln(10)}$$

$$= \left(\frac{1}{99} \cdot \frac{1}{\ln(10)} \right) \times 10^{99}$$

$$= \left(\frac{1}{99} \cdot \frac{1}{\ln(10)} \right) \times 10^{99}$$

$$= (4,38 \times 10^{-3}) \times 10^{99}$$

$$= 4,38 \times 10^{96}$$

Cantidad de # primos de max 100 dígitos: $4,38 \times 10^{96}$

Cantidad # primos de 100 dígitos $3,90 \times 10^{97}$

la cantidad de # primos de maximo 100 dígitos es aprox. la misma cantidad que la cantidad de # primos con 100 dígitos

$$R = \frac{\# \text{ primos} > 100 \text{ dígitos}}{\# \text{ primos} < 100 \text{ dígitos}} = \frac{3,90 \times 10^{97}}{4,38 \times 10^{96}} = 0,89$$

Por cada # primo con dígitos menores a 100, hay 0,89 # primos con 100 dígitos.

2.3 What is the asymptotic formula for the number of perfect squares less than or equal to n ? (20%)

El número de cuadrados perfectos menor o igual que n es \sqrt{n}

Por ejemplo: cuadrados perfectos menor o igual que 100

<u>Nº</u>	<u>Cuadrado Perfecto</u>	<u>n</u>	<u>\sqrt{n}</u>	<u># cuadrados perfectos</u>
1	1			
2	4			
3	9			
4	16			
5	25			
6	36			
7	49			
8	64			
9	81			
10	100			
11	121			

R/1

2.4 Write a program to implement Algorithm 2.3. Use it to find the primes less than or equal to 5000.

public static void sieve Eratosthenes (Integer n)?

```
Integer a = new Integer [n+1];
for (int i=2; i<=n; i++) {
    a[i] = i;
}
```

```
Integer j = 2;
while ((int) Math. pow (j, 2)) <= n) {
    if (a[j] != 0) {
        Integer i = 2*j;
        while (i <= n) {
            a[i] = 0;
            i += j;
        }
    }
}
```

```
j
System.out.println ("Input: " + n);
for (int i=2; i<=n; i++) {
    if (a[i] != 0)
        System.out.println ("Next Prime" + a[i]);
```

```
public static void main (String [] args) {
    sieve Eratosthenes (5000);
```

FROM 2.5-2.7 USE TABLE GENERATED IN EX 2.4

2.5 How many pairs of primes in the table differ by 2? A famous unsolved problem asks if there are infinitely many such pairs among all the primes?

2, 3, 5, 7, 11, 13, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73...

Ejemplo

Pairs/Twin primes $(3, 5), (5, 7), (11, 13), (29, 31), (41, 43), (59, 61), (71, 73)$
total Pairs Primes (1-100) = 7

Aplicacion con N° Primos pares menores a 5000

$$\text{Constante de Brun} \Rightarrow \pi_2(x) = O\left(\frac{x(\log \log x)^2}{(\log x)^2}\right) \quad x \geq 3$$

\rightarrow UNITE ASINTOTICO EN PRIMOS GEMELOS

$$\therefore \pi_2(5000) = O\left(\frac{5000(\log(\log(5000)))^2}{(\log(5000))^2}\right)$$

$\therefore \pi_2(5000) = 117,93$

Calculando q mrs los twin/pairs primes se obtienen 126, si bien el error es considerable, esto formula funciona para estimar en te límites muy grandes, donde el error disminuye.

la Verificación se realizo con prime sieve, un repositorio donde se soluciona este problema Haciendo mediante codigo en python.

2.6 How evenly are the primes in the table divided between those one more than a multiple of four and those which are one less than a multiple of four? Are you prepared to make any conjectures? (Can you prove conjectures?)

2	3	5	7	11	13	17	19	23	29	31	37	41	43	47	53
Multiplos de 4:															
Diferencia	+4	-4	-			+6	-	+6	+4	-	+6	+4	-	+6	-

53	59	61	67	71	73	79	83	89	...	4463	4461	4460	4459	4458
+6	-4	+6	+2	-4	+6	+4	+6	+10	-	+18	+6	+6	-	+6

Puedo asumir que los # primos cercanos a un multiplos de 4 se distribuyen con una distancia par, es decir

Sea: $x \rightarrow \# \text{ multiplo de 4}$

$P_i, P_{i+1} \rightarrow \# \text{ primos consecutivos que contiene el } x$

$$|P_i, P_{i+1}| = d \quad d \bmod 2 = 0$$

- Si puedes probar, escribiendo codigo, con una bandera que me avise que no se cumple la condicion

2.7 Can you find any patterns or unusual clusters in your list of primes.

- ① A parte de 2 y 5, los últimos dígitos de los # primos tienden a terminar en 1, 3, 7, 9.
- ② Un número primo tiende a tener repetir su último dígito en el # primo predecesor
- ③ Orden inesperado
- ④ Hiperuniformidad: Porque desordenados los números primos son vistos en pequeños grupos de agrupaciones sin embargo tienden a ordenarse (de manera oculta) en grandes escamas de agrupaciones

2.8 Write a program to implement Alg. 2.4 Use it to factor or prove primality for:

307821, 16803654, 194685276691

```
var primeNumbers = [2, 3, 5, ...]
```

```
function trialDivision(n){  
    var primeFactors = "";  
    for(var i=0; p=primeNumbers[i]; i<primeNumbers.length  
        && p*p <= n; i++; primeNumbers[i])  
    {  
        while (n % p == 0)  
        {  
            primeFactors += " " + p + " ";  
            n /= p;  
        }  
        if (n<1)  
            primeFactors += " " + n + " ";  
    }  
    return primeFactors;  
}
```

Test

$$307821 = 3, 102607$$

$$16803654 = 2, 3, 7, 400087$$

$$194685276691 = 89, 89, 24578371$$

2.9 Choose 100 consecutive seven digits numbers and factors them using trial division.

1000001 a 1000100 → sera nuestro rango

Utilizamos el siguiente algoritmo para hallar los numeros que sean primos en el rango anterior

Para todo el rango la $\sqrt{n} = 1000$

int vectorPrimos[] = {2, 3, 5, 7, ..., \sqrt{n} } → Numeros primos

int vectorRango[] = {10000001, ..., 1000100} → Entre el rango

Código Java

```
public static void trialDivision(int[] num, int[] primos){  
    int aux=0;  
    for (int i=0; i<num.length; i++) {  
        for (int j=0; j<primos.length; j++) {  
            if (num[i] % primos[j] == 0) {  
                System.out.println("Número " + num[i] + " es compuesto");  
                aux = 1;  
                break;  
            }  
        }  
        if (aux == 0) {  
            System.out.println("El número " + num[i] + " es primo");  
        }  
        aux=0;  
    }  
}
```

```
public static void main (String[] args){
```

```
    int primos[] = {2, 3, 5, ...};
```

```
    int vectorRango[] = {1000001, 1000002, ...};
```

```
    trialDivision (vectorRango, primos);
```

J.

Ex 2.10 - 2.15 Use the factored numbers from G2.9.

- 2.10 How many primes are in your list of factored numbers? How does this compare with the expected number of primes?

los primos encontrados en nuestro rango son 6.

1 000003
1 000033
1 000037
1 000039
1 000081
1 000099

Sin hacer la prueba se podría pensar que al menos hay 15 # primos pero la realidad fue 6

R/H

- 2.11 How many perfect squares are in your list of factored numbers? How does this compare with the expected number of perfect squares?

Un número es un número perfecto

Con números tan grandes es complicado que sea un número perfecto

- 2.12 What is the distribution of the number of the distinct primes dividing each of integers? What is the mean and standard deviation of this distribution?

Su distribución 2, 3, 5, 7, 11, 13, 19, 23, 29, 31, 37, 41, 43... 839, 947
+
60 en total

Su media $2 + 3 + 5 + 7 + \dots + 947 = 212$

$$\text{La desviación es: } \sqrt{\frac{(12-212)^2 + (3-212)^2 + \dots + (947-212)^2}{60}}$$

Desviación = 222,37

2.13) What is the distribution of the sizes of the primes among your integers? How many of the integers n have a prime factor larger than $n^{3/4}$, $n^{1/2}$, $n^{1/3}$? Describe the distribution, mean and standard deviation of the logarithm of the largest prime factor divided by the logarithm of n .

- (2) 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 39, 43
49, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103
107, 109, 113, 131, 137, 149, 151, 157, 163, 179, 181
191, 229, 233, 281, 293, 313, 347, 353, 367, 373,
385, 397, 409, 435, 463, 601, 661, 673, 743, 839, 847

Estos resultados se obtuvieron modificando el código para guardar los primos que dividen a los números que elegimos para realizar el algoritmo.

Se uso un ArrayList para guardar estos sin que se repiten y en orden.

② Igual modificando el código los resultados son los siguientes:

$$\# \text{num} = 0 \text{ para } n = [100001, 1000100] \text{ y } n^{3/4}$$

ya que $n^{3/4} > 31622$ $n = [100001, 1000100]$ y $n^{7/2}$

$$\# \text{num} = 0 \quad n = [1000001, 1000100] \text{ y } n^{1/2}$$

ya que $n^{1/2} > 1000$

$$\# \text{num} = 2 \quad n = [1000001, 1000100] \text{ y } n^{1/3}$$

ya que $n^{1/3} > 100$

③ Modificando otra vez el código se obtiene que el número = 100032 tiene como divisor al número primo más grande en este caso

$$\frac{947}{\log(1000032)} = 157,83 \approx 158$$

$$\text{Distri} = [2, 75]$$

$$\text{media} = [40, 5]$$

$$\text{desviación} = 38,5$$

2.14

How many of your 100 integers have the number of distinct prime factors within one standard deviation of the mean AND the logarithm of n within one standard deviation of the mean?

① Usando los factores primos obtenidos en el ejercicio anterior parte 1) y modificando el algoritmo

$$\text{media} = 212,95$$

$$\text{desviación estandar} = 224,24$$

$$\forall n \in [1000001, 1000100] \wedge \dots$$

$$\dots \wedge n \neq [1, 1000005, 1000009, 1000013, \\ 1000032, 1000035, 1000042, 1000043, \\ 1000045, 1000054, 1000064, 1000075, \\ 1000077, 1000079, 1000080, 1000089, \\ 1000093] \wedge \# \text{ primo}$$

Entonces

$$100 - 6 - 16 = 78$$

R4

② Por el resultado del anterior ejercicio la desviación es igual a 38,5

Modificando el algoritmo otra vez se obtiene

$$np < 38,5 \text{ si son iguales a}$$

$$100 - 40 - 6 = 54$$

R5

FREDDY
ABAD

2.15 How large a number can "usually" be factored using trial division up to 5000? Try your trial division algorithm on 100 consecutive integers of this size and report your results. "Usually" should mean at least 75% of the time and not more than 95% of the time.

$$\sqrt{n} < 5000$$

$$n < (5000)^2$$

$$n < 25000000$$

↓

R11

se tiene que cumplir esto para cumplir la condición

public static void trialDivision ()

```
for (int i = 0; i < vectorRango.length; i++) {
    boolean entro = false;
    for (int j = 2; j < 5000; j++) {
        if (vectorRango[i] % j == 0) {
            entro = true;
            System.out.println(vectorRango[i] + " no es primo");
        }
    }
    if (!entro)
        System.out.println(vectorRango[i] + " es primo");
}
```

int vectorRango [j=120000001, 20000002...]

R11

2.16 Prove lemma 2.6 by induction on k

$$\text{Lema 2.6} \rightarrow x^k - 1 = (x-1)(1+x+x^2+\dots+x^{k-1})$$

(a) Sean $k=1$ entonces

$$x^1 - 1 = (x-1)(1)$$

$$x-1 = (x-1) \therefore \text{Cumple}$$

(b) Se asume que se cumple

$$x^k - 1 = (x-1)(1+x+x^2+\dots+x^{k-1}) \rightarrow \text{Hipótesis de Inducción}$$

(c) $k=k+1$

$$x^{k+1} - 1 = (x-1)(1+x+x^2+\dots+x^{k-1}+x^{k+1-1})$$

$$x^{k+1} - 1 = (x-1)(1+x+x^2+\dots+x^k) \rightarrow \text{Tesis}$$

(d) Demostración

$$x^k - 1 = (x-1)(1+x+x^2+\dots+x^{k-1})$$

$$(x^{k+1} - 1) = (x-1)(1+x+x^2+\dots+x^{k+1-1})$$

$$(x^{k+1} - 1) = (x-1)(1+x+x^2+\dots+x^k)$$

$$k+1 = k'$$

$$k = k' - 1$$

$$(x^{k+1} - 1) = (x-1)(1+x+x^2+\dots+x^{k+1-1})$$

La igualdad se cumple

2.17 In the proof Teo 2.7, where did we use the fact that the power of 2 is at least one? That is to say, why can't we use this proof to characterize odd perfect numbers?

Sea $m \rightarrow$ par perfecto $\rightarrow \exists n \text{ tal que } m = 2^{n-1}(2^n - 1)$
 $\wedge (2^n - 1) \text{ es primo}$

Sea $m = 2^a \times t$, $t \text{ es impar}$
 $a \geq 1$ ya que m es par

$$\begin{aligned} n &= 3 \\ m &= 2^3 \times (2^3 - 1) \\ m &= 4 \times 7 = 28 \quad \checkmark \quad \begin{array}{l} \text{numero} \\ \text{perfecto} \\ \text{par} \end{array} \end{aligned}$$

Todos los números perfectos conocidos son pares, se conocen 48 números perfectos

Para que m sea par los exponentes a deben ser 1.
 Ya que si $a \geq 0$, m resultaría impar y el producto de 2 impares es un # impar y no es perfecto.

Sin embargo hay conjeturas que afirman que hay infinitos números perfectos o que todos son pares. Pero no hay nada demostrado)

2.18 Write a program to implement Alg 2.9. Use it to check that $M(64)$ is prime but $M(63)$ is not.

Añadir static void **algoritmo2.9()** {

```

Double n = 13.0;
Double M = Math.pow(2, n) - 1;
Double S = 4.0;
for (int i = 2; i < n; i++) {
    S = (S * S - 2) % M;
}
if (S == 0) {
    System.out.println("El # M = " + M + " es primo");
} else {
    System.out.println("El # M = " + M + " no es primo");
}
    
```

y

2.19 Use trial division to factor $M(29)$. Can you find any patterns to or properties of the prime divisors of $M(11)$, $M(23)$ and $M(29)$?

Division Trial : para saber si un # es primo

Dividir primos $\leq \sqrt{n}$ entre el numero.

Sea $n = 29$

$$\sqrt{n} = 14,25 \rightarrow \text{primos menores } 2, 3, 5, 7, 11, 13$$

2	105,5
3	70,33
5	42,2
7	30,14
11	19,18
13	16,23

ya que ninguno divide al numero de manera uniforme es un primo
es decir
 $a \bmod \text{primos}[i] \neq 0$

Donde $\text{primos}[] = \{2, 3, 5, 11, 13\}$

Sea $n = 23$

$$\sqrt{n} = 5,38 \rightarrow \text{primos menores } 2, 3, 5$$

$$n = 23$$

$$\sqrt{n} = 4,79 \rightarrow \text{primos menores } 2, 3$$

$$n = 23$$

$$\sqrt{n} = 3,3 \rightarrow \text{primos menores } 2$$

• Ninguno de estos factores divide a n de manera uniforme

Para $n = 29, 23, 11$; los numeros 2 y 3 son consecutivos y compartidos por todos los n ademas 2 es el unico primo par.

Freddy L. A.
freddy.alvarado@udistrital.edu.co



La naturaleza es parte de tí! Cuidala!



2.29 For n less than or equal to 30, when is $2^n + 1$ a prime? Can you make a conjecture or when it will be prime? Try to prove or disprove your conjecture.

$$2^n + 1 ; n \leq 30$$

$$n \quad f(n) = 2^n + 1$$

*	1	3	✓
*	2	5	✓
3		9	✗
*	4	17	✓
5		33	✗
6		65	✗
*	7	129	✗
*	8	257	✓
9		513	✗
10		1025	✗
11		2049	✗
12		4097	✗
13		8193	✗
14		16385	✗
15		32769	✗
*	16	65537	✓
17		131073	✗
18		262145	✗
19		524289	✗
20		1048577	✗
21		2097153	✗
22		4194305	✗
23		8388609	✗
24		16777217	✗
25		33554433	✗
26		67108865	✗
*	27	134217729	✗
28		268435457	✗
29		536870913	✗
30		1073741825	✗

Para $n \leq 30 \wedge n \geq 1$, se cumple la condición de primicidad cuando n es múltiplo de 2 y "x" es el resultado de la potencia de 2^x , es decir

$2^x \rightarrow$	x	2^x
	1	2
	2	4
	3	8
	4	16