Trabajo propuesto del tema 3: Teoría de Números y Criptografía

Programas de ordenador

- Pueden hacerse en grupos de como mucho tres alumnos.
- No se tienen que hacer todos los ejercicios(aunque cuantos más se hagan mejor)
- Deberá entregarse todo el trabajo en papel (los códigos de los programas impresos en papel y en un disco para poder probarlos)
- Los programas de ordenador se puede desarrollar en C ó en Mathematica.
- Me podéis consultar todas las dudas que se vayan planteando durante su realización.
- Como estáis aprendiendo a programar, la entrega de los programas de ordenador se puede retrasar hasta el mes de junio, aunque si los entregas antes, pues mucho mejor.
- El trabajo se deberá entregar personalmente por todos los alumnos participantes en el mismo. Para ello, os pondréis en contacto mediante correo electrónico ó personalmente conmigo para ver en qué momento se puede realizar dicha entrega.

Ejercicio 1: (Algoritmo extendido de Euclides)

Hacer un programa que nos pida dos números naturales a y b y nos proporcione d el máximo común divisor de ambos y los coeficientes de la identidad de Bezout, es decir nos calcule x_0, y_0 enteros tal que $a\,x_0 + b\,y_0 = d$. Por ejemplo, si se introducen como entrada los números 3120 y 270, el programa debe devolver 30 como máximo común divisor y los números de la identidad de Bezout serían 2 y –23 ya que:

$30 = 3120\,(2) + 270\,(-23)$

Ejercicio 2: (Factorización)

Hacer un programa que nos pida un número natural y nos devuelva su factorización en números primos. Por ejemplo, si le damos como entrada el número 10500 debe darnos como salida la lista:

{2, 2, 3, 5, 5, 5, 7}.

Ejercicio 3: (Criba de Eratóstenes)

Hacer un programa que nos pida un número natural n y entonces nos realice la criba de Eratóstenes, es decir, se comenzará con una lista de naturales: {2, 3, 4, 5, 6, ..., n} (todos los números naturales entre 2 y n) y se comienza eligiendo el 2 y tachando de dos en dos; luego se coge el 3 (siguiente elemento de la lista que no ha sido tachado) y se vuelven a tachar de tres en tres; después se coge el 5 (siguiente elemento de la lista que no ha sido tachado aún) y se vuelven a tachar de cinco en cinco, ... así se va razonando con números menores ó iguales que $\sqrt{n}$. La salida del programa será la lista con los números de la lista original que no han sido nunca tachados.

Por ejemplo, si $n = 200$, la salida debe ser:

{2,3,5,7,11,13,17,19,23,29,31,37,41,43,47,53,59,61,67,71,73,79,83,89,97,101,103,107,109,113,127,131,137,139,149,151,157,163,167,173,179,181,191,193,197,199}

Observar que con este proceso se consigue la lista de números primos menores ó iguales que n.

Ejercicio 4: (Lista de primos)

Hacer un programa que nos pida un número natural n y que nos proporcione la lista de números primos inferiores o iguales a n. El ejercicio anterior, la criba de Eratóstenes lo conseguía, pero ahora vamos a hacerlo de forma diferente. Suponiendo que le damos un número $n \geq 2$, entonces la lista de primos se inicializa a $\{2\}$ y después se va probando con el siguiente número natural, en primer lugar será el 3, si no es divisible por ningún elemento de la lista de primos construida hasta ese momento, se añade a la lista de primos, en caso contrario, pasamos a trabajar con el siguiente número natural. El proceso consiste en ir probando si se añade ó no un número a la lista de primos construida hasta ese momento, mientras que dicho número sea menor o igual que $\sqrt{n}$.

Ejercicio 5: (Cifrado César)

En la página web de la asignatura puedes encontrar un programa hecho con Mathematica llamado cesar.nb que sirve para cifrar un mensaje con el método de Julio César visto en clase. En este programa se usa el alfabeto de 27 caracteres: desde el 0 correspondiente a la letra A hasta el 26 correspondiente a la letra Z. El mensaje de entrada es "CESAR", la clave privada es $k = 3$ y el mensaje cifrado es "FHVDU".

Modifica este programa para hacer justo el proceso contrario, es decir, si se le proporciona como entrada el mensaje cifrado debe dar como salida el mensaje original.

Con dicho programa, descifrar el mensaje cifrado: "GUDKQWFDF", sabiendo que la clave utilizada ha sido $k = 17$.

Nota: Si lo deseas, puedes hacer el programa en C en lugar de en Mathematica.

Ejercicio 6: (Cifrado afín)

En la página web de la asignatura puedes conseguir el programa Mathematica afin.nb que sirve para cifrar con el criptosistema afín un texto como: "ALARMA FUEGO EN EL CINE" con un alfabeto de 28 caracteres que incluyen las 27 letras del castellano (desde el 0 para la letra A hasta el 26 para la letra Z, y el 27 dedicado al símbolo * para los espacios en blanco, que obtiene el siguiente mensaje encriptado:

CLCEVCSBJRM*SRFSRLSXGFR.

Modifica el programa anterior para conseguir descifrar el mensaje:

CQZNJCB*XCQZQUJWB*QMQZ*WQÑKZJÑZUXPZIXJMQ

Sabiendo que se ha encriptado con el sistema afín usando $a = 19 \quad b = 17$

Nota: Si lo deseas, puedes hacer el programa en C en lugar de en Mathematica.

Ejercicio 7: (Cifrado RSA)

En la página Web de la asignatura puedes encontrar el programa RSA.nb que sirve para encriptar un mensaje con el método RSA con un alfabeto de 28 caracteres, como el del ejercicio anterior, y encriptando letra por letra. Modifica el programa para conseguir encriptar con el método RSA un mensaje agrupando las letras de tres en tres y usando como claves p=863, q= 1223 y e=39423. Comprueba el correcto funcionamiento del programa encriptando el mensaje "SOL" y viendo que se obtiene LSBX.

<u>Ejercicio 8</u>: (Cifrado de Vernam: secreto perfecto)

Esta técnica de cifrado fue desarrollado por el ingeniero Vernam en 1917 cuando era muy joven. Se empleó durante la segunda guerra mundial y durante la guerra fría.

La clave será una secuencia aleatoria de bits, tan larga como el mensaje original. El mensaje cifrado se consigue sumando bit a bit (suma en $\mathbb{Z}_2$ ó XOR) el mensaje original con la clave. El receptor descifra el mensaje sumando bit a bit el mensaje recibido con exactamente la misma clave. Por lo tanto, la clave es la misma tanto para cifrar como para descifrar.

Por ejemplo, si usamos un alfabeto de 64 caracteres:

#	A	B	C	D	E	F	G	H	I	J	K	L	M	N	Ñ	O	P	Q	R	S	T	U	V	W	X	Y	Z	Ç
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28

0	1	2	3	4	5	6	7	8	9	@	(	)	{	}	<	>	=	+	_	*	/	%	&	º
28	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53

ª	,	;	.	:	¿	?	¡	!	‘
54	55	56	57	58	59	60	61	62	63

Donde el carácter # lo dedicamos a los espacios en blanco.

Entonces, el mensaje encriptado

A1OFKGDYGFÑUVÑJOX1P88Ç3O1QNYKP¡7QMN(IHQN

con la clave

M210JJM2IR2URJJ0OJW)3H3P11ÑNKÑ11ÑON3XZRY

se corresponde con el mensaje: LANZAMIENTO DE MISILES A LAS 16:00 HORAS

Veámoslo sólo con los dos primeros símbolos:

- Como el alfabeto tiene 64 símbolos, hacen falta 6 bits para cada símbolo.
- El primer símbolo del mensaje “A” se corresponde, según el alfabeto, con el 1 que en binario es: 000001
- El segundo símbolo del mensaje es “1” que se corresponde, según el alfabeto, con el 30 que en binario es: 011110 (la orden IntegerDigits[30,2,6] de Mathematica devuelve una cadena con los 6 dígitos de 30 escrito en base 2)
- El primer símbolo de la clave es “M”, que se corresponde con 13, que en binario es: 001101
- El segundo símbolo de la clave es “2”, que se corresponde con el 31, que en binario es: 011111
- Hacemos XOR ó suma en $\mathbb{Z}_2$:

$$\left.\begin{matrix}\text{mensaje} = 000001 \quad 011110\\ \text{clave} = \quad 001101 \quad 011111\end{matrix}\right\}$$

$$\text{resultado} = \quad 001100 \quad 000001$$

- Los primeros 6 bits del resultado: 001100 se corresponden con el 12 que es la letra "L" (la orden de Mathematica FromDigits[{0,0,1,1,0,0},2] devuelve 12). Análogamente, los siguientes 6 bits 000001 se corresponde con el 1 que es la letra "A".
- Conclusión: el descifrado de los dos primeros símbolos es "LA". De igual forma se hace con el resto.

Pues bien, el mismo mensaje cifrado pero descifrado con la clave:

ÑNOTN#EIGJNBV1FPVKO*8P)TOG0Z2P)4R,MJ>ILZÑ

se corresponde con el mensaje:

NO REGAR LAS PLANTAS MIENTRAS SEA DE DIA

El mismo mensaje puede dar lugar a cualquier frase. Sólo depende de la clave que empleemos en el descifrado.

Desarrollar un programa que lleve a cabo el descifrado completo que antes dejamos sin terminar. Averiguar también cuál debe ser la clave de descifrado para que al descifrar el mismo mensaje se obtenga, por ejemplo, la frase:

HACE UN DIA MAGNIFICO PARA IR DE PASEO

Observa que deberás añadir dos espacios en blanco (el carácter #) para conseguir que tenga tantos símbolos como el mensaje (40 en total)

Por este motivo, se dice que esta técnica de cifrado tiene secreto perfecto. Aunque se intercepte el mensaje cifrado, es imposible descifrarlo. La clave debe usarse solamente una vez, one time pad, ya que un criptonalista puede encontrar coincidencias entre varios mensajes cifrados con la misma clave. El nombre se debe a que las claves se escribían en un cuaderno (pad). En cada página se escribía una clave y cuando se usaba una, se arrancaba la página correspondiente (one time pad).

Ejercicio 9: Exponenciación modular rápida

Existe un método muy eficiente para el cálculo de las potencias en $\mathbb{Z}_n$. Éste método se basa en escribir en binario el exponente de dicha potencia. En las direcciones:

http://www.dma.fi.upm.es/java/matematicadiscreta/Aritmeticamodular/congruencias.html

http://es.wikipedia.org/wiki/Exponenciación_binaria

http://it.aut.uah.es/enrique/docencia/ii/seguridad/documentos/t4-0405.pdf (pág. 23)

puedes encontrar la idea de este método que se conoce también como "exponenciación modular rápida ó exponenciación binaria".

Haz un programa que implemente este método.