FFT $\omega_n = e^{2\pi i/n}$ $p(x) = ax+b$, $q(x) = cx+d$

$FFT([b,a,0,0],4)$: $p=[b,a,0,0]$; $p^0=[b,0]$; $p^1=[a,0]$; $FFT([b,0],2) = (FFT([b],1)+FFT([0],1), b-0)$

$FFT([a,0],2) = (a_1+0, a_2-0)$; $\omega=1: d_0 = b_1+a_1$; $\omega=i: d_1 = b_2+a_2 i$; $\omega=-1: d_2 = b_1-a_1$; $\omega=-i: d_3 = b_2-a_2 i$

$FFT(p,4) = (d_0,d_1,d_2,d_3)$; $FFT(q,4) = (d+c, d+ci, d-c, d-ci)$; $FFT(p\cdot q,4) = ((b+a)(d+c), (b+ai)(d+ci), (b-a)(d-c), (b-ai)(d-ci))$

interpolation: $FFT([\langle FFT(p\cdot q,4)\rangle],4) = (d_0,d_1,d_2,d_3)$; $a_0 = \frac{1}{4}\cdot d_0$, $a_1 = \frac{1}{4}\cdot d_3$; $a_2 = \frac{1}{4}\cdot d_2$

$a_3 = \frac{1}{4}\cdot d_1 \Rightarrow p\cdot q = a_3x^3+a_2x^2+a_1x+a_0$ | $O(n\log n)+O(n)+O(n\log n) = O(n\log n)$

RSA p,q,e | $n = p\cdot q$ | $\varphi(n) = (p-1)(q-1)$ | $e\cdot d + k\cdot\varphi(n) = 1 = \text{ggT}(e,\varphi(n)) \Rightarrow$ (gcd)

$\varphi(n) = x_1\cdot e + r_1$ $\quad r_2 = e - x_2\cdot r_1$

$e = x_2\cdot r_1 + r_2 \Rightarrow 1 = e - x_2\cdot(\varphi(n) - x_1\cdot e) = (x_1\cdot x_2+1)e - x_2\cdot\varphi(n)$ (d, k)

$r_1 = x_3\cdot r_2 + 0$

$P = (e,n)$, $S = (d,n)$, P = public key | $P(M) = M^e \bmod n$, $S(C) = C^d \bmod n$

Universal Hashing class H: $x\neq y$: $\frac{|\{h\in H : h(x)=h(y)\}|}{|H|} \leq \frac{1}{m}$ | $h: U \to [0..m-1]$, $U = [0..N-1]$

$H = \{h_{a,b}(x) \mid 1\leq a<N \wedge 0\leq b<1\}$

N prime | $h_{a,b}(x) = ((ax+b) \bmod N) \bmod m$ | collisions $\exists c\in\mathbb{N}: a(x-y) = cN \wedge (x-y)|N \Leftrightarrow (x-y) = \ell\cdot k$

$O(1+\frac{n}{m})$ | $0<\ell\leq\ell_m \Rightarrow \text{sum} = N\cdot\ell_m - \sum_{\ell=1}^{\ell_m} k\cdot\ell$

Perfect Hashing S, $n=|S|$, k, N | $h_k(x) = (kx \bmod N) \bmod n$; $W_i = \{x\in S \mid h_k(x)=i\}$; $b_i = |W_i|$;

$m_i = 2b_i(b_i-1)+1$; k_i so that $h_{k_i}(x) = (k_i x \bmod N) \bmod m_i$ injective on W_i; $s_i = \sum_{j<i} m_j$

Save $x\in S$ at $T[s_i+j]$, $i = (kx \bmod N) \bmod n$, $j = (k_i x \bmod N) \bmod m_i$

Construction time $O(n)$, space $O(n)$, access in $O(1)$

Ackerman $\alpha(n) = \min\{i\geq 1 \mid A(i,1)>n\}$

$A(0,j) = j+1$

$A(k,j) = A^{(j+1)}(k-1,j)$ for $k\geq 1$

$A^{i+1}(k,j) = A(k, A^i(k,j))$

$A^1(k,j) = A(k,j)$

Bin packing $NF(I)\leq 2OPT(I)$ | $FF(I) \leq \lceil 17/10\, OPT(I)\rceil$ | $BF(I) = OPT(I)$

$FFD(I) \leq (4\,OPT(I)+1)/3$ ($O(n)$, $O(n^2)$ $\longrightarrow$ $O(n\log n) \leftarrow O(n^2)$)

KMP	a	b	a	b	a	c	
	-1	0	0	1	2	3	0

$O(n+m)$

Dijkstra $O(n\log n + m)$ FibHeap; $O(n(T_{Ins}+T_{Empty}+T_{DelMin})+m\,T_{Deckey}+m+n)$

Bellman-Ford $O(n^2+n\cdot m)$ | $Z[v]>n \Rightarrow$ negative cycle | acyclic graph: U = list ord. by $num(v)$

Kruskal $O(m\cdot\alpha(n)+m+m\cdot\log n)$ | m-Edges | n-nodes

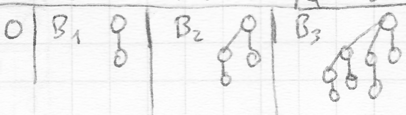
$\alpha(n)$-Acherman inverse, $m\cdot\alpha(n)$ from $O(m)$-Union-Find, $O(n)$-make-Set

L = list of E sorted by cost | add e to A if no cycle and minimal

Prim $O(n\log n+m)$; A is one tree; key is min cost from A to v. Q = Fib-Heap

	List	Heap	Bin-Q	Fib.Hp
insert	$O(1)$	$O(\log n)$	$\log n$	1
min	$O(n)$	1	$\log n$	1
del-min	$O(n)$	$\log n$	$\log n$	$\log n$*
meld	$O(1)$	n / $m\log n$	$\log n$	1
dec.key	$O(1)$	$\log n$	$\log n$	1*

BinTree B_0 B_1 B_2 B_3



B_n has 2^n nodes; B_n height n; root of B_n order n; $\binom{n}{i}$ nodes with height i in B_n

Bin. Queue n = number of keys; $B_i\in Q \Leftrightarrow (n)_2[i] = 1$

Parent	
entry	degree
child	sibling

$Q.root = null$; $Q.insert(e)$: $B_0.entry = e$, $Q.meld(B_0)$.

$Q.deletemin()$: remove min, invert children, meld.

$Q.dec.Key(v,k)$: $v.entry.key = k$, move $v.entry$ up.

Fib.Heap Qmin | Q.rootlist | Q.size

left	Parent		right
	entry	degree	
	child	mark	

$Q.init()$: $Q.rootlist = Q.min = null$.

$Q.meld(K)$: unite $Q.rootlist + K.rootlist$, update $Q.min$.

$Q.insert(e)$: Q' with e, $Q.meld(Q')$.

$Q.deletemin()$: remove $Q.min$ from $Q.rootlist$, add $Q.min.childlist$ in $Q.rootlist$, $Q.consolidate()$.

$Q.consolidate()$: $|A| = 2\log_2 n$, link of same degree, update $Q.min$.

$rang(v) = grad$, $rang(Q) = \max grad$, $Q.size = n$.

$Q.decreasekey(v,k)$: $v.key = k$, update $Q.min$, cascading cuts if $k<$ parent, mark on child remove, update degree for parent.

v becomes child $\Rightarrow v.mark = false$ | v loses child $\Rightarrow v.mark = true$ | v loses 2nd. child $\Rightarrow cut(v)$

Disjoint-set $e.makeset()$: $e.parent = e$, $e.size = 1$. $Link(x,y)$: smaller x becomes child, size of parent increased.

find-set in $O(\log n)$ | compression + union by rank $\Rightarrow \Theta(m\cdot\alpha(n))$

m operations, f-find-set, n-make-set $\rightarrow$ at most $n-1$-unions: Link by rank in $O(n+f\log n)$, find-set with compression if $f<n$: $\Theta(n+f\log n)$ else $\Theta(f\log_{1+f/n} n)$

Closest Pair $S = \{p_i\}$;

Sort S by x-axis inc.

$minDist(S)$: $S_\ell = \{p_a,p_b,p_c,p_d\}$; $S_r = \{p_e,p_f,p_g,p_h\}$

$d_\ell = minDist(S_\ell)$; $d_r = minDist(S_r)$

bound $d = \min(d_\ell, d_r)$; points within bound: (p_c,p_f,p_g)

sort by y-axis, test pairs less than 16 vertical diff.

return the min d. | $\Theta(n\cdot\log n)$

$minDist(S_\ell)$...

Interval Scheduling (S,f) = (starts, finishing times)

f-sorted inc. Pick interval with earliest f-time

$\Theta(n)$ [+ sort by f in $\Theta(n\log n)$]

ReportCuts | $O(n\log n+k)$ k = #Cuts

$ReportCuts(S) = ReportCuts(S_1) \cup RC(S_2) \cup M$

$RC(S_1)$: $L(S_1) = \{A \mid$ only left-end A in $S_1\}$

$R(S_1) = \{A \mid$ only right-end A in $S_1\}$

$RC(S_2)$ $V(S_1) = \{a \mid a$ is vertical in $S_1\}$

$L(S) = (L(S_1) \setminus R(S_2)) \cup L(S_2)$

$R(S) = (R(S_2) \setminus L(S_1)) \cup R(S_1)$; $V(S) = V(S_1)\cup V(S_2)$

L,R sorted by increasing y-pos | V sorted by inc. lower and points

$M = M_1 \cup M_2$ | $Cut(P,p)$: $p.\underline{y} \leq P.y \leq p.\overline{y}$

$M_1 = \{(P,p) \mid P\in R(S_2)\setminus L(S_1) \wedge p\in V(S_1) \wedge Cut(P,p)\}$

$M_2 = \{(P,p) \mid P\in L(S_1)\setminus R(S_2) \wedge p\in V(S_2) \wedge Cut(P,p)\}$

Matrix-chain product | $((A_{i\ldots k})(A_{k+1\ldots j}))$ $i \le k < j$ | $m[i,i]=0$ | $m[i,j]=\min\{m[i,k]+m[k+1,j] + p_{i-1}p_k p_j\}$

$p < p_0, p_1, \ldots p_n >$ $p_{i-1} \times p_i : A_i$ | $s[i,j]=$ opt. k | $O(n^3)$ | other alg in $O(n \log n)$

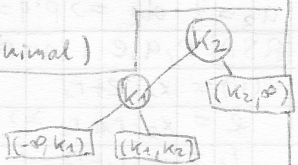
Optimal Search Trees | $S=\{k_1, \ldots k_n\}$, $-\infty = k_0 < k_1 \ldots < k_n < k_{n+1} = \infty$; a_i rate of k_i ; b_j rate of $x \in (k_j, k_{j+1})$

weighted path-length $P(T)$ for search tree T of S: $P(T)=\sum_{i=1}^{n}(\text{depth}(k_i)+1)a_i + \sum_{j=0}^{n}\text{depth}((k_j,k_{j+1}))b_j$

$W(i,i)=b_i$ $\Rightarrow$ T has only one node for range (k_i, k_{i+1}) $\Leftarrow$ $0 \le i < j \le n$ | $O(n^3)$ | space $O(n^2)$

$W(i,j)=W(i,j-1)+a_j+b_j$

$P(i,i)=0$; $P(i,j)=W(i,j)+\min\{P(i,l-1)+P(l,j)\}$ | $i < l \le j$ $r(i,j)=l$ (minimal)



$\boxed{\text{EX:}}$ k_1 $(a_1=2)$, k_2 $(a_2=4)$, $(-\infty,k_1)$ $(b_0=1)$, (k_1,k_2) $(b_1=3)$, (k_2,∞) $(b_2=5)$

$h=j-i=0$: $P(0,0)=0$, $P(1,1)=0$, $P(2,2)=0$, $W(0,0)=1$, $W(1,1)=3$, $W(2,2)=5$

$h=$ " $P(0,1)=6$, $P(1,2)=12$, $W(0,1)=6$, $W(1,2)=12$

$h=1$ $P(0,2)=W(0,2)+\min\{P(0,l-1)+P(l,2)\} = W(0,2)+P(0,1)+P(2,2)=15+6+0=21$ with $l=2$

$0 < l \le 2$

Dijkstra | V, E, c | U is FibHeap

```
DIST[s]=0 ; U.insert(0,s)
[DIST[v]=inf for v in V if v!=s]=DIST (1)
while not U.empty():
    (d,u) = U.deleteMin()
    for (u,v) in E:
        if DIST[v]>DIST[u]+c(u,v):
            DIST[v] = DIST[u]+c(u,v)
            U.decreaseKey(v, DIST[v])
```

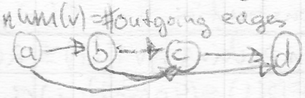
Bellman-Ford | V, E, c | U is LinkedList

```
DIST[s] = 0 ; Z[s]=0
(1) + Z[v]=0 ; U.insert(s)
while not U.empty():
    u = U.headpop() ; Z[u]+=1
    if Z[u]>n return "negative cycle"
    for (u,v) in E:
        if DIST[v]>DIST[u]+c(u,v):
            DIST[v] = DIST[u]+c(u,v)
            U.insert(v)
```

acyclic | sort V by num

```
DIST[s]=0 else ∞
U = sorted v in V
while not U.empty():
    u = min(U)
    for (u,v) in E:
        DIST[v]=min(..DIST[u]+c
```

num(v) = #outgoing edges



Primality test

prime Test	power		$x \cdot x \% n$		$p\%2==1$
result	p	x	result	is Prob Prime	result
	n-1				
	(n-1)/2	O			(1)
	⋮				

Kruskal | $A=[\,]$; $B_v=\{v\}$ $v \in V$

```
L = List of e∈E sorted inc. by weight
for (u,v) in L:
    B1 = find-set(u) ; B2 = find-set(v)
    if B1 ≠ B2:
        A.insert((u,v)) ; B1 = union(B1,B2)
```

Prim | Q.insert(∞, v) $v \in V$; Pick $s \in V$

```
Q.deckey(s,0) ; P[s]=nil
while not Q.empty():
    (d,u) = Q.delMin()
    for (u,v) in E:
        if v in Q and c(u,v) < keyof(v):
            Q.deckey(v, c(u,v)) ; P[v]=u
```

Perfect hashing

i	$W_i=\{x \in S : h_k(x)=i\}$	$b_i=\lvert W_i \rvert$	$m_i=2b_i(b_i-1)+1$	$S_i=\sum m_j$
0				
1				
⋮				

k_i such that $h_{k_i}(x)=((k_i x) \bmod N) \bmod m_i$ injective on W_i

W_1 :

x	$h_{k_1}(x)$
c	a
d	b

$k_1 = \alpha$ so that $a \ne b$

⋮

$pos(x)=S_i+j$

$i=h_k(x)$, $j=h_{k_i}(x)$

Primality | n

```
isprime = True
a = 7
result = power(a, n-1, n)
if result != 1 or not isprime:
    return false
return true
```

```
power(a, p, n):
    if p==0:
        return 1
    x = power(a, p/2, n)
    result = (x^2)%n
    if result==1 and x!=1
                 and x!=n-1:
        isprime = false
    if p%2==1:
        result = (a*result)%n
    return result
```