CÓDIGO

2 0 2 0 1 0 3 1 1

NOMBRE DEL ALUMNO

MATOS	CANGALAYA	JEREMY JEFFREY
APELLIDO PATERNO	APELLIDO MATERNO	NOMBRES

CARRERA SECCIÓN

CS 1

FIRMA

PRÁCTICA CALIFICADA DEL CURSO:

ADA Nº 2

NOTA

15.5

NOMBRE DEL PROFESOR

JUAN GUTIERREZ ALVA

FIRMA

IMPORTANTE

- El alumno deberá completar los datos indicados en la carátula antes de iniciar la práctica.
- La práctica deberá ser desarrollada únicamente con lapicero negro o azul.
- El trabajo en limpio se desarrollará en la cara derecha de cada hoja, el lado izquierdo podrá ser usado como borrador.
- Se considerará importante para la calificación el orden, la ortografía y limpieza de la práctica.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

E-1

a)

```
HEAP-EXTRACT-MAX(A)                         lg(n)
  if A.size == 0
    error "empty heap"
  n = A.size
  exchange A[1] with A[n]                   1/1.5
  A.size = A.size - 1
  MAX-HEAPIFY(A,1)
  return max
```

```
HEAP-INCREASE-KEY (A, i, Key)               lg(n)
  if Key < A[i]
    error "Key is lesser than current value"
  A[i] = Key
  while i > 1 and A[PARENT(i)] < A[i]       1.5/1.5
    exchange A[PARENT(i)] with A[i]
    i = PARENT(i)
```

b)

```
HEAP-DELETE (A, i)
  if i > A.size or i < 1
    error "index out of bounds"
  HEAP-INCREASE-KEY (A, i, INF)        lg(n)     2/2
  HEAP-EXTRACT-MAX(A)                  lg(n)
```

E-2

```
ALGO(n)
1:  K ← RANDOM(n)
2:  for i=1 to K
3:    cont = cont + 1
4:  return cont
```

a) Sea X la variables aleatoria que cuenta el # veces que se ejecuta la l.3 del algoritmo

$$E[X] = \sum_{x \in X} x \cdot Pr\{X = x\} = \sum_{i=1}^{n} i \cdot Pr\{X = i\}$$

0/2

$X = RANDOM(n) = \{1, 2, \ldots, n\}$

b) Para cada $i \in \{1, 2, \ldots, n\}$ se define la v.i. X_i según:

$$X_i = \begin{cases} 1: & \text{la línea 3 se ejecutó en la iteración } i \\ 0: & \text{caso contrario} \end{cases}$$

$$\begin{aligned} E[X_i] = 1 \cdot Pr\{X_i = 1\} + \cancel{0 \cdot Pr\{X_i = 0\}} &= 1 \cdot Pr\{X_i = 1\} \\ &= Pr\{X_i = 1\} \\ &= Pr\{\text{la línea 3 se ejecute en la iteración } i\} \\ &= Pr\{i \le K\} \\ &= K/n \end{aligned}$$

$\underbrace{1 \ldots K}_{i} \ldots n$

1/2

$$E[X] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} \frac{K}{n} = \frac{K}{n} \frac{n(n+1)}{2} = \frac{K(n+1)}{2}$$

0/1

E-3

(b) QS([1, 2, 3, 4, 7, 6, 5]) n-1 comp. → 6 comp.

$\underbrace{\underbrace{\underbrace{1}_{1comp.}\ 2}_{2comp}\ 3\ 4}_{3comp}\ \underline{5}\ \underbrace{6\ 7}_{1comp}$

Dado el arreglo [1, 2, 3, 4, 7, 6, 5] se resuelve QS mediante el sgte. árbol de recurrencia

```
                    QS(1,7)
   QS(1,4)    T(7)         QS(6,7)
        T(4)              T(2)
QS(1,3)  /   \ QS(5,4)    /   \  QS(8,7)
   T(3)       T(0)      T(1)    T(0)
QS(1,2) /  \ QS(4,3)    QS(6,6)
  T(2)      T(0)
  /   \ QS(3,2)
T(1)   T(0)
QS(1,1)
```

Siendo T(n) una llamada a QS que representa el # de comparaciones entre elementos y pivot durante la operación Partition.

Además
$$T(n) = \begin{cases} 0 & , \quad cc \\ n-1 & \quad n > 0 \end{cases}$$

∴ Si se suman los valores de todos los T(n) del árbol se obtiene
$1 + 2 + 3 + 1 + 6 = 13$ comparaciones

(a)
```
PARTITION (A, p, r)
  x = A[r]
  i = p-1
  for j = p to r-1
    if A[j] ≤ x
      :
      :
  ....
  return i+1
```

Debido a la naturaleza del algoritmo, mostrado en el pseudo, sobre un subarreglo A[p ... r] se toma como elemento pivot al último elemento (A[r]) a partir de ahí todos los demás elementos (A[p...r-1]) se comparan con él.

Entonces existen |A[p... r-1]| comparaciones en una llamada a partition con parámetros p y r:

$$|A[p \ldots r-1]| = (r-1) - p + 1 = r - p$$

Rspta: r-p comparaciones

5

E-4

(a) MIN-PARTITION

OPT(n, K)

Sea [redacted] el peso mínimo de una K-partición sobre $A[1, \dots, n]$

$$\mathrm{OPT}(n,K) \text{ [redacted]} = \begin{cases} \text{[redacted]} \sum_{i=1}^{n} A[i] & , n = K \text{ o } K = 1 \\ \min_{j=K-1}^{n-1} \{ \max \{ \mathrm{OPT}(\text{[redacted]} j, K-1), \sum_{l=j+1}^{n} A[l] \} \} & , cc \end{cases}$$

$A = [2, 1, 3, 2]$, $K = 3$

Matriz / Tabla

$n \downarrow$ \ $K \rightarrow$	1	2	3
1	2	-	-
2	3	2	-
3	6	3	3
4	8	5	3

b) Recurrencia de MOCHILA:

Input : $w[1 \dots n]$, $V[1 \dots n]$, W enteros no negativos

Obj: $K \subseteq \{1, \dots, n\}$ tal que $\sum_{i \in K} w_i \leq W$ y $\sum_{i \in K} V_i$ es máximo

Sea $\mathrm{OPT}(n, W)$: el máximo valor obtenido tomando un subconjunto de items tal que su valor sea máximo y $\leq$ que W.

$$\mathrm{OPT}(n,W) = \begin{cases} 0 & n == 0 \text{ o } W == 0 \\ \mathrm{OPT}(n-1, W) & w_n > W \\ \max \{ \mathrm{OPT}(n-1, W), \mathrm{OPT}(n-1, W - w_n) + V_n \} & , w_n \leq W \end{cases}$$

[2 1 2 3] 5

n \ W	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	2	2	2	2
2	0					
3	0					
4	0					