

--

[illegible]

## Question 1

(4 marks)

A. Give the names of two *stable* sorting algorithms, together with their worst-case time complexities. Write the names and complexities in the box:

B. Give the names of two *unstable* sorting algorithms, together with their worst-case time complexities. Write the names and complexities in the box:

## Question 2

(4 marks)

We are given an array  $A$  holding  $n$  integers, for some large  $n$ . **The array is sorted**, and the values in  $A$  range from -2147483648 to 2147483647, evenly distributed. Give  $\Theta$  expressions for the following tasks:

A. Running the insertion sort algorithm on the array  $A$ :

B. Running the selection sort algorithm on the array  $A$ :

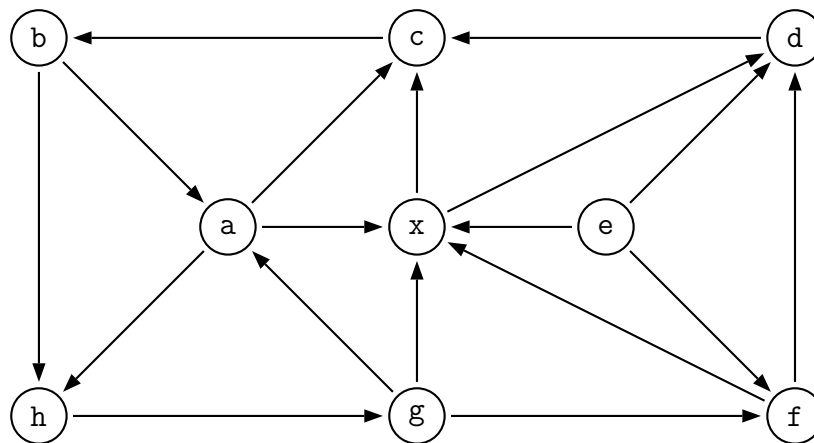
C. Performing binary search for integer  $k$  which is not in  $A$ :

D. Performing interpolation search for integer  $k$  not in  $A$ :

### Question 3

(4 marks)

For the directed graph below, list the order in which the nine nodes are visited during a depth-first (DFS) traversal, as well as the order in which they are visited during a breadth-first (BFS) traversal. As always, assume that any **ties are resolved** by taking nodes in alphabetical order. Write the answers in the boxes given.



DFS sequence:

BFS sequence:

### Question 4

(4 marks)

Given the pattern A T G A and the text

T C A T C A T C C A T G C A C A A T G A C T T T

how many character comparisons will Horspool's algorithm make before locating the pattern in the text? Write the number in the box:

#shift  
A 3  
T 2  
G 1  
\_ 4

Handwritten notes showing the pattern ATGA being shifted and compared against the text. The pattern is highlighted in red. The text is T C A T C A T C C A T G C A C A A T G A C T T T. The pattern is found at index 10 (0-indexed) or index 11 (1-indexed). The handwritten notes show the pattern being shifted and compared against the text, with the final match at index 10.

**Question 5****(4 marks)**

Assume the array  $A$  holds the keys 77, 64, 15, 43, 28, 91, 80, 32, 56 in index positions 1 to 9. Show the heap that results after application of the linear-time bottom-up heap construction algorithm. You may show the heap as a tree or as an array.

**Question 6****(4 marks)**

The functions  $A$ – $D$  are defined recursively as follows (all divisions round down, to the closest integer):

$$\begin{aligned}
 A(n) &= 2 A(n/3) + 2, & \text{with } A(1) &= 1 \\
 B(n) &= B(n/2) + n/2, & \text{with } B(1) &= 1 \\
 C(n) &= 512 C(n/8) + 4n^2, & \text{with } C(1) &= 4 \\
 D(n) &= 4 D(n/2) + n^2, & \text{with } D(1) &= 2
 \end{aligned}$$

In the following table, for each of the four functions, tick the most precise correct statement about the function's rate of growth:

	$O(n)$	$\Theta(n)$	$O(n \log n)$	$\Theta(n^2)$	$O(n^2 \log n)$	$\Theta(n^2 \sqrt{n})$	$O(n^3)$
$A$							
$B$							
$C$							
$D$							

## Question 7

(4 marks)

For each of **A–D** below, answer yes or no, and, in each case, briefly explain your reasoning (just a justification of your answer, rather than detailed calculations). A yes/no answer that is not justified will not attract marks, even if correct.

Question	Answer/explanation
<b>A.</b> Is $\sqrt{n} \in \Omega(n)$ ?	$\lim_{n \rightarrow \infty} \frac{\sqrt{n}}{n} \wedge \sqrt{n} \in O(n)$ FALSE
<b>B.</b> Is $n^{2 \log n} \in O(n^{\log n})$ ?	$\lim_{n \rightarrow \infty} \frac{n^{2 \log n}}{n^{\log n}} \vee n^{2 \log n} \in O(n^{\log n})$ FALSE
<b>C.</b> Is $\Theta(\log(n^{2 \log n})) = \Theta(\log(n^{\log n}))$ ?	<del>Yes</del> <del>No</del> <del><math>\frac{2 \log n \log n}{\log n \log n}</math></del>
<b>D.</b> Is $\Theta(\log(2^n)) = \Theta(\log(3^n))$ ?	<del>Yes</del> <del>No</del> <del>Yes</del> $\frac{\log 2^n}{\log 3^n} = \frac{n \log 2}{n \log 3}$

## Question 8

(6 marks)

A. The box below contains a weighted undirected graph with eight nodes. Give a minimum spanning tree for the graph. You may do that either by outlining a minimum spanning tree on the graph itself, or by drawing the tree in the empty space next to the graph.

*Prim's algorithm*

	a	b	c	d	e	f	g	h	set P
1		5/a	∞	∞	∞	∞	∞	6/a	ab
2			7/b	∞	∞	∞	∞	4/b	abh
3				7/b	∞	∞	8/h		abh c
4				4/d	∞	∞	6/c		abhcd
5					7/d	3/d	5/d		abhcd f
6						7/d	4/f		abhcd f g
7							7/d		abhcd f g e

B. Given a weighted graph  $G = \langle V, E \rangle$ , a subgraph  $\langle V, E' \rangle$  (that is,  $E' \subseteq E$ ) which is a tree with minimal weight is a *maximum spanning tree* for  $G$ .

We want a transformation of the graph  $G$  so that we can run Prim's algorithm on the transformed graph  $G'$ , and the algorithm will find a maximum spanning tree for  $G$ .

Describe such a (systematic) transformation from  $G$  to  $G'$ .

**Question 9****(6 marks)**

Consider the function  $F$  below. The function takes as input an integer array  $A$ , and the size  $n$  of  $A$ . The array indices run from 1 to  $n$ . The division used is integer division, that is, it rounds down to the closest smaller (or equal) integer value.

In the box, give a  $\Theta$  expression for the function's time complexity.

```

function  $F(A[\cdot], n)$ 
   $s \leftarrow 0$ 
   $m \leftarrow n$ 
  while  $m > 0$  do
    for  $i \leftarrow 1$  to  $m$  do
       $s \leftarrow s + A[i]$ 
     $m \leftarrow m/2$ 

```

$$n \quad \frac{n}{2} \quad \frac{n}{4} \quad \frac{n}{8} \dots 1 \quad n = 2^k$$

$$2^k + 2^{k-1} + 2^{k-2} + \dots + 2^{k-k}$$

$$a_1 \frac{1 - 9^n}{1 - 9} = 2 \frac{1 - 2^k}{1 - 2}$$

$$2^k \frac{1 - 2}{1 - 2}$$

$$2^k + 1 - 1$$

## Question 10

(10 marks)

Using pseudo-code, give an algorithm for deleting the smallest element of a binary search tree (a BST). Assume a non-empty binary tree  $T$  has attributes  $T.left$ ,  $T.right$ , and  $T.root$  which denote  $T$ 's left sub-tree, right sub-tree, and the key of  $T$ 's root node, respectively. You can use these tests if they seem useful:  $ISLEAF(T)$  tests whether the binary tree  $T$  is a leaf, and  $ISEMPTY(T)$  tests whether it is empty.



**Question 11****(10 marks)**

Consider an array  $A$  of  $n$  distinct integers (that is, all elements are different). It is known that  $A$  was originally sorted in ascending order, but  $A$  was then right-rotated  $r$  places, where  $0 < r < n$ . In other words, the last  $r$  elements were moved from the end of the array to the beginning, with all other elements being pushed  $r$  positions to the right. For example, for  $n = 7$  and  $r = 3$ , the result may look like this:

[43, 46, 58, 12, 20, 29, 34]

For  $r = 5$ , the result, based on the same original array, would be

[29, 34, 43, 46, 58, 12, 20]

You know that the given  $A[0..n-1]$  has this particular form, that is, for some  $r$ , the sequence  $A[r], \dots, A[n-1], A[0], \dots, A[r-1]$  is in ascending order, but you do not know what  $r$  is. Design an algorithm to find the largest integer in  $A$ . Full marks are given for an algorithm that works in time  $O(\log n)$ ; half marks are given for a solution that is correct, but less efficient.

## Question 12

(10 marks)

Two programmers face the following problem. Given an array containing  $n$  random integers in random order, find the largest integer. The integers are placed in cells  $A[1] \dots A[n]$ .

Programmer  $X$  has come up with the code shown below, on the left. (In the programming language used, arrays are indexed from 0, but  $X$ 's method does not use  $A[0]$ .)

```

function X( $A[\cdot], n$ )
     $max \leftarrow A[1]$ 
     $i \leftarrow 2$ 
    while  $i \leq n$  do
        if  $A[i] > max$  then
             $max \leftarrow A[i]$ 
         $i \leftarrow i + 1$ 
    return  $max$ 

```

```

function Y( $A[\cdot], n$ )
     $i \leftarrow n$ 
    while  $i > 0$  do
         $A[0] \leftarrow A[i]$ 
         $i \leftarrow i - 1$ 
        while  $A[0] > A[i]$  do
             $i \leftarrow i - 1$ 
    return  $A[0]$ 

```

Programmer  $Y$  has solved the same problem differently, as shown above on the right.

Compare the two solutions using three criteria: Correctness, time complexity class, and the number of comparisons performed. Write your analysis in the box:

$n=5$

	0	1	2	3	4	5
		2	4	3	7	1
$X$ :		$\uparrow$ $max$	$\uparrow$ $max$	$\uparrow$	$\uparrow$ $max$ ✓	$\uparrow$
		$i$	$i$	$i$	$i$	$i$
$Y$ :	$\uparrow$ $A[0]$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$	$\uparrow$
	$i$	$i \leftarrow i$	$i \leftarrow i$	$i \leftarrow i$	$i \leftarrow i$	$i$

*(Handwritten diagram illustrating the execution of both algorithms for n=5. The array contains [2, 4, 3, 7, 1]. Programmer X starts with max=2 and iterates through the array, updating max to 4, then 7. Programmer Y starts with i=5, moves A[5] to A[0], and then iterates from i=4 down to 1, comparing A[0] with A[i].)*

## Overflow space

Use this page if you ran out of writing space in some question. Make sure to leave a pointer to this page from the relevant question.