UNIVERSIDAD DE COSTA RICA

Escuela Ciencias de la Computación e Informática

CI-1221 Estructuras de Datos y Análisis de Algoritmos
II ciclo 2013, grupo 3

I Examen Parcial

Viernes 20 de setiembre

Nombre: ______________________________ Carné: ______________________________

El examen consta de 9 preguntas que suman 145 puntos, pero no se reconocerán más de 110 (10 % extra). Cada pregunta indica el tema tratado y su valor. Si la pregunta tiene subítemes, el puntaje de cada uno es indicado en los subítemes. Se recomienda echar un vistazo a los temas de las preguntas y a su puntaje antes de resolver el examen, para así distribuir su tiempo y esfuerzo de la mejor manera.

Las preguntas se pueden responder en cualquier orden, pero se debe indicar en la tabla mostrada abajo los números de página del cuaderno de examen en la que están las respuestas. Para esto debe numerar las hojas del cuaderno de examen en la esquina superior externa de cada página. Si la respuesta está en el enunciado del examen favor indicarlo con la letra E en vez del número de página.

Pregunta	Puntos	Páginas	Calificación
1. *Algoritmos iterativos*	40		
2. *Algoritmos recursivos e iterativos*	52		
3. *Solución de recurrencias*	20		
4. *Ordenamiento por selección*	5		
5. *Ordenamiento por inserción*	5		
6. *Ordenamiento por mezcla*	5		
7. *Ordenamiento por montículos*	8		
8. *Ordenamiento rápido*	5		
9. *Ordenamiento por residuos*	5		
Total	145		

El examen se puede realizar con lápiz o lapicero. *No se permite el uso de dispositivos electrónicos (calculadoras, teléfonos, audífonos, etc.).*

Las preguntas 4 en adelante están inspiradas en el partido de fútbol de las selecciones nacionales de Costa Rica y EE. UU. efectuado el 6 de setiembre de 2013, partido clave en la clasificación de

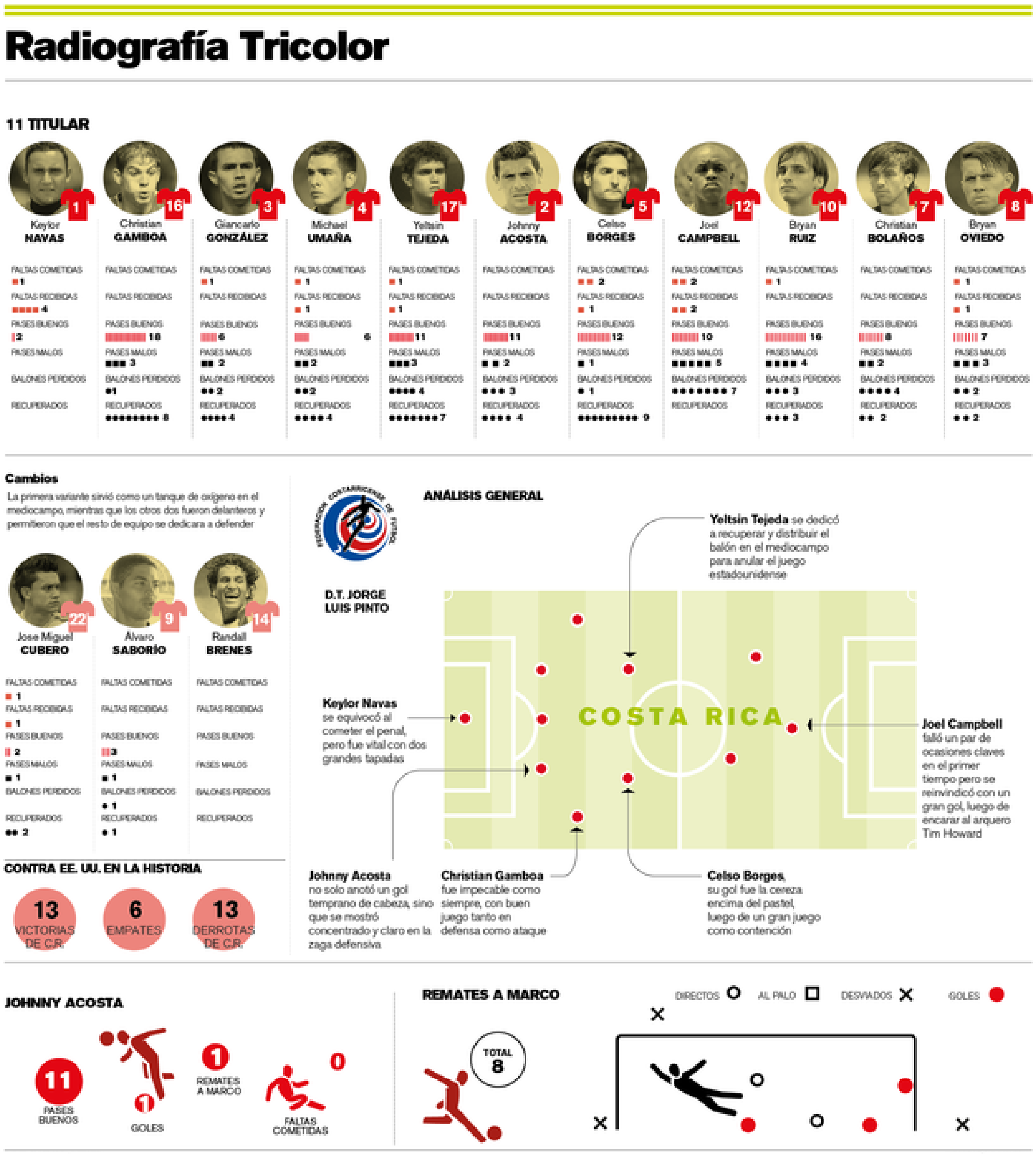


Figura 1: Tomada del periódico la Nación del 7 de setiembre de 2013.

la selección de Costa Rica al Campeonato Mundial Brasil 2014. Específicamente, el examen cita los jugadores más destacados del partido, según el periódico la Nación (ver figura 1).

1. *Algoritmos iterativos.* [40 pts.]
   Sea el polinomio
   $$P(x) = \sum_{k=0}^{n} a_k x^k.$$
   a) Escriba un algoritmo en C que evalúe $P(x)$ de la forma *directa*, es decir, calculando explícitamente cada término de la suma y acumulándolo en un total. Utilice el siguiente encabezado: [10 pts.]
   ```
   float P( float a[], int n, float x );
   ```
   b) Establezca la correctitud del algoritmo identificando un invariante apropiado para el ciclo principal del algoritmo [5 pts.] y mostrando cómo los pasos de inicialización [5 pts.], mantenimiento [5 pts] y terminación [2 pts.] permiten determinar la correctitud del algoritmo.
   c) Asumiendo que una asignación toma un tiempo t_a, una comparación t_c, un incremento o decremento t_i, una suma t_s y una multiplicación t_m, escriba una fórmula para el tiempo de ejecución del algoritmo [8 pts.] y determine una cota asintótica lo más precisa posible para el tiempo de ejecución [5 pts.].

2. *Algoritmos recursivos e iterativos.* [52 pts.]
   La regla de Horner calcula el polinomio
   $$P(x) = \sum_{k=0}^{n} a_k x^k$$
   por medio de la siguiente identidad:
   $$P(x) = a_0 + x(a_1 + x(a_2 + \cdots + x(a_{n-1} + x a_n) \cdots)).$$
   a) Escriba un algoritmo *recursivo* en C que implemente la regla de Horner. Use el siguiente encabezado: [10 pts.]
   ```
   float HornerRecursivo( float a[], int n, float x );
   ```
   Si lo necesita, puede crear una subrutina.
   b) Escriba una recurrencia para el tiempo de ejecución del algoritmo [5 pts.] y resuélvala [5 pts.].
   c) Escriba un algoritmo *iterativo* en C que implemente la regla de Horner usando el siguiente encabezado: [10 pts.]
   ```
   float HornerIterativo( float a[], int n, float x );
   ```
   d) Establezca la correctitud del algoritmo *iterativo* identificando un invariante apropiado para el ciclo principal del algoritmo [5 pts.] y mostrando cómo los pasos de inicialización [5 pts.], mantenimiento [5 pts] y terminación [2 pts.] permiten determinar su correctitud.
   e) Asumiendo que una asignación toma un tiempo t_a, una comparación t_c, un incremento o decremento t_i, una suma t_s y una multiplicación t_m, escriba una fórmula para el tiempo de ejecución del algoritmo en c) [5 pts.] y determine una cota asintótica lo más precisa posible para este tiempo [5 pts.]

3. *Solución de recurrencias.* [20 pts.]
   Resuelva las siguientes recurrencias asumiendo que $T(n) = \Theta(1)$ para $n \leq 1$ y que $k > 1$.
   a) $T(n) = k^2\, T(n/k) + k$. [7 pts.]
   b) $T(n) = T(n-1) + n$. [13 pts.]

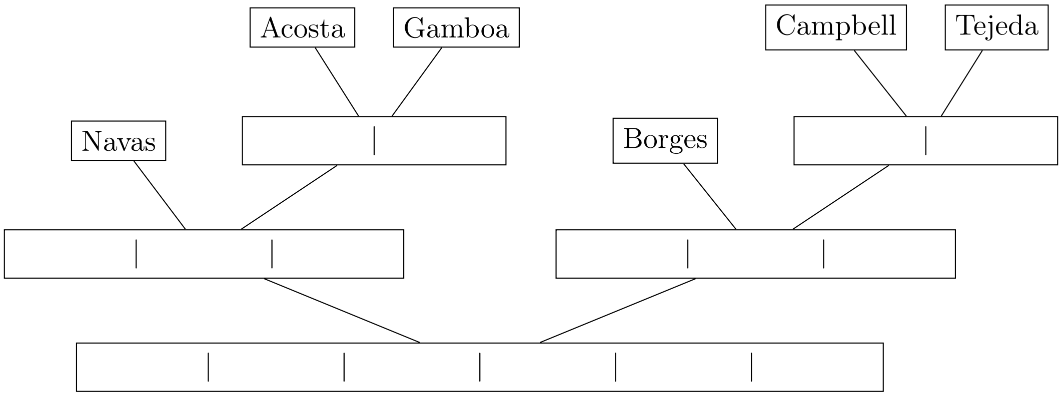
4. *Ordenamiento por selección.* [5 pts.]
   Simule la ejecución del algoritmo de ordenamiento por selección sobre el arreglo mostrado abajo. Muestre el estado del arreglo al finalizar cada una de las iteraciones del ciclo principal (externo). Si no muestra el estado de una casilla se asume que conserva el mismo valor que en la iteración anterior. Después de la primer inserción incorrecta el resto de inserciones no suman puntos.

	Posición					
Iteración	1	2	3	4	5	6
0	Navas	Acosta	Gamboa	Borges	Campbell	Tejeda
1						
2						
3						
4						
5						

5. *Ordenamiento por inserción.* [5 pts.]
   Simule la ejecución del algoritmo de ordenamiento por inserción sobre el arreglo mostrado abajo. Muestre el estado del arreglo al finalizar cada una de las iteraciones del ciclo principal (externo). Si no muestra el estado de una casilla se asume que conserva el mismo valor que en la iteración anterior. Después de la primer inserción incorrecta el resto de inserciones no suman puntos.

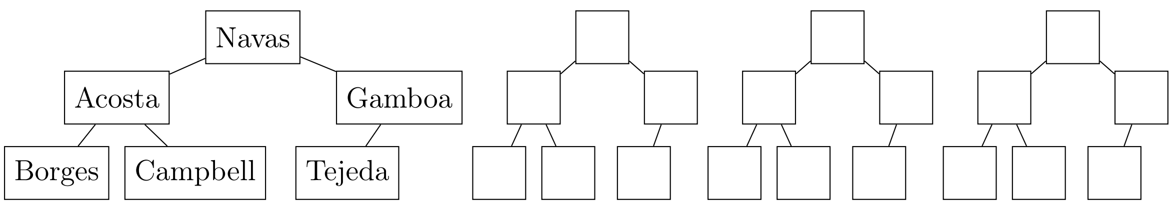
	Posición					
Iteración	1	2	3	4	5	6
1	Navas	Acosta	Gamboa	Borges	Campbell	Tejeda
2						
3						
4						
5						
6						

6. *Ordenamiento por mezcla.* [5 pts.]
   Simule la ejecución del algoritmo de ordenamiento por mezcla sobre el arreglo mostrado abajo. Muestre el estado del arreglo al finalizar cada una de las mezclas. Si no muestra el estado de una casilla se asume que conserva el mismo valor que en la iteración anterior. Si el tamaño del (sub) arreglo es impar, *redondee* el punto medio hacia abajo, es decir, la parte izquierda del subarreglo debe ser más pequeña que la parte derecha. Por ejemplo, un arreglo de tamaño 3 debe ser partido en un arreglo de tamaño 1 (a la izquierda) y uno de tamaño 2 (a la derecha). Después de la primer mezcla incorrecta el resto de mezclas no suman puntos.

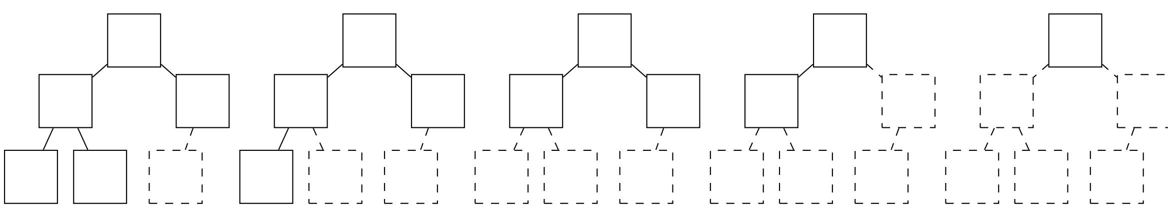


7. *Ordenamiento por montículos.* [8 pts.]
   Simule la ejecución del algoritmo de ordenamiento por montículos sobre el arreglo mostrado abajo en forma de árbol. Muestre el estado del *montículo* después de cada llamado a Corregir-Cima y Extraer-Máximo. (Los nodos con línea discontinua los puede dejar vacíos si lo desea). Después de la primer operación incorrecta el resto de operaciones no suman puntos.

   Monticularizar:



   Ordenar:



8. *Ordenamiento rápido.* [5 pts.]
   Simule la ejecución del algoritmo de ordenamiento rápido sobre el arreglo mostrado abajo. Muestre el estado del subarreglo a ordenar después de cada llamado a Partición e indique la posición del pivote devuelto. Si no muestra el estado de una casilla se asume que conserva el mismo valor que en la iteración anterior. Después del primer estado incorrecto el resto de estados no suman puntos.

Llamado	Estado del (sub)arreglo					
	1	2	3	4	5	6
	Navas	Acosta	Gamboa	Borges	Campbell	Tejeda
1.º						
2.º						
3.º						
4.º						
5.º						

9. *Ordenamiento por residuos.* [5 pts.]
Simule la ejecución del algoritmo de ordenamiento por residuos sobre las iniciales de los jugadores de la selección de Costa Rica destacados por la Nación en la figura 1. Muestre lo siguiente para cada llamado a la subrutina de ordenamiento por conteo:

   I. El histograma C. [2½ pts.]
   II. El histograma acumulativo C'. [2½ pts.]
   III. El arreglo resultante B. [2½ pts.]
   IV. El estado final del histograma acumulativo C'' (después de producir el arreglo B). [2½ pts.]

   Después del primer arreglo incorrecto el resto de arreglos no suman puntos.

	A
1	KN
2	JA
3	CG
4	CB
5	JC
6	YT

C:

A	B	C	G	N	T

C':

A	B	C	G	N	T

C'':

A	B	C	G	N	T

	$B \to A$
1	
2	
3	
4	
5	
6	

C:

C	J	K	Y

C':

C	J	K	Y

C'':

C	J	K	Y

	B
1	
2	
3	
4	
5	
6	