



Chulalongkorn University

# What Name ?

Pakapim E., Pasin P., Sarit P.

template from KACTL

2025-07-20

1	Template
2	Mathematics
3	Combinatorial
4	Numerical
5	Group
6	Data Structures
7	Number Theory
8	Graph
9	Tree
10	Polynomials
11	Strings
12	Geometry
13	Dynamic Programming
14	Convolutions
15	Various

# Template (1)

```
template.cpp31 lines

#pragma once
#include <bits/stdc++.h>
#define sz(x) (int)(x).size()
#define all(x) (x).begin(), (x).end()

using namespace std;

typedef long long ll;
typedef double db;
typedef long double ld;
typedef pair<int, int> pii;
typedef pair<ll, ll> pll;

template<typename T> bool ckmin(T &a, const T &b) { return b < a ? a = b, 1 : 0; }
template<typename T> bool ckmax(T &a, const T &b) { return a < b ? a = b, 1 : 0; }

mt19937 rng(chrono::steady_clock::now().time_since_epoch().count());

const char nl = '\n';
const int INF = 0x3fffffff;
const int MOD=1000000007;
```

```
1 // const int MOD = 998244353;
const ll LINF = 0xfffffffffffffff;
1 const db DINF = numeric_limits<db>::infinity();
const db EPS = 1e-9;
const db PI = acos(db(-1));

1 signed main(){
    ios_base::sync_with_stdio(0); cin.tie(NULL);
2     return 0;
}

2 c.sh2 lines

2 g++ -std=gnu++2a -Wall $1 -o a.out
./a.out
```

## Mathematics (2)

### 2.1 Goldbatch’s Conjecture

- Even number can be written in sum of two primes (Up to 1e12)
- Range of  $N^{th}$  prime and  $N + 1^{th}$  prime will be less than or equal to 300 (Up to 1e12)

### 2.2 Divisibility

Number of divisors of  $N$  is given by  $\prod_{i=1}^k (a_i + 1)$  where  $N = \prod_{i=1}^k p_i^{a_i}$  and  $p_i$  are prime factors of  $N$ .

## Combinatorial (3)

### 3.1 Permutations

#### 3.1.1 Factorial

$n$	1	2	3	4	5	6	7	8	9	10
$n!$	1	2	6	24	120	720	5040	40320	362880	3628800
$n$	11	12	13	14	15	16	17			
$n!$	4.0e7	4.8e8	6.2e9	8.7e10	1.3e12	2.1e13	3.6e14			
$n$	20	25	30	40	50	100	150	171		
$n!$	2e18	2e25	3e32	8e47	3e64	9e157	6e262	>DBL_MAX		

IntPerm.h

#### 3.1.2 Cycles

Let  $g_S(n)$  be the number of  $n$ -permutations whose cycle lengths all belong to the set  $S$ . Then

$$\sum_{n=0}^{\infty} g_S(n) \frac{x^n}{n!} = \exp \left( \sum_{n \in S} \frac{x^n}{n} \right)$$

#### 3.1.3 Derangements

Permutations of a set such that none of the elements appear in their original position.

$$D(n) = (n - 1)(D(n - 1) + D(n - 2)) = nD(n - 1) + (-1)^n = \left\lfloor \frac{n!}{e} \right\rfloor$$

#### 3.1.4 Burnside’s lemma

Given a group  $G$  of symmetries and a set  $X$ , the number of elements of  $X$  up to symmetry equals

$$\frac{1}{|G|} \sum_{g \in G} |X^g|,$$

where  $X^g$  are the elements fixed by  $g$  ( $g.x = x$ ).

If  $f(n)$  counts “configurations” (of some sort) of length  $n$ , we can ignore rotational symmetry using  $G = \mathbb{Z}_n$  to get

$$g(n) = \frac{1}{n} \sum_{k=0}^{n-1} f(\gcd(n, k)) = \frac{1}{n} \sum_{k|n} f(k) \phi(n/k).$$

### 3.2 Partitions and subsets

#### 3.2.1 Partition function

Number of ways of writing  $n$  as a sum of positive integers, disregarding the order of the summands.

$$p(0) = 1, \quad p(n) = \sum_{k \in \mathbb{Z} \setminus \{0\}} (-1)^{k+1} p(n - k(3k - 1)/2)$$

$$p(n) \sim 0.145/n \cdot \exp(2.56\sqrt{n})$$

$n$	0	1	2	3	4	5	6	7	8	9	20	50	100
$p(n)$	1	1	2	3	5	7	11	15	22	30	627	$\sim 2e5$	$\sim 2e8$

#### 3.2.2 Lucas’ Theorem

Let  $n, m$  be non-negative integers and  $p$  a prime. Write  $n = n_k p^k + \dots + n_1 p + n_0$  and  $m = m_k p^k + \dots + m_1 p + m_0$ . Then  $\binom{n}{m} \equiv \prod_{i=0}^k \binom{n_i}{m_i} \pmod{p}$ .

#### 3.2.3 Binomials

multinomial.h

### 3.3 General purpose numbers

#### 3.3.1 Bernoulli numbers

EGF of Bernoulli numbers is  $B(t) = \frac{t}{e^t - 1}$  (FFT-able).

$$B[0, \dots] = [1, -\frac{1}{2}, \frac{1}{6}, 0, -\frac{1}{30}, 0, \frac{1}{42}, \dots]$$

Sums of powers:

$$\sum_{i=1}^n n^m = \frac{1}{m+1} \sum_{k=0}^m \binom{m+1}{k} B_k \cdot (n+1)^{m+1-k}$$

Euler-Maclaurin formula for infinite sums:

$$\begin{aligned} \sum_{i=m}^{\infty} f(i) &= \int_m^{\infty} f(x) dx - \sum_{k=1}^{\infty} \frac{B_k}{k!} f^{(k-1)}(m) \\ &\approx \int_m^{\infty} f(x) dx + \frac{f(m)}{2} - \frac{f'(m)}{12} + \frac{f'''(m)}{720} + O(f^{(5)}(m)) \end{aligned}$$

3.3.2 Stirling numbers of the first kind

Number of permutations on  $n$  items with  $k$  cycles.

$$c(n,k)=c(n-1,k-1)+(n-1)c(n-1,k),\; c(0,0)=1$$
$$\sum_{k=0}^nc(n,k)x^k=x(x+1)\ldots(x+n-1)$$

$$c(8,k)=8,0,5040,13068,13132,6769,1960,322,28,1$$
$$c(n,2)=0,0,1,3,11,50,274,1764,13068,109584,\ldots$$

3.3.3 Eulerian numbers

Number of permutations  $\pi\in S_n$  in which exactly  $k$  elements are greater than the previous element.  $k$   $j$ :s s.t.  $\pi(j)>\pi(j+1)$ ,  $k+1$   $j$ :s s.t.  $\pi(j)\geq j$ ,  $k$   $j$ :s s.t.  $\pi(j)>j$ .

$$E(n,k)=(n-k)E(n-1,k-1)+(k+1)E(n-1,k)$$

$$E(n,0)=E(n,n-1)=1$$

$$E(n,k)=\sum_{j=0}^k(-1)^j\binom{n+1}{j}(k+1-j)^n$$

3.3.4 Stirling numbers of the second kind

Partitions of  $n$  distinct elements into exactly  $k$  groups.

$$S(n,k)=S(n-1,k-1)+kS(n-1,k)$$

$$S(n,1)=S(n,n)=1$$

$$S(n,k)=\frac{1}{k!}\sum_{j=0}^k(-1)^{k-j}\binom{k}{j}j^n$$

3.3.5 Bell numbers

Total number of partitions of  $n$  distinct elements.  $B(n)=1,1,2,5,15,52,203,877,4140,21147,\ldots$ . For  $p$  prime,

$$B(p^m+n)\equiv mB(n)+B(n+1)\pmod{p}$$

3.3.6 Labeled unrooted trees

# on  $n$  vertices:  $n^{n-2}$   
# on  $k$  existing trees of size  $n_i$ :  $n_1n_2\cdots n_kn^{k-2}$   
# with degrees  $d_i$ :  $(n-2)!/((d_1-1)!\cdots(d_n-1)!)$

3.3.7 Catalan numbers

$$C_n=\frac{1}{n+1}\binom{2n}{n}=\binom{2n}{n}-\binom{2n}{n+1}=\frac{(2n)!}{(n+1)!n!}$$

$$C_0=1,\; C_{n+1}=\frac{2(2n+1)}{n+2}C_n,\; C_{n+1}=\sum C_iC_{n-i}$$

$$C_n=1,1,2,5,14,42,132,429,1430,4862,16796,58786,\ldots$$

- sub-diagonal monotone paths in an  $n\times n$  grid.
- strings with  $n$  pairs of parenthesis, correctly nested.
- binary trees with with  $n+1$  leaves (0 or 2 children).

template c

- ordered trees with  $n+1$  vertices.
- ways a convex polygon with  $n+2$  sides can be cut into triangles by connecting vertices with straight lines.
- permutations of  $[n]$  with no 3-term increasing subseq.

Numerical (4)

4.1 Newton’s Method

if  $F(Q)=0$ , then  $Q_{2n}\equiv Q_n-\frac{F(Q_n)}{F'(Q_n)}\pmod{x^{2n}}$

$$Q=P^{-1}:Q_{2n}\equiv Q_n\cdot(2-P\cdot Q_n^2)\pmod{x^{2n}}$$

$$Q=\ln P=\int\frac{P'}{P}\mathrm{d}x$$

$$Q=e^P:Q_{2n}\equiv Q_n(1+P-\ln Q_n)\pmod{x^{2n}}$$

$$Q=\sqrt{P}:Q_{2n}\equiv\frac{1}{2}(Q_n+P\cdot Q_n^{-1})\pmod{x^{2n}}$$

$$Q=P^k=\alpha^kx^{kt}e^{k\ln T};P=\alpha\cdot x^t\cdot T,T(0)=1$$

Group (5)

5.1 Monoid

monoid/MonoidBase.hpp

5.2 Action

action/MonoidActionBase.hpp

action/DefaultAction.hpp

Data Structures (6)

OrderedSet.hpp

Number Theory (7)

ExtendedEuclid.hpp

euclid.h

CRT.hpp

phiFunction.hpp

FloorSum.hpp

7.1 Prime Numbers

MillerRabin.hpp

LinearSieve.hpp

FastEratosthenes.hpp

GolbatchConjecture.hpp

Graph (8)

8.1 Matching

HopcroftKarp.hpp

Kuhn.hpp

WeightedMatching.hpp

8.2 Network Flow

Dinic.hpp

MinCostFlow.hpp

BinaryOptimization.hpp

KaryOptimization.hpp

8.3 Connectivity

SCC.hpp

LowLink.hpp

Tree (9)

HLD.hpp

CentroidDecom.hpp

Polynomials (10)

FormalPowerSeries.hpp

FFT.hpp

NTT.hpp

Strings (11)

Manacher.hpp

SuffixArray.hpp

ZAlgo.hpp

PrefixFunction.hpp

SuffixAutomaton.hpp

Geometry (12)

12.1 Geometric primitives

Point.h

lineDistance.h

SegmentDistance.h

SegmentIntersection.h

lineIntersection.h

sideOf.h

OnSegment.h

linearTransformation.h

LineProjectionReflection.h

Angle.h

12.2 Circles

CircleIntersection.h

CircleTangents.h

CircleLine.h

CirclePolygonIntersection.h

circumcircle.h

MinimumEnclosingCircle.h

12.3 Polygons

InsidePolygon.h

PolygonArea.h

PolygonCenter.h

PolygonCut.h

ConvexHull.h

HullDiameter.h

PointInsideHull.h

LineHullIntersection.h

12.4 Misc. Point Set Problems

ClosestPair.h

ManhattanMST.h

kdTree.h

DelaunayTriangulation.h

FastDelaunay.h

12.5 3D

PolyhedronVolume.h

Point3D.h

3dHull.h

sphericalDistance.h

ComplexGeometry.hpp

DefiniteIntegral.hpp

Dynamic Programming (13)

DVC.hpp

SlopeTrick.hpp

Convolutions (14)

AndConvolution.hpp

GCDCconvolution.hpp

LCMConvolution.hpp

ORConvolution.hpp

XORConvolution.hpp

MaxPlusConvolution.hpp

Various (15)

GaussianElimination.hpp

BinaryTrie.hpp

InfixPropostfix.hpp

RangeXor.hpp

15.1 LP Duality

Maximization	Minimization
Inequality constraint $\leq$	Nonnegative variable $\geq$
Inequality constraint $\geq$	Nonpositive variable $\leq$
Equality constraint $=$	Free variable
Nonnegative variable $\geq$	Inequality constraint $\geq$
Nonpositive variable $\leq$	Inequality constraint $\leq$
Free variable	Equality constraint $=$

15.2 Optimization tricks

`__builtin_ia32_ldmxcsr(40896);` disables denormals (which make floats 20x slower near their minimum value).

15.2.1 Bit hacks

- `x & -x` is the least bit in `x`.
- `for (int x = m; x; ) { --x &= m; ... }` loops over all subset masks of `m` (except `m` itself).
- `c = x&-x, r = x+c; (((r^x) >> 2)/c) | r` is the next number after `x` with the same number of bits set.
- `rep(b,0,K) rep(i,0,(1 << K)) if (i & 1 << b) D[i] += D[i^(1 << b)];` computes all sums of subsets.

15.2.2 Pragmas

- `#pragma GCC optimize ("Ofast")` will make GCC auto-vectorize loops and optimizes floating points better.
- `#pragma GCC target ("avx2")` can double performance of vectorized code, but causes crashes on old machines.
- `#pragma GCC optimize ("trapv")` kills the program on integer overflows (but is really slow).

# Competitive Programming Topics

## (A)

topics.txt

159 lines

```
Recursion
Divide and conquer
    Finding interesting points in N log N
Algorithm analysis
    Master theorem
    Amortized time complexity
Greedy algorithm
    Scheduling
    Max contiguous subvector sum
    Invariants
    Huffman encoding
Graph theory
    Dynamic graphs (extra book-keeping)
    Breadth first search
    Depth first search
    * Normal trees / DFS trees
    Dijkstra's algorithm
    MST: Prim's algorithm
    Bellman-Ford
    Konig's theorem and vertex cover
    Min-cost max flow
    Lovasz toggle
    Matrix tree theorem
    Maximal matching, general graphs
    Hopcroft-Karp
    Hall's marriage theorem
    Graphical sequences
    Floyd-Warshall
    Euler cycles
    Flow networks
    * Augmenting paths
    * Edmonds-Karp
    Bipartite matching
    Min. path cover
    Topological sorting
    Strongly connected components
    2-SAT
    Cut vertices, cut-edges and biconnected components
    Edge coloring
    * Trees
    Vertex coloring
    * Bipartite graphs (=> trees)
    * 3^n (special case of set cover)
    Diameter and centroid
    K'th shortest path
    Shortest cycle
Dynamic programming
    Knapsack
    Coin change
    Longest common subsequence
    Longest increasing subsequence
    Number of paths in a dag
    Shortest path in a dag
    Dynprog over intervals
    Dynprog over subsets
    Dynprog over probabilities
    Dynprog over trees
    3^n set cover
    Divide and conquer
    Knuth optimization
    Convex hull optimizations
    RMQ (sparse table a.k.a 2^k-jumps)
    Bitonic cycle
```

```
    Log partitioning (loop over most restricted)
Combinatorics
    Computation of binomial coefficients
    Pigeon-hole principle
    Inclusion/exclusion
    Catalan number
    Pick's theorem
Number theory
    Integer parts
    Divisibility
    Euclidean algorithm
    Modular arithmetic
    * Modular multiplication
    * Modular inverses
    * Modular exponentiation by squaring
    Chinese remainder theorem
    Fermat's little theorem
    Euler's theorem
    Phi function
    Frobenius number
    Quadratic reciprocity
    Pollard-Rho
    Miller-Rabin
    Hensel lifting
    Vieta root jumping
Game theory
    Combinatorial games
    Game trees
    Mini-max
    Nim
    Games on graphs
    Games on graphs with loops
    Grundy numbers
    Bipartite games without repetition
    General games without repetition
    Alpha-beta pruning
Probability theory
Optimization
    Binary search
    Ternary search
    Unimodality and convex functions
    Binary search on derivative
Numerical methods
    Numeric integration
    Newton's method
    Root-finding with binary/ternary search
    Golden section search
Matrices
    Gaussian elimination
    Exponentiation by squaring
Sorting
    Radix sort
Geometry
    Coordinates and vectors
    * Cross product
    * Scalar product
    Convex hull
    Polygon cut
    Closest pair
    Coordinate-compression
    Quadtrees
    KD-trees
    All segment-segment intersection
Sweeping
    Discretization (convert to events and sweep)
    Angle sweeping
    Line sweeping
    Discrete second derivatives
Strings
```

```
    Longest common substring
    Palindrome subsequences
    Knuth-Morris-Pratt
    Tries
    Rolling polynomial hashes
    Suffix array
    Suffix tree
    Aho-Corasick
    Manacher's algorithm
    Letter position lists
Combinatorial search
    Meet in the middle
    Brute-force with pruning
    Best-first (A*)
    Bidirectional search
    Iterative deepening DFS / A*
Data structures
    LCA (2^k-jumps in trees in general)
    Pull/push-technique on trees
    Heavy-light decomposition
    Centroid decomposition
    Lazy propagation
    Self-balancing trees
    Convex hull trick (wcipeg.com/wiki/Convex_hull_trick)
    Monotone queues / monotone stacks / sliding queues
    Sliding queue using 2 stacks
    Persistent segment tree
```

troubleshooting.txt

52 lines

```
Pre-submit:
Write a few simple test cases if sample is not enough.
Are time limits close? If so, generate max cases.
Is the memory usage fine?
Could anything overflow?
Make sure to submit the right file.

Wrong answer:
Print your solution! Print debug output, as well.
Are you clearing all data structures between test cases?
Can your algorithm handle the whole range of input?
Read the full problem statement again.
Do you handle all corner cases correctly?
Have you understood the problem correctly?
Any uninitialized variables?
Any overflows?
Confusing N and M, i and j, etc.?
Are you sure your algorithm works?
What special cases have you not thought of?
Are you sure the STL functions you use work as you think?
Add some assertions, maybe resubmit.
Create some testcases to run your algorithm on.
Go through the algorithm for a simple case.
Go through this list again.
Explain your algorithm to a teammate.
Ask the teammate to look at your code.
Go for a small walk, e.g. to the toilet.
Is your output format correct? (including whitespace)
Rewrite your solution from the start or let a teammate do it.

Runtime error:
Have you tested all corner cases locally?
Any uninitialized variables?
Are you reading or writing outside the range of any vector?
Any assertions that might fail?
Any possible division by 0? (mod 0 for example)
Any possible infinite recursion?
Invalidated pointers or iterators?
Are you using too much memory?
```

Debug with resubmits (e.g. remapped signals, see Various).

Time limit exceeded:  
Do you have any possible infinite loops?  
What is the complexity of your algorithm?  
Are you copying a lot of unnecessary data? (References)  
How big is the input and output? (consider scanf)  
Avoid vector, map. (use arrays/unordered\_map)  
What do your teammates think about your algorithm?

Memory limit exceeded:  
What is the max amount of memory your algorithm should need?  
Are you clearing all data structures between test cases?