Contents

1 Contest Setup

1.1 vimrc

```vim
set number        " Show line numbers
set mouse=a       " Enable inaction via mouse
set showmatch       " Highlight matching brace
set cursorline      " Show underline
set cursorcolumn     " highlight vertical column

filetype on "enable file detection
syntax on    "syntax highlight

set autoindent       " Auto-indent new lines
set shiftwidth=4     " Number of auto-indent spaces
set smartindent      " Enable smart-indent
set smarttab         " Enable smart-tabs
set tabstop=4    " Number of spaces per Tab

" ---------Optional----------

set undolevels=10000    " Number of undo levels
set scrolloff=5      " Auto scroll

set hlsearch      " Highlight all search results
set smartcase     " Enable smart-case search
set ignorecase    " Always case-insensitive
set incsearch     " Searches for strings incrementally

highlight Comment ctermfg=cyan
set showmode

set encoding=utf-8
set fileencoding=utf-8
scriptencoding=utf-8
```

1.2 bashrc

```bash
alias g++="g++ -Wall -Wextra -std=c++11 -O2"
```

1.3 Grep Error and Warnings

```bash
g++ main.cpp 2>&1 | grep -E 'warning|error'
```

1.4 C++ template

```cpp
#include <bits/stdc++.h>

using namespace std;

typedef long long int ll;
typedef pair<int, int> ii;

```

```cpp
int main()
{
    return 0;
}
```

1.5 Java template

```java
import java.io.*;
import java.util.*;

public class Main
{
    public static void main(String[] args)
    {
        MyScanner sc = new MyScanner();
        out = new PrintWriter(new BufferedOutputStream(System.out));
        // Start writing your solution here.

        // Stop writing your solution here.
        out.close();
    }

    public static PrintWriter out;

    public static class MyScanner
    {
        BufferedReader br;
        StringTokenizer st;

        public MyScanner()
        {
            br = new BufferedReader(new InputStreamReader(System.in));
        }

        boolean hasNext()
        {
            while (st == null || !st.hasMoreElements()) {
                try {
                    st = new StringTokenizer(br.readLine());
                } catch (Exception e) {
                    return false;
                }
            }
            return true;
        }

        String next()
        {
            if (hasNext())
                return st.nextToken();
            return null;
        }

        int nextInt()
        {
```

```java
49          return Integer.parseInt(next());
50      }
51
52      long nextLong()
53      {
54          return Long.parseLong(next());
55      }
56
57      double nextDouble()
58      {
59          return Double.parseDouble(next());
60      }
61
62      String nextLine()
63      {
64          String str = "";
65          try {
66              str = br.readLine();
67          } catch (IOException e) {
68              e.printStackTrace();
69          }
70          return str;
71      }
72  }
73 }
```

1.5.1 Java Issues

1. Random Shuffle before sorting: $Random\ rnd = new\ Random();\ rnd.nextInt();$
2. Use StringBuilder for large output

2 System Testing

1. Setup bashrc and vimrc
2. Look for compilation parameter and code it into bashrc
3. Test if c++ and java templates work properly on local and judge machine
4. Test "divide by 0" → RE/TLE?
5. Test heap size

3 Reminder

1. 隊友的建議，要認真聽！通常隊友的建議都會突破你盲點
2. Read the problem statements carefully. Input and output specifications and constraints are crucial!
3. Estimate the **time complexity** and **memory complexity** carefully.
4. Time penalty is 20 minutes per WA, **don't rush**!
5. Sample test cases must all be tested and passed before every submission!
6. Test the corner cases, such as 0, 1, -1. Test all edge cases of the input specification.
7. Bus error: the code has $scanf, fgets$ but have nothing to read! Check if you have early termination but didn't handle it properly.
8. Binary search? 數學算式移項合併後查詢?
9. Two Pointer <-> Binary Search
10. Directed graph connectivity -> DFS. Undirected graph -> Union Find
11. Check connectivity of the graph if the problem statement doesn't say anything
12. $longlong = int * int$ won't work!
13. Shifting for $longlongint$ should be something like $1LL << 35$
14. For continuous input problems, be sure to read in all input BEFORE terminating and start processing next the input.
15. Don't use anonymous struct

16. 因式分解

4 Topic list

1. enumeration
2. greedy
3. sorting, topological sort
4. binary search
5. 離散化
6. 矩陣快速冪

5 Useful code

5.1 Leap year

```
year % 400 == 0 || (year % 4 == 0 && year % 100 != 0)
```

5.2 Fast Exponentiation $O(log(exp))$

Fermat's little theorem: 若 m 是質數, 則 $a^{m-1} \equiv 1 \pmod{m}$

```cpp
ll fast_pow(ll a, ll b, ll M) {
    ll ans = 1;
    ll base = a % M;
    while (b) {
        if (b & 1)
            ans = ans * base % M;
        base = base * base % M;
        b >>= 1;
    }
    return ans;
}
```

5.3 Mod Inverse

Case 1: $gcd(a, m) = 1$: ax + my = gcd(a, m) = 1 (use ext_gcd)

Case 2: $m\ is\ prime$: $a^{m-2} \equiv a^{-1} mod\ m$

5.4 GCD $O(log(a + b))$

注意負數的 case! C++ 是看被除數決定正負號的。

```cpp
ll gcd(ll a, ll b)
{
    return b == 0 ? a : gcd(b, a % b);
}
```

5.5 Extended Euclidean Algorithm GCD $O(log(a + b))$

Bezout identity $ax + by = gcd(a, b)$, where $gcd(a, b)$ is the smallest positive integer that can be written as ax + by, and every integer of the form ax + by is a multiple of $gcd(a, b)$.

```
ll extgcd(ll a, ll b, ll& x, ll&y) {
    if (b == 0) {
        x = 1;
        y = 0;
        return a;
    }
    else {
        ll d = extgcd(b, a % b, y, x);
        y -= (a / b) * x;
        return d;
    }
}
```

5.6 Prime Generator

```
const ll MAX_NUM = 1e6; // 要是合数
bool is_prime[MAX_NUM];
vector<ll> primes;

void init_primes() {
    fill(is_prime, is_prime + MAX_NUM, true);
    is_prime[0] = is_prime[1] = false;
    for (ll i = 2; i < MAX_NUM; i++) {
        if (is_prime[i]) {
            primes.push_back(i);
            for (ll j = i * i; j < MAX_NUM; j += i)
                is_prime[j] = false;
        }
    }
}
```

5.7 scanf/printf reference

5.8 STL quick reference

5.8.1 Map

```
map<T1, T2> m; // iterable
void clear();
void erase(T1 key);
it find(T1 key); // <key, val>
void insert(pair<T1, T2> P);
T2 &[](T1 key); // if key not in map, new key will be inserted with
    default val
it lower_bound(T1 key); // = m.end() if not found, *it = <key, val>
it upper_bound(T1 key); // = m.end() if not found, *it = <key, val>
```

5.8.2 Set

```
set<T> s; // iterable
void clear();
size_t count(T val); // number of val in set
void erase(T val);
it find(T val); // = s.end() if not found
```

```
void insert(T val);
it lower_bound(T val); // = s.end() if not found, *it = <key, val>
it upper_bound(T val); // = s.end() if not found, *it = <key, val>
```

5.8.3 Algorithm

```
// return if i is smaller than j
comp = [&](const T &i, const T &j) -> bool;
vector<T> v;
bool any_of(v.begin(), v.end(), [&](const T &i) -> bool);
bool all_of(v.begin(), v.end(), [&](const T &i) -> bool);
void copy(inp.begin(), in.end(), out.begin());
int count(v.begin(), v.end(), int val); // number of val in v
it unique(v.begin(), v.end());          // it - v.begin() = size
// after calling, v[nth] will be n-th smallest elem in v
void nth_element(v.begin(), nth_it, bin_comp);
void merge(in1.begin(), in1.end(), in2.begin(), in2.end(), out.begin(),
    comp);
// include union, intersection, difference, symmetric_difference(xor)
void set_union(in1.begin(), in1.end(), in2.begin(), in2.end(), out.
    begin(), comp);
bool next_permutation(v.begin(), v.end());
// v1, v2 need sorted already, whether v1 includes v2
bool inclues(v1.begin(), v1.end(), v2.begin(), v2.end());
it find(v.begin(), v.end(), T val); // = v.end() if not found
it search(v1.begin(), v1.end(), v2.begin(), v2.end());
it lower_bound(v.begin(), v.end(), T val);
it upper_bound(v.begin(), v.end(), T val);
bool binary_search(v.begin(), v.end(), T val); // exist in v ?
void sort(v.begin(), v.end(), comp);
void stable_sort(v.begin(), v.end(), comp);
```

5.8.4 String

5.8.5 Priority Queue

```
bool cmp(ii a, ii b)
{
    if(a.first == b.first)
        return a.second > b.second;
    return b.first > a.first;
}

priority_queue< ii, vector<ii>, function<bool(ii, ii)> > pq(cmp);
```

6 Search

6.1 Ternary Search

6.2 折半完全列舉

能用 vector 用 vector

6.3 Two-pointer 爬行法

7 Basic data structure

7.1 1D BIT

```cpp
// BIT is 1-based
const int MAX_N = 20000; //這個記得改!
ll bit[MAX_N + 1];

ll sum(int i) {
    int s = 0;
    while (i > 0) {
        s += bit[i];
        i -= (i & -i);
    }
    return s;
}

void add(int i, ll x) {
    while (i <= MAX_N) {
        bit[i] += x;
        i += (i & -i);
    }
}
```

7.2 2D BIT

```cpp
// BIT is 1-based
const int MAX_N = 20000, MAX_M = 20000; //這個記得改!
ll bit[MAX_N + 1][MAX_M + 1];

ll sum(int a, int b) {
    ll s = 0;
    for (int i = a; i > 0; i -= (i & -i))
        for (int j = b; j > 0; j -= (j & -j))
            s += bit[i][j];
    return s;
}

void add(int a, int b, ll x) {
    // MAX_N, MAX_M 須適時調整!
    for (int i = a; i <= MAX_N; i += (i & -i))
        for (int j = b; j <= MAX_M; j += (j & -j))
            bit[i][j] += x;
}
```

7.3 Union Find

```cpp
#define N 20000 // 記得改
struct UFDS {
    int par[N];

    void init() {
```

```cpp
        memset(par, -1, sizeof(par));
    }

    int root(int x) {
        return par[x] < 0 ? x : par[x] = root(par[x]);
    }

    void merge(int x, int y) {
        x = root(x);
        y = root(y);

        if (x != y) {
            if (par[x] > par[y])
                swap(x, y);
            par[x] += par[y];
            par[y] = x;
        }
    }
}
```

7.4 Segment Tree

7.5 Sparse Table

```cpp
struct {
    int sp[MAX_LOG_N][MAX_N]; // MAX_LOG_N = ceil(lg(MAX_N))

    void build(int inp[], int n) {
        for (int j = 0; j < n; j++) {
            sp[0][j] = inp[j];
        }

        for (int i = 1; (1 << i) <= n; i++)
            for (int j = 0; j + (1 << i) <= n; j++)
                sp[i][j] =
                    min(sp[i - 1][j], sp[i - 1][j + (1 << (i - 1))]);
    }

    int query(int l, int r) { // [l, r)
        int k = floor(log2(r - l));

        return min(sp[k][l], sp[k][r - (1 << k)]);
    }
} sptb;
```

8 Dynamic Programming

9 Tree

9.1 LCA

9.2 Tree Centroid

9.3 Treap

10 Graph

10.1 Articulation point / edge

10.2 CC

10.2.1 BCC vertex

10.2.2 BCC edge

10.2.3 SCC

10.3 Shortest Path

10.3.1 Dijkatra (next-to-shortest path)

10.3.2 SPFA

10.3.3 Bellman-Ford

10.3.4 Floyd-Warshall

If diagonal numbers are negative ← cycle

10.4 MST

10.4.1 Kruskal

1. Store the graph by $(weight, (from, to))$
2. Sort the graph by $weight$
3. Start from the smallest weight, and keep adding edges that won't form a cycle with the current MST set
4. Early termination condition: $n - 1$ edges has been added, NOT size of the union-find set

10.4.2 Prim

10.5 Flow

10.5.1 Max Flow (Dinic)

10.5.2 Min-Cut

10.5.3 Min Cost Flow

10.5.4 Maximum Bipartite Graph

11 String

11.1 Rolling Hash

1. Use two rolling hashes if needed.
2. The prime for pre-calculation can be 137 and 257, for modulo can be $1e9 + 7$ and $0xdefaced$

```cpp
#define N 1000100
#define B 137
#define M 1000000007

typedef long long ll;

char inp[N];
int len;
ll p[N], h[N];

void init()
{ // build polynomial table and hash value
    p[0] = 1; // b to the ith power
    for (int i = 1; i <= len; i++) {
        h[i] = (h[i - 1] * B % M + inp[i - 1]) % M; // hash value
        p[i] = p[i - 1] * B % M;
    }
}

ll get_hash(int l, int r) // [l, r] of the inp string array
{
    return ((h[r + 1] - (h[l] * p[r - l + 1])) % M + M) % M;
}
```

11.2 KMP

```cpp
void fail()
{
    int len = strlen(pat);

    f[0] = 0;
    int j = 0;
    for (int i = 1; i < len; i++) {
        while (j != 0 && pat[i] != pat[j])
            j = f[j - 1];

        if (pat[i] == pat[j])
            j++;
```

```
13
14          f[i] = j;
15      }
16 }
17
18 int match()
19 {
20     int res = 0;
21     int j = 0, plen = strlen(pat), tlen = strlen(text);
22
23     for (int i = 0; i < tlen; i++) {
24         while (j != 0 && text[i] != pat[j])
25             j = f[j - 1];
26
27         if (text[i] == pat[j]) {
28             if (j == plen - 1) { // find match
29                 res++;
30                 j = f[j];
31             } else {
32                 j++;
33             }
34         }
35     }
36
37     return res;
38 }
```

11.3 Z Algorithm

11.4 Trie

注意 count 的擺放位置，視題意可以擺在迴圈外

```
1 #define N 600010
2 struct node {
3     int child[26];
4     bool ending;
5 } trie[N];
6
7 /*
8 root is 0
9 memset(trie, 0, sizeof(trie));
10 freeNode = 1;
11 */
12 int freeNode;
13 void insert(string &str, int pos, int node)
14 {
15     if (pos == (int)str.length()) {
16         trie[node].ending = true;
17     } else { // find which way to go
18         int c = str[pos] - 'a';
19         if (trie[node].child[c] == 0) // give a new node
20             trie[node].child[c] = freeNode++;
21         insert(str, pos + 1, trie[node].child[c]);
22     }
23 }
```

11.5 Suffix Array

12 Matrix

12.1 高斯消去法

12.2 高斯喬登

13 Geometry

1. Keep things in integers as much as possible!
2. Try not to divide
3. If you have decimals, if they are fixed precision, you can usually just multiply all the input and use integers instead

13.1 EPS

$= 0$: $fabs \leq eps$
< 0: $< -eps$
> 0: $> +eps$

13.2 Template

```
1 // if the points are given in doubles form, change the code accordingly
2
3 typedef long long ll;
4
5 typedef pair<ll, ll> pt;  // points are stored using long long
6 typedef pair<pt, pt> seg; // segments are a pair of points
7
8 #define x first
9 #define y second
10
11 #define EPS 1e-9
12
13 pt operator+(pt a, pt b)
14 {
15     return pt(a.x + b.x, a.y + b.y);
16 }
17
18 pt operator-(pt a, pt b)
19 {
20     return pt(a.x - b.x, a.y - b.y);
21 }
22
23 pt operator*(pt a, int d)
24 {
25     return pt(a.x * d, a.y * d);
26 }
27
28 ll cross(pt a, pt b)
29 {
30     return a.x * b.y - a.y * b.x;
31 }
32
33 int ccw(pt a, pt b, pt c)
34 {
```

```cpp
35        ll res = cross(b - a, c - a);
36        if (res > 0) // left turn
37            return 1;
38        else if (res == 0) // straight
39            return 0;
40        else // right turn
41            return -1;
42    }
43
44    double dist(pt a, pt b)
45    {
46        double dx = a.x - b.x;
47        double dy = a.y - b.y;
48        return sqrt(dx * dx + dy * dy);
49    }
50
51    bool zero(double x)
52    {
53        return fabs(x) <= EPS;
54    }
55
56    bool overlap(seg a, seg b)
57    {
58        return ccw(a.x, a.y, b.x) == 0 && ccw(a.x, a.y, b.y) == 0;
59    }
60
61    bool intersect(seg a, seg b)
62    {
63        if (overlap(a, b) == true) { // non-proper intersection
64            double d = 0;
65            d = max(d, dist(a.x, a.y));
66            d = max(d, dist(a.x, b.x));
67            d = max(d, dist(a.x, b.y));
68            d = max(d, dist(a.y, b.x));
69            d = max(d, dist(a.y, b.y));
70            d = max(d, dist(b.x, b.y));
71
72            // d > dist(a.x, a.y) + dist(b.x, b.y)
73            if (d - (dist(a.x, a.y) + dist(b.x, b.y)) > EPS)
74                return false;
75            return true;
76        }
77        //                      |
78        // Equal sign for -----| case
79        // non qequal sign => proper intersection
80        if (ccw(a.x, a.y, b.x) * ccw(a.x, a.y, b.y) <= 0 &&
81            ccw(b.x, b.y, a.x) * ccw(b.x, b.y, a.y) <= 0)
82            return true;
83        return false;
84    }
85
86    double area(vector<pt> pts)
87    {
88        double res = 0;
89        int n = pts.size();
90        for (int i = 0; i < n; i++)
91            res += (pts[i].y + pts[(i + 1) % n].y) * (pts[(i + 1) % n].x -
    pts[i].x);
92        return res / 2.0;
93    }
94
95    vector<pt> halfHull(vector<pt> &points)
96    {
97        vector<pt> res;
98
99        for (int i = 0; i < (int)points.size(); i++) {
100            while ((int)res.size() >= 2 &&
101                    ccw(res[res.size() - 2], res[res.size() - 1], points[i])
     < 0)
102                res.pop_back(); // res.size() - 2 can't be assign before
    size() >= 2
103            // check, bitch
104
105            res.push_back(points[i]);
106        }
107
108        return res;
109    }
110
111    vector<pt> convexHull(vector<pt> &points)
112    {
113        vector<pt> upper, lower;
114
115        // make upper hull
116        sort(points.begin(), points.end());
117
118        upper = halfHull(points);
119        // make lower hull
120        reverse(points.begin(), points.end());
121        lower = halfHull(points);
122
123        // merge hulls
124        if ((int)upper.size() > 0) // yes sir~
125            upper.pop_back();
126        if ((int)lower.size() > 0)
127            lower.pop_back();
128
129        vector<pt> res(upper.begin(), upper.end());
130        res.insert(res.end(), lower.begin(), lower.end());
131
132        return res;
133    }
134
135    bool completelyInside(vector<pt> &outer, vector<pt> &inner)
136    {
137        int even = 0, odd = 0;
138        for (int i = 0; i < (int)inner.size(); i++) {
139            // y = slope * x + offset
140            int cntIntersection = 0;
141            ll slope = rand() % INT_MAX + 1;
142            ll offset = inner[i].y - slope * inner[i].x;
143
```

```
144          ll farx = 111111 * (slope >= 0 ? 1 : -1);
145          ll fary = farx * slope + offset;
146          seg a = seg(pt(inner[i].x, inner[i].y), pt(farx, fary));
147          for (int j = 0; j < (int)outer.size(); j++) {
148              seg b = seg(outer[j], outer[(j + 1) % (int)outer.size()]);
149
150              if ((b.x.x * slope + offset == b.x.y) ||
151                  (b.y.x * slope + offset == b.y.y)) { // on-line
152                  i--;
153                  break;
154              }
155
156              if (intersect(a, b) == true)
157                  cntIntersection++;
158          }
159
160          if (cntIntersection % 2 == 0) // outside
161              even++;
162          else
163              odd++;
164      }
165
166      return odd == (int)inner.size();
167 }
168
169 // srand(time(NULL))
170 // rand()
```

14 Math

14.1 Euclid's formula (Pythagorean Triples)

$a = p^2 - q^2$
$b = 2pq$ **(always even)**
$c = p^2 + q^2$

14.2 Difference between two consecutive numbers' square is odd

$(k + 1)^2 - k^2 = 2k + 1$

14.3 Summation

$\sum_{k=1}^{n} 1 = n$
$\sum_{k=1}^{n} k = \frac{n(n+1)}{2}$
$\sum_{k=1}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}$
$\sum_{k=1}^{n} k^3 = \frac{n^2(n+1)^2}{4}$

14.4 FFT

14.5 combination

14.5.1 Pascal triangle

```
1  #define N 210
2  ll C[N][N];
3
4  void Combination() {
5      for(ll i=0; i<N; i++) {
6          C[i][0] = 1;
7          C[i][i] = 1;
8      }
9
10     for(ll i=2; i<N; i++) {
11         for(ll j=1; j<=i; j++) {
12             C[i][j] = (C[i-1][j] + C[i-1][j-1])%1000000;
13         }
14     }
15 }
```

14.5.2 線性

```
1  ll binomialCoeff(ll n, ll k)
2  {
3      ll res = 1;
4
5      if ( k > n - k ) // Since C(n, k) = C(n, n-k)
6          k = n - k;
7
8      for (int i = 0; i < k; ++i) // n...n-k / 1...k
9      {
10         res *= (n - i);
11         res /= (i + 1);
12     }
13
14     return res;
15 }
```
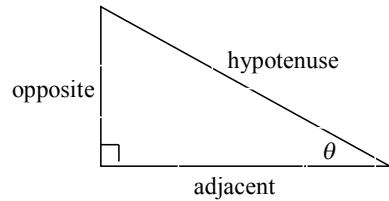
14.6 重複組合

Trig Cheat Sheet

Definition of the Trig Functions
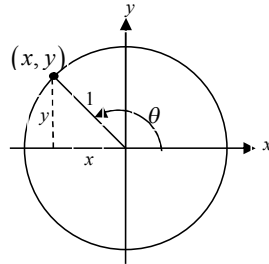
Right triangle definition

For this definition we assume that
$0 < \theta < \frac{\pi}{2}$ or $0° < \theta < 90°$.

$$\sin\theta = \frac{\text{opposite}}{\text{hypotenuse}} \qquad \csc\theta = \frac{\text{hypotenuse}}{\text{opposite}}$$

$$\cos\theta = \frac{\text{adjacent}}{\text{hypotenuse}} \qquad \sec\theta = \frac{\text{hypotenuse}}{\text{adjacent}}$$

$$\tan\theta = \frac{\text{opposite}}{\text{adjacent}} \qquad \cot\theta = \frac{\text{adjacent}}{\text{opposite}}$$

Unit circle definition

For this definition θ is any angle.

$$\sin\theta = \frac{y}{1} = y \qquad \csc\theta = \frac{1}{y}$$

$$\cos\theta = \frac{x}{1} = x \qquad \sec\theta = \frac{1}{x}$$

$$\tan\theta = \frac{y}{x} \qquad \cot\theta = \frac{x}{y}$$

Facts and Properties

Domain

The domain is all the values of θ that can be plugged into the function.

$\sin\theta$, $\quad \theta$ can be any angle

$\cos\theta$, $\quad \theta$ can be any angle

$\tan\theta$, $\quad \theta \neq \left(n + \frac{1}{2}\right)\pi$, $\ n = 0, \pm 1, \pm 2, \ldots$

$\csc\theta$, $\quad \theta \neq n\pi$, $\ n = 0, \pm 1, \pm 2, \ldots$

$\sec\theta$, $\quad \theta \neq \left(n + \frac{1}{2}\right)\pi$, $\ n = 0, \pm 1, \pm 2, \ldots$

$\cot\theta$, $\quad \theta \neq n\pi$, $\ n = 0, \pm 1, \pm 2, \ldots$

Range

The range is all possible values to get out of the function.

$-1 \leq \sin\theta \leq 1 \qquad \csc\theta \geq 1$ and $\csc\theta \leq -1$

$-1 \leq \cos\theta \leq 1 \qquad \sec\theta \geq 1$ and $\sec\theta \leq -1$

$-\infty < \tan\theta < \infty \qquad -\infty < \cot\theta < \infty$

Period

The period of a function is the number, T, such that $f(\theta + T) = f(\theta)$. So, if ω is a fixed number and θ is any angle we have the following periods.

$$\sin(\omega\theta) \ \rightarrow \ T = \frac{2\pi}{\omega}$$

$$\cos(\omega\theta) \ \rightarrow \ T = \frac{2\pi}{\omega}$$

$$\tan(\omega\theta) \ \rightarrow \ T = \frac{\pi}{\omega}$$

$$\csc(\omega\theta) \ \rightarrow \ T = \frac{2\pi}{\omega}$$

$$\sec(\omega\theta) \ \rightarrow \ T = \frac{2\pi}{\omega}$$

$$\cot(\omega\theta) \ \rightarrow \ T = \frac{\pi}{\omega}$$

Formulas and Identities

Tangent and Cotangent Identities

$$\tan\theta = \frac{\sin\theta}{\cos\theta} \qquad \cot\theta = \frac{\cos\theta}{\sin\theta}$$

Reciprocal Identities

$$\csc\theta = \frac{1}{\sin\theta} \qquad \sin\theta = \frac{1}{\csc\theta}$$

$$\sec\theta = \frac{1}{\cos\theta} \qquad \cos\theta = \frac{1}{\sec\theta}$$

$$\cot\theta = \frac{1}{\tan\theta} \qquad \tan\theta = \frac{1}{\cot\theta}$$

Pythagorean Identities

$$\sin^2\theta + \cos^2\theta = 1$$

$$\tan^2\theta + 1 = \sec^2\theta$$

$$1 + \cot^2\theta = \csc^2\theta$$

Even/Odd Formulas

$$\sin(-\theta) = -\sin\theta \qquad \csc(-\theta) = -\csc\theta$$

$$\cos(-\theta) = \cos\theta \qquad \sec(-\theta) = \sec\theta$$

$$\tan(-\theta) = -\tan\theta \qquad \cot(-\theta) = -\cot\theta$$

Periodic Formulas

If n is an integer.

$$\sin(\theta + 2\pi n) = \sin\theta \qquad \csc(\theta + 2\pi n) = \csc\theta$$

$$\cos(\theta + 2\pi n) = \cos\theta \qquad \sec(\theta + 2\pi n) = \sec\theta$$

$$\tan(\theta + \pi n) = \tan\theta \qquad \cot(\theta + \pi n) = \cot\theta$$

Double Angle Formulas

$$\sin(2\theta) = 2\sin\theta\cos\theta$$

$$\cos(2\theta) = \cos^2\theta - \sin^2\theta$$

$$= 2\cos^2\theta - 1$$

$$= 1 - 2\sin^2\theta$$

$$\tan(2\theta) = \frac{2\tan\theta}{1 - \tan^2\theta}$$

Degrees to Radians Formulas

If x is an angle in degrees and t is an angle in radians then

$$\frac{\pi}{180} = \frac{t}{x} \quad \Rightarrow \quad t = \frac{\pi x}{180} \quad \text{and} \quad x = \frac{180t}{\pi}$$

Half Angle Formulas   (alternate form)

$$\sin\frac{\theta}{2} = \pm\sqrt{\frac{1 - \cos\theta}{2}} \qquad \sin^2\theta = \frac{1}{2}\left(1 - \cos(2\theta)\right)$$

$$\cos\frac{\theta}{2} = \pm\sqrt{\frac{1 + \cos\theta}{2}} \qquad \cos^2\theta = \frac{1}{2}\left(1 + \cos(2\theta)\right)$$

$$\tan\frac{\theta}{2} = \pm\sqrt{\frac{1 - \cos\theta}{1 + \cos\theta}} \qquad \tan^2\theta = \frac{1 - \cos(2\theta)}{1 + \cos(2\theta)}$$

Sum and Difference Formulas

$$\sin(\alpha \pm \beta) = \sin\alpha\cos\beta \pm \cos\alpha\sin\beta$$

$$\cos(\alpha \pm \beta) = \cos\alpha\cos\beta \mp \sin\alpha\sin\beta$$

$$\tan(\alpha \pm \beta) = \frac{\tan\alpha \pm \tan\beta}{1 \mp \tan\alpha\tan\beta}$$

Product to Sum Formulas

$$\sin\alpha\sin\beta = \frac{1}{2}\left[\cos(\alpha - \beta) - \cos(\alpha + \beta)\right]$$

$$\cos\alpha\cos\beta = \frac{1}{2}\left[\cos(\alpha - \beta) + \cos(\alpha + \beta)\right]$$

$$\sin\alpha\cos\beta = \frac{1}{2}\left[\sin(\alpha + \beta) + \sin(\alpha - \beta)\right]$$

$$\cos\alpha\sin\beta = \frac{1}{2}\left[\sin(\alpha + \beta) - \sin(\alpha - \beta)\right]$$

Sum to Product Formulas

$$\sin\alpha + \sin\beta = 2\sin\left(\frac{\alpha + \beta}{2}\right)\cos\left(\frac{\alpha - \beta}{2}\right)$$

$$\sin\alpha - \sin\beta = 2\cos\left(\frac{\alpha + \beta}{2}\right)\sin\left(\frac{\alpha - \beta}{2}\right)$$

$$\cos\alpha + \cos\beta = 2\cos\left(\frac{\alpha + \beta}{2}\right)\cos\left(\frac{\alpha - \beta}{2}\right)$$

$$\cos\alpha - \cos\beta = -2\sin\left(\frac{\alpha + \beta}{2}\right)\sin\left(\frac{\alpha - \beta}{2}\right)$$

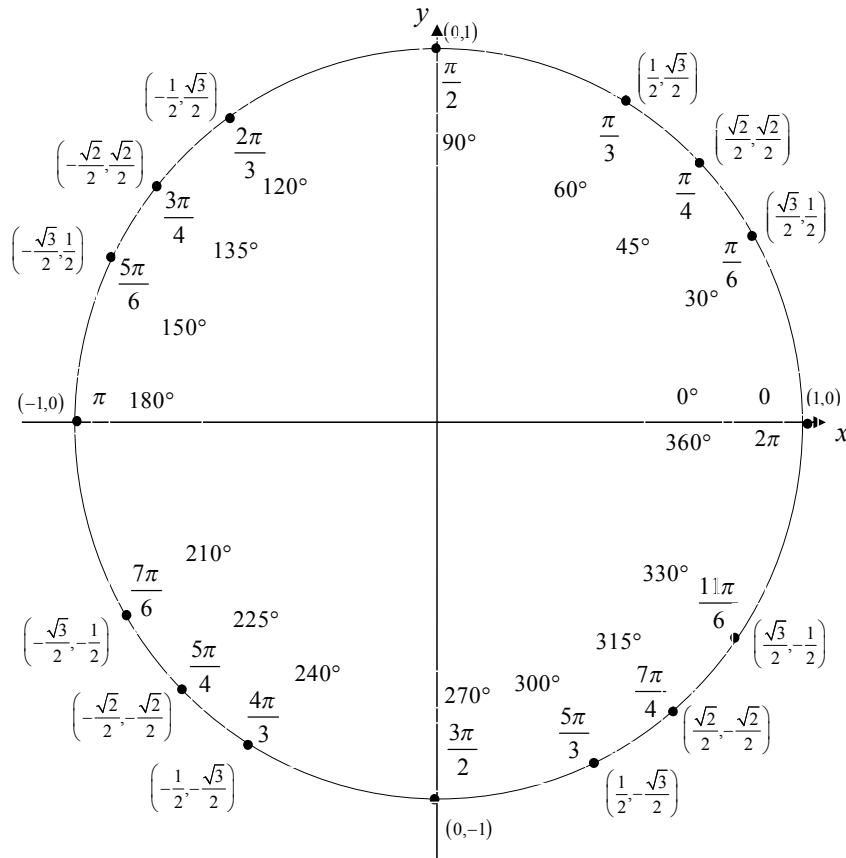**Cofunction Formulas**

$$\sin\left(\frac{\pi}{2} - \theta\right) = \cos\theta \qquad \cos\left(\frac{\pi}{2} - \theta\right) = \sin\theta$$

$$\csc\left(\frac{\pi}{2} - \theta\right) = \sec\theta \qquad \sec\left(\frac{\pi}{2} - \theta\right) = \csc\theta$$

$$\tan\left(\frac{\pi}{2} - \theta\right) = \cot\theta \qquad \cot\left(\frac{\pi}{2} - \theta\right) = \tan\theta$$

Unit Circle

For any ordered pair on the unit circle (x, y) : $\cos\theta = x$ and $\sin\theta = y$

Example

$$\cos\left(\frac{5\pi}{3}\right) = \frac{1}{2} \qquad \sin\left(\frac{5\pi}{3}\right) = -\frac{\sqrt{3}}{2}$$

Inverse Trig Functions

Definition

$y = \sin^{-1} x$ is equivalent to $x = \sin y$

$y = \cos^{-1} x$ is equivalent to $x = \cos y$

$y = \tan^{-1} x$ is equivalent to $x = \tan y$

Domain and Range

Function	Domain	Range
$y = \sin^{-1} x$	$-1 \le x \le 1$	$-\dfrac{\pi}{2} \le y \le \dfrac{\pi}{2}$
$y = \cos^{-1} x$	$-1 \le x \le 1$	$0 \le y \le \pi$
$y = \tan^{-1} x$	$-\infty < x < \infty$	$-\dfrac{\pi}{2} < y < \dfrac{\pi}{2}$

Inverse Properties

$\cos\left(\cos^{-1}(x)\right) = x \qquad \cos^{-1}\left(\cos(\theta)\right) = \theta$

$\sin\left(\sin^{-1}(x)\right) = x \qquad \sin^{-1}\left(\sin(\theta)\right) = \theta$

$\tan\left(\tan^{-1}(x)\right) = x \qquad \tan^{-1}\left(\tan(\theta)\right) = \theta$
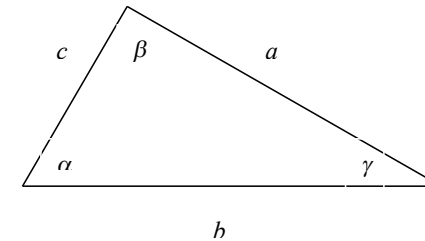
Alternate Notation

$\sin^{-1} x = \arcsin x$

$\cos^{-1} x = \arccos x$

$\tan^{-1} x = \arctan x$

Law of Sines, Cosines and Tangents

Law of Sines

$$\frac{\sin\alpha}{a} = \frac{\sin\beta}{b} = \frac{\sin\gamma}{c}$$

Law of Cosines

$$a^2 = b^2 + c^2 - 2bc\cos\alpha$$
$$b^2 = a^2 + c^2 - 2ac\cos\beta$$
$$c^2 = a^2 + b^2 - 2ab\cos\gamma$$

Mollweide's Formula

$$\frac{a+b}{c} = \frac{\cos\frac{1}{2}(\alpha - \beta)}{\sin\frac{1}{2}\gamma}$$

Law of Tangents

$$\frac{a-b}{a+b} = \frac{\tan\frac{1}{2}(\alpha - \beta)}{\tan\frac{1}{2}(\alpha + \beta)}$$

$$\frac{b-c}{b+c} = \frac{\tan\frac{1}{2}(\beta - \gamma)}{\tan\frac{1}{2}(\beta + \gamma)}$$

$$\frac{a-c}{a+c} = \frac{\tan\frac{1}{2}(\alpha - \gamma)}{\tan\frac{1}{2}(\alpha + \gamma)}$$