EECS 4020 Algorithms

HW1

I. Getting Started

Q1

- Let $A[1..n]$ be an array of distinct numbers
- We say

$$\boxed{(\,i, j\,) \text{ is an inversion if } i < j \text{ and } A[i] > A[j]}$$

Q1(a)

How many inversions in the array below?

(2, 3, 8, 6, 1)

Q1(a)

How many inversions in the array below?

(2, 3, 8, 6, 1)

Ans: 5

Q1(b)

Which permutation of $(1, 2, \ldots, n)$

has the most # of inversions ?

Ans: $(n,\ n-1,\ \ldots,\ 2,\ 1)$

Q1(c)

What is the relationship between
running time of insertion sort and # inversions ?

Ans: # swaps in insertion sort = # inversions

$\rightarrow$ running time of insertion sort

= O(n + # inversions)

Q1(d)

How to count # inversions in O(n log n) time ?

Ans: Let A[1..n] be an array

If 1^{st} half of A is increasing, and

2^{nd} half of A is also increasing

→ we can count # inversions in O(n) time

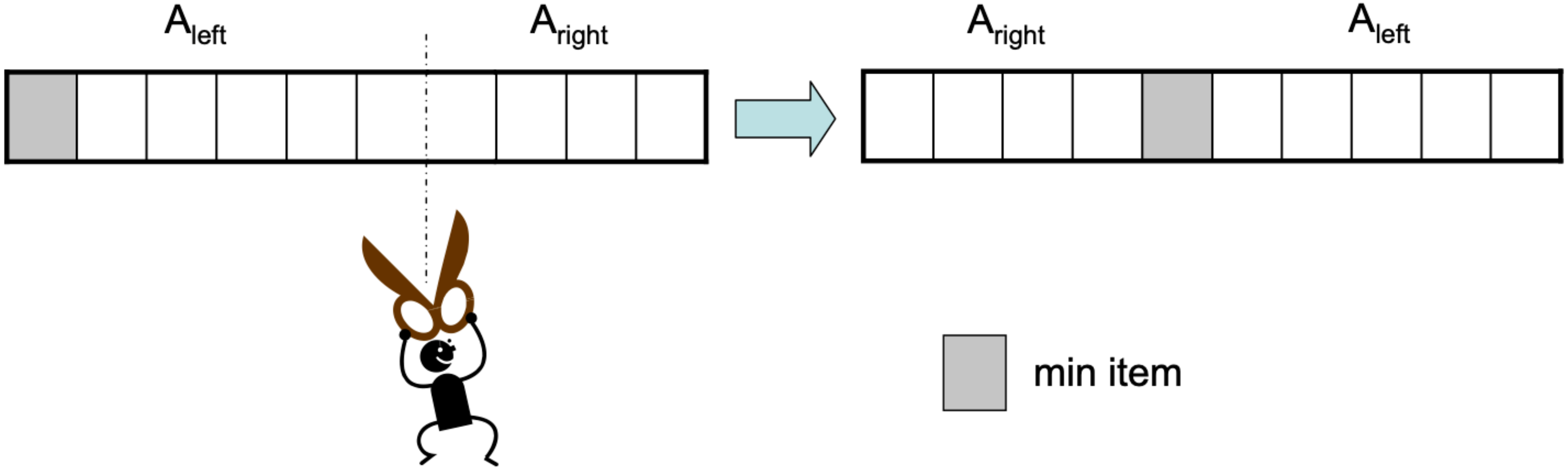
Q1(d)

Ans: To count # inversions in general,
we modify MergeSort as follows:

1. Sort 1st half, count # inversions within
2. Sort 2nd half, count # inversions within
3. Merge 1st half and 2nd half,
   count # inversions between them

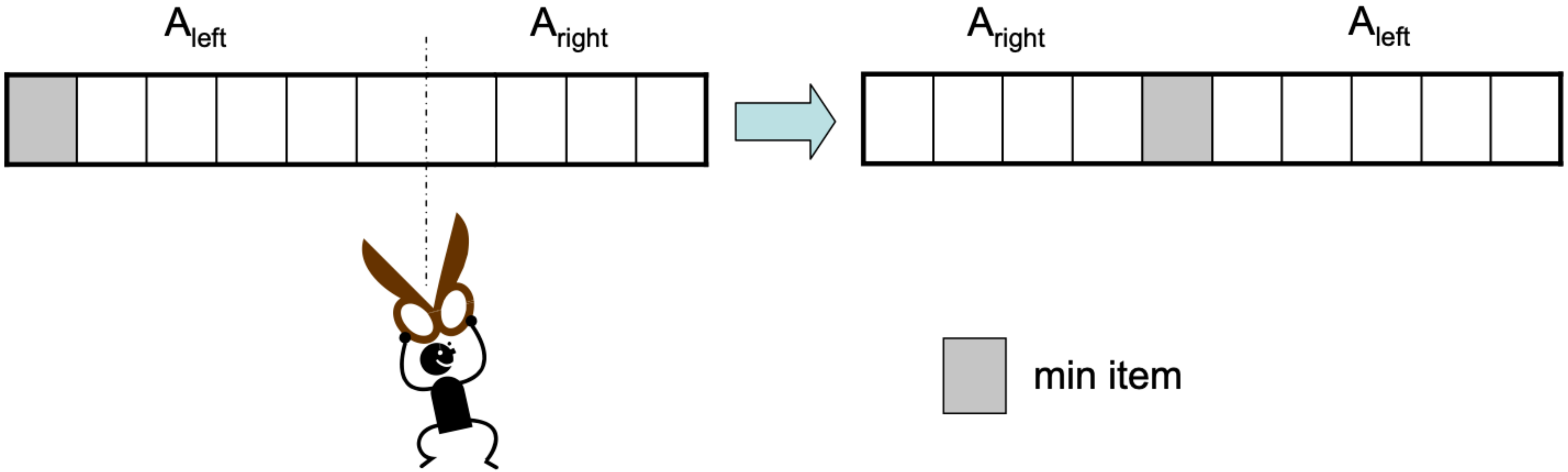
Q2

An increasing array is modified by moving its left part (of unknown length) to its right



Q2

How to find the min item in O(log n) time ?



Q2

Ans : Let B[1..n] be the array after the movement

If B[1] < B[n] the array is not moved

Else, at any location p,

- if B[p] < B[1] → min item is on p's left side
- Else, → min item is on p's right side

This allows us to perform binary search

Q3

Consider the code below:

```
ComputeCount()
1. Input a positive integer n;
2. Set count = 0;
2. for j = 1, 2, ..., n
3.     if j is a factor of n
4.      { Update count to become 1 - count; }
5. Output count;
```

Q3

What does it do ? How to rewrite to run faster ?

`ComputeCount()`

1. Input a positive integer n;
2. Set `count` $= 0$;
2. **for** $j = 1, 2, \ldots, n$
3. **if** j is a factor of n
4. { Update `count` to become 1 - `count`; }
5. Output `count`;

Q3

Ans: Check if n is a square. How to run faster ?

```
ComputeCount()
1. Input a positive integer n;
2. Set count = 0;
2. for j = 1, 2, ..., n
3.     if j is a factor of n
4.      { Update count to become 1 - count; }
5. Output count;
```

II. Growth of Functions

Q1

Given that

$$f(n) = a_m n^m + a_{m-1} n^{m-1} + \ldots + a_1 n + a_0, \quad \text{where } a_m > 0.$$

Show that

$$\boxed{f(n) = \Theta(n^m)}$$

Q1

Ans:

Let A denote $|a_0| + |a_1| + \ldots + |a_{m-1}|$

Observe that

- $f(n) \geq (a_m / 2) n^m$ when $n > 2A / a_m$
- $f(n) \leq (a_m + A) n^m$

Q2

Show that

$$\boxed{k \ln k = \Theta(n) \rightarrow k = \Theta(n / \ln n)}$$

Q2

Ans:

- $cn < k \ln k < Cn$ for some c and C in long run
- Then, $cn < k^2$ and $k < Cn$
- Also, $cn / \ln k < k < Cn / \ln k$

$\Rightarrow$ $cn / \ln (Cn) < k < Cn / \ln (cn)^{1/2}$

Q3

What's wrong with this ?

"Since $n = O(n)$, and $2n = O(n)$, $\ldots$, we have

$$\sum_{k=1}^{n} k \cdot n = \sum_{k=1}^{n} O(n) = O(n^2)."$$

Q4

Is the following true ?

$$O(f(n)) - O(f(n)) = 0.$$

Q5

Classify the following functions in increasing order :

e^n	$(\lg n)^{\lg n}$	$(\sqrt{2})^{\lg n}$	n^2	$n!$	$(\lg n)!$
$4^{\lg n}$	$(n+1)!$	$n \lg n$	n^3	$\lg^2 n$	$\lg(n!)$
$\sqrt{\lg n}$	$2^{\sqrt{2 \lg n}}$	2^{2^n}	$n^{1/\lg n}$	$\ln \ln n$	$n \cdot 2^n$
n	2^n	$n^{\lg \lg n}$	$\ln n$	1	$2^{\lg n}$

Q5 (solution)

<< << << <<

e^n	$(\lg n)^{\lg n}$	$(\sqrt{2})^{\lg n}$	n^2	$n!$	$(\lg n)!$
$4^{\lg n}$	$(n+1)!$	$n \lg n$	n^3	$\boxed{\lg^2 n}$	$\lg(n!)$
$\boxed{\sqrt{\lg n}}$	$2^{\sqrt{2 \lg n}}$	2^{2^n}	$n^{1/\lg n}$	$\ln \ln n$	$n \cdot 2^n$
n	2^n	$n^{\lg \lg n}$	$\ln n$	1	$2^{\lg n}$

Q5 (solution)

$$\bigcirc \ll \bigcirc \ll \square \ll \bigcirc \ll \square$$

e^n	$(\lg n)^{\lg n}$	$(\sqrt{2})^{\lg n}$	$\boxed{n^2}$	$n!$	$(\lg n)!$
$\boxed{4^{\lg n}}$	$(n+1)!$	$n \lg n$	n^3		$\lg(n!)$
	$2^{\sqrt{2 \lg n}}$	2^{2^n}			$n \cdot 2^n$
$\boxed{n}$	2^n	$n^{\lg \lg n}$			$\boxed{2^{\lg n}}$

Q5 (solution)

$\bigcirc \ll \bigcirc \ll \square \ll \bigcirc \ll \square$

e^n

$\boxed{(\lg n)^{\lg n}}$

$n!$

$(\lg n)!$

$(n+1)!$

n^3

$\boxed{n \cdot 2^n}$

2^{2^n}

2^n

$\boxed{n^{\lg \lg n}}$

Q5 (solution)

$\bigcirc \ll \square \ll \bigcirc \ll \square$

e^n

$n!$

$(n+1)!$

2^{2^n}

III. Solving Recurrences

Q1

Solve the recurrence:

$$T(n) = T(n-1) + T(n/2) + n$$

Q1 (solution)

By brute force expansion, we see that :

- $T(n) < n\,T(n/2) + n^2$
- $T(n) > (n/2)\,T(n/4)$

$\Rightarrow$ $T(n) = n^{\Theta(\log n)}$

Q2

Solve the recurrence:

$$T(n) = T(n / 3) + T(2n / 3) + n$$

Ans: $$T(n) = \Theta(n \log n)$$

IV. Heapsort

Q1

- K sorted lists, total length = n

$$\boxed{\text{How to merge them into one sorted list ?}}$$

Ans: Two methods

(1) Use a heap of size O(K)

(2) Pair up and merge; then recursion

Q2

- A list with n nearly-sorted numbers
- Each located at most d pos from its correct pos

$\boxed{\text{How to sort the list ?}}$

Q2

Ans: Three methods

(1) Use a heap of size $O(d)$
(2) Create $O(d)$ sorted list, then merge
(3) $O(n/d)$ rounds of sorting,
each round sorts $O(d)$ numbers