PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

ALGORITMIA
Examen 1
(Primer semestre 2016)

Indicaciones generales:

- Duración: 3h.
- Materiales o equipos a utilizar: Apuntes de clase personales escritos a mano.
- Al inicio de cada programa, el alumno deberá incluir, a modo de comentario, la estrategia que utilizará para resolver el problema. De no incluirse dicho comentario o si la implementación es significativamente diferente a la estrategia indicada, el alumno no obtendrá el puntaje completo en dicha pregunta.
- Un programa que no muestre resultados coherentes y/o útiles será corregido sobre el 60 % del puntaje asignado a dicha pregunta.
- **Debe utilizar comentarios para explicar la lógica seguida en el programa elaborado.**
- El orden será considerado dentro de la evaluación.
- Cada programa debe ser guardado en un archivo con el nombre *preg#codigo_de_alumno.c* y subido a PAIDEIA en el espacio indicado por los Jefes de Práctica.
- **La presentación, la ortografía y la gramática de los trabajos influirá en la calificación.**

Puntaje total: 20 puntos

Cuestionario:

Pregunta 1 (7 puntos)

Se tiene un arreglo de enteros A que es infinito. Existen N elementos mayores a cero almacenados en este arreglo, los cuales están contiguamente almacenados desde la primera posición. A partir de la posición $N + 1$ el arreglo contiene solo ceros. El problema consiste en encontrar N usando un algoritmo de complejidad O(log N). Diseñe un algoritmo en pseudocódigo para encontrar el valor N, dado el arreglo infinito A. Presente un ejemplo que demuestre que su algoritmo funciona y explique por qué cumple con la eficiencia solicitada.

Pregunta 2 (7 puntos)

A John le gusta bucear y cazar tesoros. Para su buena suerte, él acaba de encontrar la ubicación de un barco pirata lleno de tesoros. El sofisticado sistema sonar a bordo de su barco le permite identificar la ubicación, profundidad y cantidad de oro en cada tesoro hundido. Desafortunadamente, John olvidó traer un dispositivo GPS y las posibilidades de encontrar esta ubicación de nuevo son muy escasas, por lo que tiene que apropiarse del oro ahora. Para empeorar la situación, John solo tiene un tanque de aire comprimido.

John quiere bucear con el tanque de aire comprimido para recuperar tanto oro como sea posible del naufragio. Escriba un programa que John pueda usar para seleccionar cuáles tesoros debe recoger para maximizar la cantidad de oro recuperado.

El problema tiene las siguientes restricciones:

- Existen n tesoros $\{(d_1, v_1), (d_2, v_2), ..., (d_n, v_n)\}$ representados por el par (profundidad, cantidad de oro). Existen como máximo 30 tesoros.
- El tanque de aire solo permite estar t segundos bajo el agua. t es como máximo 1000 segundos.
- En cada inmersión, John puede traer como máximo un tesoro a la vez.
- El tiempo para descender es $td_i = w * d_i$, donde w es un número entero constante.
- El tiempo para ascender es $ta_i = 2w * d_i$, donde w es un número entero constante.
- Debido a limitaciones de los instrumentos, todos los parámetros son números enteros.

Entrada: La entrada para este programa consiste de una secuencia de valores enteros. La entrada contiene varios casos de prueba. La primera línea de cada conjunto de prueba contiene los valores t y w. La segunda línea contiene el número de tesoros. Cada una de las siguientes líneas contiene la profundidad d_i y la cantidad de oro v_i de un tesoro diferente. Una línea en blanco separa cada caso de prueba.

Salida: La primera línea de la salida para cada conjunto de prueba debe contener la máxima cantidad de oro que John puede recoger del naufragio. La segunda línea debe contener el número de tesoros recuperados. Cada una de las siguientes líneas debe contener la profundidad y la cantidad de oro de cada tesoro recuperado. Los tesoros deben ser presentados en el mismo orden que en la entrada. Imprima una línea en blanco entre las salidas de diferentes conjuntos de prueba.

Ejemplo de Entrada:	**Ejemplo de Salida:**
210 4 3 10 5 10 1 7 2	7 2 10 5 7 2

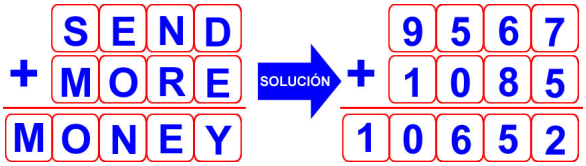
En este ejemplo de entrada, el tanque de aire comprimido tiene un capacidad de 210 segundos, la constante w tiene el valor de 4 y existen 3 tesoros. El primero está a una profundidad de 10 metros y vale 5 monedas de oro, el segundo está a una profundidad de 10 metros y vale 1 moneda de oro, y el tercero está a una profundidad de 7 metros y vale 2 monedas de oro.

PARTE ELECTIVA

Resuelva **UNA** de las preguntas que se presentan a continuación usando la estrategia **Backtracking**.

Pregunta 3 (6 puntos) Criptoaritmética

Periódicos y revistas a menudo tienen ejercicios de criptoaritmética de la forma:



El objetivo es asignar a cada letra un dígito del 0 al 9 de tal forma que la aritmética funcione correctamente. Las reglas son que a todas las ocurrencias de un letra se les debe asignar el mismo dígito, y ningún dígito puede ser asignado a más de una letra. Implemente un programa en C que reciba como parámetros las letras que conforman cada sumando y la suma, y que imprima el valor que debe tener cada letra. Usted defina el formato de la entrada y de la salida.

Pregunta 4 (6 puntos) Distancia de Hamming

La distancia de Hamming entre dos cadenas de bits es el número de posiciones de bits correspondientes que difieren. Esta distancia puede ser encontrada usando la función lógica XOR sobre los bits correspondientes. Por ejemplo:

A	0 1 0 0 1 0 1 0 0 0
B	1 1 0 1 0 1 0 1 0 0
A XOR B =	1 0 0 1 1 1 1 1 0 0

La distancia Hamming entre estas dos cadenas de bits de 10 elementos es 6, el número de 1's en la cadena resultante del XOR. Escribir un programa en C que imprima todas las cadenas de bits de longitud N que tienen una distancia Hamming H con respecto a la cadena de bits que contiene puros ceros.

Entrada: La entrada consiste de diferentes conjuntos de datos. La primera línea de la entrada contiene el número de conjuntos de datos y es seguida por una línea en blanco. Cada conjunto de datos contiene N (la longitud de las cadenas de bits) y H (la distancia Hamming) en la misma línea. Existe una línea en blanco entre los diferentes casos de test.

Salida: Para cada conjunto de datos, imprimir una lista de todas las posibles cadenas de bits de longitud N que tienen una distancia Hamming H con respecto a la cadena de bits que contiene sólo ceros. Es decir, imprimir todas las cadenas de bits de longitud N con exactamente H unos impresos en orden lexicográfico ascendente.

Ejemplo de Entrada:	**Ejemplo de Salida:**
1 4 2	0 0 1 1 0 1 0 1 0 1 1 0 1 0 0 1 1 0 1 0 1 1 0 0

Profesores del curso: Fernando Alva
Iván Sipirán

Pando, 14 de mayo de 2016