What does $O(< expr >)$ mean?

1

What does $\Theta(< expr >)$ mean?

2

What does $\Omega(< expr >)$ mean?

3

*What are the best, average and worst case complexities of **Bubble Sort**?*

4

*What are the best, average and worst case complexities of **Merge Sort**?*

5

Give pseudo code for merging 2 sorted lists, as part of merge sort.

6

Give pseudo code for MergeSort(L).

7

*What are the best, average and worst case complexities of **Quick Sort**?*

8

The complexity (i.e. space/running time) has the complexity proportional to $< expr >$.

2

The complexity (i.e. running time/space) is bounded by the $< expr >$.

1

Best: $O(n)$,
Average: $O(n^2)$,
Worst: $O(n^2)$

4

The complexity (i.e. running time/space) is at least by the $< expr >$.

3

Merge(L_1, L_2)
  if $L_1 = []$ *return* L_2
  if $L_2 = []$ *return* L_1
  $x_1 = L_1[0]$
  $x_2 = L_2[0]$
  $L_1' = L_1[1 : |L_1| - 1]$
  $L_2' = L_2[1 : |L_2| - 1]$
  if $x_1 \leq x_2$
    return $[x_1]$+*Merge*(L_1' ,L_2)
  return $[x_2]$+*Merge*(L_1 ,L_2')

Merge two sorted lists

6

Best: $O(n \log_2 n)$,
Average: $O(n \log_2 n)$,
Worst: $O(n \log_2 n)$

5

Best: $O(n \log_2 n)$,
Average: $O(n \log_2 n)$,
Worst: $O(n^2)$

8

MergeSort(L)
  if $|L| \leq 1$
    return L
  Split L *into roughly equal halves,* L_l *and* L_r
  return Merge(*MergeSort*(L_l),*MergeSort*(L_r))

MergeSort(L)

7

What would the pseudo code be for Quick Sort?

9

Say that the input represents a positive integer, x, what is the size of n?

10

What does it mean by $O(1)$?

11

What is the minimum time for any sorting algorithm that uses only number comparisons?

12

What would the pseudo code be for Euclid's algorithm?

13

What would the pseudo code be for Fast Modular Exponentiation?

14

What are some of the advantages of ElGamal encryption?

15

What is the basic procedure for an encryption and decryption using publik key cryptography if Alice wants to send a message to Bob?

16

$\lfloor \log_b x \rfloor + 1$ *Where b is the number representation, usually binary (so 2).*

10

quicksort(L)
    if length of $L \leq 1$
        return L
    remove the first element, x, from L
    $L_{\leq}$:= *elements of L less than or equal to x*
    $L_{>}$:= *elements of L greater than x*
    L_l := *quicksort*($L_{\leq}$)
    L_r := *quicksort*($L_{>}$)
    return L_l + [x] + L_r

Quick Sort

9

$n \log_2 n$

12

It takes a constant time, no matter the amount of data, to perform the operation.

11

fme(a,b,k)
    $d = a$
    $e = b$
    $s = 1$
    While $e > 0$
        if e is odd
            $s = (s.d) mod k$
        $d = d^2 mod k$
        $e = \lfloor e/2 \rfloor$
    return s

Fast Modular Exponentiation

14

// Assume a >= b
hcf(a,b)
    if $b = 0$
        return a
    $r = a mod b$
    return hcf(b,r)

Euclid's algorithm

13

Alice generates a private random integer a and Bob generates a private random integer b
Alice generates her public value $g^a \bmod p$
Bob generates his public value $g^b \bmod p$
Alice computes $g^{ab} = (g^a)^b \bmod p$
Bob computes $g^{ba} = (g^b)^a \bmod p$
Now they have a shared secret k since $k = g^{ab} = g^{ba}$

16

Sender Verification
Private key remains with owner
Public key is freely distributable
No secret channel needed at any point
No need for pre-shared keys

15

Describe public key generation in ElGamal encryption using p as the Prime Modulus and g as the Primitive root (as described in the COMP26120 lab)

17

Describe the encryption procedure used in the ElGamal cryptosystem given that person B wants to send message M to preson A

18

Describe the decryption process used in the ElGamal cryptosystem given that person A has received cyphertext (γ, δ) from person B, encrypted encrypted using the public key (p, g, g^a)

19

Consider the equation $a^x = y \bmod p$. If a is a primitive root of modulo p, then for every $y (1 \leq y < p)$, such an $x (1 \leq x < p)$ exists. What is x?

20

The ______ is the inverse of exponentiation.

21

Why can a private key in the ElGamal cryptosystem not, in practice, be recovered using the public key when p is large?

22

What is one way you can argue correctness of Euclid's algorithm?

23

What would half the correctness proof be for Euclid's algorithm?

24

Obtain A's public key (p, g, g^a)
Represent the message M as integers in the range $0, ..., p-1$
Select a random integer k from $1 \leq k \leq p-2$
Compute $\gamma = g^k \bmod p$ *and* $\delta = m \cdot (g^a)^k$
Send ciphertext $c = (\gamma, \delta)$ *to A*

18

Generate a large p and a g in $1 \leq g < p$
Generate a random integer a in $1 \leq a \leq p-2$
Compute $g^a \bmod p$*. The public key is*

$$(p, g, g^a)$$

The private key is a

17

X is the ***discrete logarithm*** *of y with base a, modulo p.*

20

Use private key a to compute $(\gamma^{p-1-a}) \bmod p$
NOTE THAT: $(\gamma^{p-1-a}) = \gamma^{-a} = g^{-ak}$
Recover the message M by computing $(\gamma^{-a} \cdot \delta \bmod p)$
Note that this evaluates to $(g^{-ak} \cdot g^{ak} \cdot M \bmod p)$ *or* $1 \cdot M \bmod p$

19

To calculate a public key, y, a private key, x is needed. The equation for modular exponentiation can be used to generate the public key: $y = g^x \bmod p$ *where g is a primitive root of the modulus p.*
It is considered a one-way, or trapdoor function - easy to compute, hard to invert. For a large p, one of the few ways to figure out the private key x would be to calculate $g^x \bmod p$ *for every x in* $1 \leq x < p$ *and find when one of these results matches y*

22

The discrete logarithm is the inverse of exponentiation.

21

As $r = a \bmod b$*,* $\exists q$ *such that* $a = bq + r$*,*
$\therefore r = a - bq$*.*
Suppose x is a factor of a and b, then $\exists yandz$ *such that* $a = xy$*,* $b = xz$*.*
Hence: $r = xy - xzq$*,* $r = x(y - zq)$*.*
$\therefore$ *x is a factor of r (and also of b and r).*

24

Let $r = amodb$*.* $hcf(a, b) = hcf(b, r)$ *because all factors of a and b are also factors of b and r and vice versa. If they have the same factors, they have the same highest common factor.*

23

$(a.b) mod k =$

25

Let p be a prime number. What is meant by a primitive root modulo p?

26

What does saying that algorithm A runs in time g mean?

27

What is a permutation of a set?

28

What do we mean by a composition of two permutations?

29

What is the number of possible permutations on an n-element set?

30

In the context of a permutation, what do we mean by a transposition?

31

Convert this pair of simultaneous equations into matrix form

$a_{1,1}x_1 + a_{1,2}x_2 = b_1$

$a_{2,1}x_2 + a_{2,2}x_2 = b_2$

32

The numbers r_x between 1 and $p-1$ that, when raised by the numbers between 1 and $p-1$ compute all the numbers between 1 and $p-1$ in some order with no repetitions.

26

$$(a.b) mod k = (a mod k . b mod k) mod k$$

25

A 1-to-1 map of the set onto itself. In basic terms, it is a set mapped to another order of itself. i.e $[0, 1, 2, 3, 4] \mapsto [2, 4, 1, 0, 3]$

28

Given an input of size n, the number of operations executed by A is bounded above by g(n).

27

$n!$

30

The composition is the product of two permutations, α and β, on a set n, given by $\alpha \cdot \beta(n)$ or $\beta(\alpha(n))$

29

$$\begin{pmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

32

A transposition is a special kind of permutation where only 2 elements in a set are affected (they are swapped). On a set X a transposition $\sigma = (i, j)$ is given by

$$\sigma(k) = \begin{cases} j & \text{if } k = i \\ i & \text{if } k = j \\ k & \text{ow.} \end{cases}$$

31

What is the determinant of the matrix:

$$\begin{pmatrix} a_1 & a_2 \\ a_3 & a_4 \end{pmatrix}$$

33

What is an upper triangular matrix and how do you calculate its determinant?

34

Which 4 operations have no effect on a matrix's determinant?

35

It is a matrix where all of its entries below the diagonal are zero.

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ 0 & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{n,n} \end{pmatrix}$$

Its determinant is calculated by taking the product of the entries on the diagonal. i.e $a_{1,1} \cdot a_{2,2} \cdot ... \cdot a_{n,n}$

$a_1 a_4 - a_2 a_3$

Often denoted as:

$$\begin{vmatrix} a_1 & a_2 \\ a_3 & a_4 \end{vmatrix}$$

The original system of equations to which the matrix corresponds only has a unique solution if the determinant is non-zero.

Transposing two rows
Transposing two columns
Adding a multiple of one row to another
Adding a multiple of one column to another
Also note that if all entries in any row or column are 0 then the determinant is 0