PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

ALGORITMIA
Examen 1
(Primer semestre 2015)

Indicaciones generales:

- Duración: 2h 55 min.
- Materiales o equipos a utilizar: No se permite el uso de material de consulta.
- 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 perderá el derecho a reclamo en esa 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.**
- 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.

Puntaje total: 20 puntos

Cuestionario:

Pregunta 1 (5 puntos) Sorting

Indique el nombre o número de equipo en el que realizó el primer trabajo grupal.

Pregunta 2 (5 puntos) Searching in Rotated Array

Dado un vector $V[1..n]$ y un número natural k entre 1 y $n-1$, rotar un arreglo k veces significa que los k primeros elementos de V han sido transpuestos con los elementos en las $n-k$ últimas posiciones. Por ejemplo, si V es el siguiente vector:

a b c d e f g h i j

y se ha rotado $k = 3$ veces, se obtendría el siguiente resultado:

d e f g h i j a b c

Dado un vector ordenado de n números enteros que ha sido rotado un número desconocido de veces, diseñe un algoritmo que permita encontrar un elemento en dicho vector de forma **eficiente**. Puede asumir que el vector estaba originalmente ordenado en orden ascendente y que no hay repeticiones de números.

Exprese el algoritmo en pseudocódigo o lenguaje natural y mediante un ejemplo demuestre que cumple con la tarea planteada. Además, indique y justifique adecuadamente cuál es la eficiencia del algoritmo.

Pregunta 3 (5 puntos) UVa 147 - Dollars (Traducción Libre)

La distribución de monedas de Nueva Zelanda se compone de $100, $50, $20, $10 y $5 en billetes y $2, $1, 50c, 20c, 10c y 5c en monedas. Escriba un programa que calcule, para cualquier cantidad dada, el número de formas distintas en que se puede componer dicha cantidad usando billetes y monedas. Cambiar el orden de la distribución no aumenta el conteo. De este modo, 20c puede ser expresado de 4 maneras diferentes: 1 x 20c, 2 x 10c, 10c + 2 x 5c y 4 x 5c.

Entrada: La entrada consiste en una serie de números reales no mayor de los $300.00 cada uno en líneas separadas. Las cantidades ingresadas serán válidas; es decir, debe ser un múltiplo de 5c. La lectura de números terminará cuando se ingresa una línea con el valor cero (0.00).

Salida: La salida consistirá en una línea por cada monto ingresado. Cada línea está compuesta por dos valores: el monto de dinero con dos decimales y el número de maneras diferentes en el que ese monto puede componerse.

Ejemplo de Entrada:	**Ejemplo de Salida:**
0.20 2.00 0.00	0.20 4 2.00 293

Implemente un programa en C que resuelva el problema descrito.

PARTE ELECTIVA

Para **UNA** de las preguntas que se presentan a continuación, elabore un programa en C que resuelva el problema descrito.

Pregunta 4 (5 puntos) Laberinto del Minotauro

Teseo ingresa a un laberinto en busca de un minotauro que no sabe dónde está. Se trata de implementar una función **ariadna** que le ayude a encontrar el minotauro y a salir después del laberinto. El laberinto debe representarse como una matriz de entrada a la función cuyas casillas contienen uno de los siguientes tres valores: 0 para "camino libre", 1 para "pared" (no se puede ocupar) y 2 para "minotauro". Teseo puede tomar la dirección Norte, Sur, Este u Oeste siempre que no haya una pared. La función **ariadna** debe devolver la secuencia de casillas que componen el camino de regreso desde la casilla ocupada por el minotauro hasta la casilla $(1, 1)$.

Pregunta 5 (5 puntos) Maximum Sum Increasing Subsequence

Given an array of n positive integers, we want to find the sum of the maximum sum subsequence of the given array such that the integers in the subsequence are sorted in increasing order. For example, if input is {1, 101, 2, 3, 100, 4, 5}, then output should be 106 (1 + 2 + 3 + 100), if the input array is {3, 4, 5, 10}, then output should be 22 (3 + 4 + 5 + 10) and if the input array is {10, 5, 4, 3}, then output should be 10. Your program should print the maximum sum as well as the numbers that compose the subsequence.

Profesores del curso: Fernando Alva
Robert Ormeño

Pando, 16 de mayo de 2015