Este es el primer examen parcial del curso *Programación Imperativa Modular (PIMO)*, 2015-1. El examen tiene 4 preguntas; otorga un total de 50 puntos y 30 de bono. El examen es *individual* y no es permitido el uso de libros, apuntes ni equipos electrónicos.

Nombre y código: ____

Pregunta	Invariantes	Dividir y conquistar	Orden asintótico y el Teorema Maestro	Programación dinámica	Total
Puntos	15	15	10	10	50
Bono	0	10	10	10	30
Puntaje					

Pregunta 1: Invariantes . . . . (15 puntos)

Considere la siguiente especificación:

Entrada: un arreglo $a[0..N)$ de números, $N \geq 0$.

Salida: el valor del producto $(\times i \mid 0 \leq i < N : a[i])$

El siguiente fragmento de código Python implementa una solución del problema anterior:

```
1 def prod(a):
2   N = len(a)
3   r,n = 1,0
4   while n != N:
5     r,n = r*a[n],n+1
6   return r
```

(a) (2 puntos) Calcule, exhibiendo una simulación, el valor de `prod([5,4,1,6])`.

(b) (3 puntos) En términos de notación O, ¿cuál es el tiempo de ejecución de la función `prod`? Puede suponer que las operaciones aritméticas y de asignación toman tiempo $O(1)$. Justifique su respuesta.

(c) (10 puntos) Considere los siguientes invariantes para el ciclo `while` en las líneas 4–5:

P_0**:** $0 \leq n \leq N$.

P_1**:** $r = (\times i \mid 0 \leq i < n : a[i])$.

Demuestre las propiedades de iniciación, estabilidad y terminación para P_0 y P_1. Concluya que, al momento de terminar, la función `prod(a)` calcula el producto de los números en `a`.

Pregunta 2: Dividir y conquistar . . . . (15 puntos)

Considere un arreglo creciente $a[0..N)$ de números, con $N \geq 0$, y un número x.

(a) (3 puntos) Especifique el problema de encontrar un índice i en a tal que $a[i] = x$; si x no es un valor en a, entonces $i = -1$.

(b) (5 puntos) Diseñe un algoritmo que resuelva el problema especificado en el numeral (a) en orden temporal $O(\log N)$ (esto debe demostrarlo).

(c) (7 puntos) Diseñe un algoritmo que resuelva el problema especificado en el numeral (a) y que garantice que si x aparece en a, entonces la respuesta i corresponde al primer índice tal que $a[i] = x$, en orden temporal $O(\log N)$ (esto debe demostrarlo).

(d) (10 +) Diseñe un algoritmo que determine la cantidad de ocurrencias de x en a en orden temporal $O(\log N)$.

Pregunta 3: Orden asintótico y el Teorema Maestro . . . . (10 puntos)

(a) (2 puntos) Sea f una función de los números naturales en los reales positivos. Defina $O(f)$.

(b) (2 puntos) Enuncie el Teorema Maestro.

(c) (3 puntos) Clasifique ascendentemente por orden asintótico las siguientes funciones:

$$\sqrt{\frac{n^3}{2}} \qquad \log n^n \qquad 10n^2 + 1000n \qquad 2^{log\, n} \qquad \log\ n^5$$

(d) (3 puntos) Use el Teorema Maestro para calcular $T(n)$ en cada uno de los siguientes casos:

i. $T(n) = 2T(n/2) + 1$

ii. $T(n) = T(5n/3) + n$

iii. $T(n) = 3T(n/4) + 2n^2$

(e) (10 +) Sean f_1, f_2, g_1, g_2 funciones de los números naturales en los números reales positivos. Demuestre que si $f_1 \in O(g_1)$ y $f_2 \in O(g_2)$, entonces $f_1 \cdot f_2 \in O(g_1 \cdot g_2)$.

Pregunta 4: Programación dinámica . . . . (10 puntos)

El problema de *cambio de moneda* es el problema de determinar la cantidad de formas de obtener un valor usando denominaciones de ciertos billetes.

Entrada: un arreglo $a[0..N)$ de números enteros positivos distintos, con $N \geq 0$, y un número $V \geq 0$.

Salida: cantidad de formas de obtener V usando denominaciones de billetes en $a[0..N)$.

También considere la siguiente función $\rho : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$:

$\rho(n, v)$: "cantidad de formas de obtener v usando denominaciones de billetes en $a[0..n)$."

con $0 \leq v \leq V$ y $0 \leq n \leq N$, y definida recurrentemente de la siguiente manera:

$$\rho(n, v) = \begin{cases} 1 & \text{, si } n = 0 \land v = 0 \\ 0 & \text{, si } n = 0 \land v \neq 0 \\ \rho(n-1, v) & \text{, si } n > 0 \land a[n-1] > v \\ \rho(n-1, v) + \rho(n, v - a[n-1]) & \text{, si } n > 0 \land a[n-1] \leq v \end{cases}$$

(a) (2 puntos) Escriba el objetivo del problema planteado inicialmente en términos de ρ.

(b) (2 puntos) Aplique la función ρ, con los parámetros adecuados, para calcular cantidad de formas de obtener 10 usando denominaciones de billetes en $[1, 2, 5]$.

(c) (6 puntos) Diseñe una tabulación para la función ρ y un algoritmo que la procese.

(d) (10 +) Diseñe una tabulación para la función ρ y un algoritmo que la procese, de manera tal que la complejidad temporal del algoritmo sea $O(N \cdot V)$ y la complejidad espacial sea $O(V)$.