Template

```cpp
#include<bits/stdc++.h>
#define fi first
#define se second
#define pb push_back
#define endl '\n'

using ll = long long;
using namespace std;

void solve(ll cs){
    ll n, m, a=-1, b=0, c, x, y, k, q, i, j, mn = 1e12, mod = 1e9 + 7;

}

int main(){
    ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);

    ll t=1, cs=1;
    cin >> t;
    while(t--){
        solve(cs++);
        cout << endl;
    }
}
```

Compiler

```
{
    "cmd": ["g++ -std=c++17 -Wshadow -Wall -o -O2 -Wno-unused-result ${file} -o ${file_path}/${file_base_name} && echo '~ Build Finished, Now hit the terminal and Run the test cases!!!'  && ${file_path}/${file_base_name}"],
    "shell" : true
}
```

Number Theory

Sieve of Eratosthenes

```cpp
const int limit = 1e7 + 7;
// Sieve of Eratosthenes
// Time Complexity O(n log log n)
// can be used until 10^9
vector<bool> is_prime(limit+1, true); // define every number as prime
vector<long long> primes; // for storing the primes
void sieve_of_eratosthenes() {
    // Finding out the primes in simple way
    is_prime[0] = is_prime[1] = false;
    for (int i = 2; i * i <= limit; ++i) {
        if (is_prime[i]) {
            primes.push_back(i);
            for (int j = i * i; j <= limit; j += i) {
                is_prime[j] = false;
            }
        }
    }
}
```

Prime Factorization

```cpp
// Method 1
// faster process
// Time Complexity O(sqrt(n)/ln(sqrt(n)) + log2(n))

vector<long long> primes_factors(long long n) {
    vector<long long> factors;
    int root = sqrt(n);
    for (int i = 0; i < (int)primes.size() && primes[i] <= root; ++i) {
        if (is_prime[n]) {
            break;
        }
        if (n%primes[i] == 0) {
            while (n%primes[i] == 0) { // log2(n)
                n /= primes[i];
                factors.push_back(primes[i]);
            }
            root = sqrt(n);
        }
    }
    if (n != 1) {
        factors.push_back(n);
    }
    return factors;
}
// Method 2
// a bit slow process for big integers
vector<long long> prime_divisors(long long n) {
    vector<long long> divisor;
    while (n%2 == 0) {
        divisor.push_back(2);
        n /= 2;
    }
    for (long long d = 3; d * d <= n; d += 2) {
        while (n%d == 0) {
            divisor.push_back(d);
            n /= d;
        }
    }
    if (n > 1) {
        divisor.push_back(n);
    }
    return divisor;
}
```

Modular Multiplicable Inverse

```cpp
// A modular inverse based solution to
// compute nCr % p
#include <bits/stdc++.h>
using namespace std;

/* Iterative Function to calculate (x^y)%p in O(log y) */
unsigned long long power(unsigned long long x, int y, int p)
{
    unsigned long long res = 1;
    x = x % p;
    while (y > 0)
    {
        if (y & 1) res = (res * x) % p;

        y = y >> 1; // y = y/2
        x = (x * x) % p;
    }
    return res;
}

// Returns n^(-1) mod p
unsigned long long modInverse(unsigned long long n, int p)
{
    return power(n, p - 2, p);
}

// Returns nCr % p using Fermat's little
// theorem.
unsigned long long nCrModPFermat(unsigned long long n, int r, int p)
{
    if (n < r) return 0;
    if (r == 0) return 1;

    unsigned long long fac[n + 1];
    fac[0] = 1;
    for (int i = 1; i <= n; i++)  fac[i] = (fac[i - 1] * i) % p;

    return (fac[n] * modInverse(fac[r], p) % p * modInverse(fac[n - r], p) % p) % p;
}

int main()
{
    int n = 10, r = 2, p = 13;
    cout << "Value of nCr % p is " << nCrModPFermat(n, r, p);
    return 0;
}
```

EulersTotient

```cpp
#include <bits/stdc++.h>
using namespace std;
const int maxn = (int)1e5 + 7;
int phi[maxn];
// Time Complexity - O(n log log n)
void phi_1_to_n() {
    for (int i = 0; i <= maxn; ++i) {
        phi[i] = i;
    }
    for (int i = 2; i <= maxn; ++i) {
        if (phi[i] == i) {
            for (int j = i; j <= maxn; j += i) {
                phi[j] -= phi[j]/i;
            }
        }
    }
}
// sum of coprimes until n
int sum_of_coprimes_untill_n(int n) {
    return (phi[n]/2) * n;
}
int main(int argc, char const *argv[]) {
    ios_base::sync_with_stdio(false), cin.tie(nullptr);
    phi_1_to_n();
    int n;
    cin >> n;
    for (int i = 2; i < 13; ++i) {
        cout << phi[i] << ' ';
    }
    cout << '\n';
    cout << sum_of_coprimes_untill_n(n) << '\n';
    return 0;
}
```

Graphs Traversal

BFS

```cpp
#include<bits/stdc++.h>
#define endl '\n'

using ll = long long;
using namespace std;

vector<ll> v[100005];
ll vis[100005], d[100005];

void bfs(ll x){
    ll n;
    queue<ll> q;
    for(ll i=0;i<100005;i++) vis[i] = 0;
    q.push(x);
    d[x] = 0;
    while(!q.empty()){
        x = q.front();
        vis[x] = 1;
        q.pop();
        for(auto xx : v[x]){
            if(vis[xx]==0){
                vis[xx] = 1;
                q.push(xx);
                d[xx] = d[x] + 1;
            }
        }
    }
}

void solve(ll cs){
    ll m, i, h, w, e, j, a=1, n, k=0, x, y, b, l=0, r=0, ans=2, mod=998244353;
    string s;
    cin >> n >> k;

    while(k--){
        cin >> x >> y;
        v[y].push_back(x);
        v[x].push_back(y);
    }

    bfs(1);
    for(i=0;i<10;i++) cout << d[i] << " ";

}
```

DFS

```cpp
#include<bits/stdc++.h>
#define fi first
#define se second
#define endl '\n'

using ll = long long;
using namespace std;

ll mod = 1e9+7;
vector<ll> v[1000006];

void dfs(ll i, vector<ll> &col, vector<ll> &d, vector<ll> &f, vector<ll> &par, ll
&time){
    col[i] = 1;
    d[i] = ++time;
    for(auto x : v[i]){
        if(col[x]==0) {
            par[x] = i;
            dfs(x, col, d, f, par, time);
        }
    }

    col[i] = 2;
    f[i] = ++time;
}

void solve(ll cs){

    ll n, m, i, j;
    cin >> n >> m;

    while(m--){
        cin >> i >> j;
        v[i].push_back(j);
    }

    ll time = 0;
    vector<ll> col(n+1, 0), d(n+1), f(n+1), par(n+1, NULL);
    for(i=1;i<=n;i++) if(col[i]==0) dfs(i, col, d, f, par, time);

    vector<pair<ll, ll>> vs;
    for(i=1;i<=n;i++) vs.push_back({f[i], i});
    sort(vs.rbegin(), vs.rend());

    time = 0;
    col = vector<ll> (n+1, 0);
    for(auto x : vs){
        if(col[x.se]==0) dfs(x.se, col, d, f, par, time);
    }

    for(i=1;i<=n;i++) cout << d[i] << " " << f[i] << " " << par[i] << endl;

}
```

Dijkstra

```cpp
#include<bits/stdc++.h>
#define endl '\n'

using namespace std;
using ll = long long;

vector<vector<pair<ll, ll>>> v;
map<ll, ll> dis;

ll dijkstra(ll i){
    priority_queue<pair<ll, ll>, vector<pair<ll, ll>>, greater<pair<ll, ll>>> pq;
    dis[i] = 0;
    pair<ll, ll> pi;
    pq.push({0, i});
    while(!pq.empty()){
        pi = pq.top();
        pq.pop();
        ll u = pi.second;
        for(auto x : v[u]){
            if(dis[u] + x.second < dis[x.first]){
                dis[x.first] = dis[u] + x.second;
                pq.push({dis[x.first], x.first});
            }
        }
    }
}

int main(){

    ll n, m, i, j;

    cin >> n >> m;
    v = vector<vector<pair<ll, ll>>> (n+1, vector<pair<ll, ll>>(0));

    for(i=1;i<=n;i++) dis[i] = 1e12;
    for(i=0;i<m;i++){
        ll u, vv, cost;
        cin >> u >> vv >> cost;
        v[u].push_back({vv, cost});
        v[vv].push_back({u, cost});
    }

    ll ans = dijkstra(1);

    for(auto x : dis) cout << x.second << " ";

}
```

Prim MST

```cpp
#include<bits/stdc++.h>
#define fi first
#define se second
#define endl '\n'
using ll = long long;
using namespace std;

ll mod = 1e9+7;
vector<pair<ll, ll>> v[1000006];

double primmst(ll i, ll n){
    priority_queue<pair<ll, ll>, vector<pair<ll, ll>>, greater<pair<ll, ll>>> pq;

    vector<ll> key(n+1, 1e9);
    vector<ll> par(n+1, -1);
    vector<bool> inmst(n+1, false);

    ll src = 1, tot = 0;
    key[src] = 0;
    pq.push({key[src], src});

    while(!pq.empty()){
        pair<ll, ll> pi = pq.top();
        pq.pop();

        if(inmst[pi.se]) continue;
        inmst[pi.se] = true;
        tot += pi.fi;

        for(auto x : v[pi.se]){
            if(!inmst[x.fi] and key[x.fi]>x.se){
                key[x.fi] = x.se;
                pq.push({key[x.fi], x.fi});
                par[x.fi] = pi.se;
            }
        }
    }
    for(i=2;i<n+1;i++) cout << par[i] << " " << i << endl;
    return tot;
}

void solve(ll cs){
    ll n, m=0, j, q, k=0, i=0, t, x=0, y, a, b, d, c;
    cin >> n >> m;
    while(m--){
        cin >> a >> b >> c;
        v[a].push_back({b, c});
        v[b].push_back({a, c});
    }
    ll ans = primmst(1, n);
    cout << ans << endl;
}
```

Topological Sort

```cpp
#include<bits/stdc++.h>
#define endl '\n'

using namespace std;
using ll = long long;

vector<ll> v[1000000];

void toposort(ll i, vector<bool> &vis, stack<ll> &st){
    vis[i] = true;

    for(auto x : v[i]){
        if(!vis[x]) toposort(x, vis, st);
    }

    st.push(i);
}

void solve(ll cs){
    ll i, n, j, m, a, b;

    cin >> n >> m;
    vector<bool> vis(n, false);

    while(m--){
        cin >> a >> b;
        v[a].push_back(b);
    }

    stack<ll> st;

    for(i=0;i<n;i++) {
        if(!vis[i]) toposort(i, vis, st);
    }

    while(!st.empty()){
        cout << st.top() << endl;
        st.pop();
    }
}
```

Disjoint Set Union Find

```
/* DISJOIN SET UNION FIND*/

#include<bits/stdc++.h>
#define ll long long
#define endl '\n'

using namespace std;

vector<ll> par;

ll fd(ll r){
    if(r==par[r]) return r;
    par[r] = fd(par[r]);
    return par[r];
}

void uni(ll a, ll b){
    ll u = fd(a);
    ll v = fd(b);
    if(u==v){
        cout << "They are already friends" << endl;
    }
    else{
        par[u] = v;
    }
}

int main(){
    ios_base::sync_with_stdio(false); cin.tie(NULL); cout.tie(NULL);
    ll n, i, x;
    cin >> n;
    for(i=0;i<n+1;i++) {
        par.push_back(i);
    }
    uni(1,2);
    uni(2,3);
    uni(4,5);
    uni(5,3);
    cout << fd(1) << endl;
}
```

```cpp
/* Prefix Trie */
#include<bits/stdc++.h>
#define ll long long
#define endl '\n'
using namespace std;
struct node{
    bool endmark;
    node *next[26+1];
    node(){
        for(ll i=0;i<26;i++) next[i] = NULL;
        endmark = false;
    }
};
node *root;
void insert(string s){
    ll n = s.size();
    node *curr = root;
    for(ll i=0;i<n;i++){
        if(curr->next[s[i]-'a']==NULL) curr->next[s[i]-'a'] = new node();
        curr = curr->next[s[i]-'a'];
    }
    curr->endmark = 1;
}
bool search(string s){
    ll n = s.size();
    node *curr = root;
    for(ll i=0;i<n;i++){
        if(curr->next[s[i]-'a']==NULL) return false;
        curr = curr->next[s[i]-'a'];
    }
    return curr->endmark;
}
void del(node* curr){
    for(ll i=0;i<26;i++){
        if(curr->next[i]!=NULL) del(curr->next[i]);
    }
    delete (curr);
}
int main(){
    root = new node();
    ll i, n; cin >> n;
    for(i=0;i<n;i++){
        string s;
        cin >> s;
        insert(s);
    }
    ll q; cin >> q;
    while(q--){
        string s;
        cin >> s;
        if(search(s)) cout << "Found" << endl;
        else cout << "Not Found" << endl;
    }
    del(root);
}
```

Segment Tree

```cpp
#include<bits/stdc++.h>
#define ll long long
#define endl '\n'
using namespace std;
struct node{
    ll val, prop;
};
vector<node> seg(1000001);
vector<ll> arr;

void init(ll node, ll l, ll r){
    if(l==r){
        seg[node].val = arr[l];
        seg[node].prop = 0;
        return;
    }
    ll mid = (l+r)/2;
    init(2*node, l, mid);
    init(2*node+1, mid+1, r);
    seg[node].val = seg[2*node].val + seg[2*node+1].val;
    seg[node].prop = 0;
}

ll query(ll node, ll l, ll r, ll i, ll j, ll carry = 0){
    if(r<i or l>j) return 0;
    if(i<=l and r<=j) return seg[node].val + carry*(r-l+1);
    ll mid = (l+r)/2;
    ll x = query(2*node, l, mid, i, j, carry+seg[node].prop);
    ll y = query(2*node+1, mid+1, r, i, j, carry+seg[node].prop);
    return x+y;
}

void update(ll node, ll l, ll r, ll i, ll j, ll k){
    if(i<=l and r<=j){
        seg[node].prop = k;
        seg[node].val += (r-l+1)*k;
        return;
    }
    if(j<l or i>r) return;
    ll mid = (l+r)/2;
    update(2*node, l, mid, i, j, k);
    update(2*node+1, mid+1, r, i, j, k);
    seg[node].val = seg[2*node].val + seg[2*node+1].val + (r-l+1)*seg[node].prop;
}

int main(){
    ll i, n, j, k; cin >> n;
    arr = vector<ll>(n);
    for(auto &x : arr) cin >> x;
    init(1, 0, n-1);
    k = query(1, 0, n-1, 0, 6);
    cout << k << endl;
    update(1, 0, n-1, 4, 4, 10);
    k = query(1, 0, n-1, 0, 6); cout << k << endl;
}
```

Coin DP

```cpp
#include<bits/stdc++.h>
using ll = long long;
using namespace std;

vector<ll> v;
ll mem[1000][1000];

ll dp(ll i, ll n, ll k){
    if(i==n and k!=0) return 1e9;
    if(k==0) return 0;
    if(mem[i][k]!=0) return mem[i][k];

    ll r1 = 1e9, r2 = 1e9;
    if(k-v[i]>=0) r1 = 1+dp(i+1, n, k-v[i]);
    r2 = dp(i+1, n, k);

    return mem[i][k] = min(r1, r2);

}

int main(){

    ll i, n, j, k, ans;

    cin >> n >> k;
    v = vector<ll>(n);
    for(auto &x : v) cin >> x;

    ans = dp(0, n, k);
    cout << ans << endl;
}
```

Coin not greater than K

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

using ll = long long;
using namespace std;

vector<ll> v;
ll mem[1000][1000];

ll dpop(ll i, ll n, ll w, ll k){
    if(w<0) return 1e9;
    if(i==n and w!=0) return 1e9;
    if(w==0) return 0;
    if(mem[i][w]!=0) return mem[i][w];

    ll ans = 1e9;
    for(ll j=0;j<=k;j++){
        ans = min(ans, j+dpop(i+1, n, w-j*v[i], k));
    }

    return mem[i][w] = ans;
}

int main(){

    ll i, n, j, k, w, ans;

    cin >> n >> w >> k;
    v = vector<ll>(n);
    for(auto &x : v) cin >> x;

    ans = dpop(0, n, w, k);
    cout << ans << endl;

}
```

Coin opt

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

using ll = long long;
using namespace std;

vector<ll> v;
ll mem[10000];

ll dpop(ll n, ll k){
    if(k<0) return 1e9;
    if(k==0) return 0;
    if(mem[k]!=0) return mem[k];

    ll ans = 1e9;
    for(ll i=0;i<n;i++){
        ans = min(ans, 1+dpop(n, k-v[i]));
    }

    return mem[k] = ans;

}

int main(){

    ll i, n, j, k, ans;

    cin >> n >> k;
    v = vector<ll>(n);
    for(auto &x : v) cin >> x;

    ans = dpop(n, k);
    cout << ans << endl;

}
```

Edit Dis

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

using namespace std;
using ll = long long;

int n, m;
vector<vector<ll>> mem(1000, vector<ll> (1000, -1));

int dp(int i, int j, string s1, string s2){
    if(i==n) return m-j;
    if(j==m) return n-i;

    if(mem[i][j]!=-1) return mem[i][j];

    int ans = 0;
    if(s1[i]==s2[j]) ans = dp(i+1, j+1, s1, s2);
    else{
        ans = 1 + min(dp(i+1, j, s1, s2),
                      min(dp(i, j+1, s1, s2), dp(i+1, j+1, s1, s2)));
    }
    return mem[i][j] = ans;
}

int main(){
    ll i, j;

    string s1, s2;
    cin >> s1 >> s2;

    n = s1.size();
    m = s2.size();

    int ans = dp(0, 0, s1, s2);

    cout << ans << endl;

}
```

Knapsack

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

using ll = long long;
using namespace std;

vector<ll> pv, wv;
ll mem[1000][1000];

ll dpop(ll i, ll n, ll w){
    if(i==n) return 0;
    if(w==0) return 0;
    if(mem[i][w]!=0) return mem[i][w];

    ll r1=0, r2=0;
    if(w-wv[i]>=0) r1 = pv[i] + dpop(i+1, n, w-wv[i]);
    r2 = dpop(i+1, n, w);

    return mem[i][w] = max(r1, r2);
}

int main(){

    ll i, num_of_int, j, k, weight, ans;

    cin >> num_of_int >> weight;
    pv = vector<ll>(num_of_int);
    wv = vector<ll>(num_of_int);
    for(auto &x : pv) cin >> x;
    for(auto &x : wv) cin >> x;

    ans = dpop(0, num_of_int, weight);
    cout << ans << endl;

}
```

LCS

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

using namespace std;
using ll = long long;

int main(){
    ll i, j, n, m;

    string s1, s2;
    cin >> s1 >> s2;

    n = s1.size();
    m = s2.size();

    vector<vector<ll>> mem(n+1, vector<ll> (m+1, 0));

    for(i=n-1;i>-1;i--){
        for(j=m-1;j>-1;j--){
            if(s1[i]==s2[j]){
                mem[i][j] = 1 + mem[i+1][j+1];
            }
            else{
                mem[i][j] = max(mem[i+1][j], mem[i][j+1]);
            }
        }
    }

    cout << mem[0][0] << endl;

}
```

LIS

```cpp
#include<bits/stdc++.h>
using ll = long long;
using namespace std;

int main(){
    ll i, n, j, k, ans;
    cin >> n;

    vector<ll> v(n);
    for(auto &x : v) cin >> x;

    vector<ll> mem(n, -1), lis, next(n);

    for(i=n-1;i>-1;i--){
        ll tmp = 1, pos = i;
        for(j=i+1;j<n;j++){
            if(v[i]<v[j] and tmp<mem[j]+1) tmp = mem[j]+1, pos = j;
        }
        mem[i] = tmp;
        next[i] = pos;
    }

    ans = 0;
    for(i=0;i<n;i++) {
        if(ans<mem[i]){
            ans = mem[i];
            k = i;
        }
    }

    cout << ans << endl;
    cout << v[k] << " ";
    while(k!=next[k]){
        k = next[k];
        cout << v[k] << " ";
    }
}
```

String

KMP (Knuth Morris Pattern)

```
#include <bits/stdc++.h>
using namespace std;
// Time Complexity - O(m + n)
vector<int> prefix_functiion(string s) {
  int n = (int)s.size();
  vector<int> pi(n, 0);
  for (int i = 1; i < n; ++i) {
    int j = pi[i - 1];
    while (j > 0 && s[i] != s[j]) {
      j = pi[j - 1];
    }
    if (s[i] == s[j]) {
      ++j;
    }
    pi[i] = j;

  }
  return pi;
}
int main(int argc, char const *argv[]) {
    ios_base::sync_with_stdio(false), cin.tie(nullptr);
    string s = "na";
    vector<int> prefix = prefix_functiion(s);
    string t = "apnacollege";
    int pos = -1;
    int i = 0, j = 0;
    while (i < (int)t.size()) {
        if (t[i] == s[j]) {
            ++j;
            ++i;
        }
        else {
            if (j != 0) {
                j = prefix[j - 1];
            }
            else {
                ++i;
            }
        }
        if (j == (int)s.size()) {
            pos = i - (int)s.size();
            break;
        }
    }
    cout << pos << '\n';
    return 0;
}
```

Sparse Table

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

using namespace std;
const int MAX_N = 1e5 + 5;
const int LOG = 17;
int a[MAX_N];
int m[MAX_N][LOG];
int bin_log[MAX_N];

int query(int L, int R) {
    int len = R - L + 1;
    int k = bin_log[len];
    return min(m[L][k], m[R - (1 << k) + 1][k]);
}
int main(int argc, char const *argv[]) {
    ios_base::sync_with_stdio(false), cin.tie(nullptr);
    int n;
    cin >> n;
    // finding the logarithmic number
    bin_log[1] = 0;
    for (int i = 2; i <= n; ++i) {
        bin_log[i] = bin_log[i/2] + 1;
    }
    for (int i = 0; i < n; ++i) {
        cin >> a[i];
        m[i][0] = a[i];
    }
    // Preprocessing O(N*log(N))
    for (int k = 1; k < LOG; ++k) {
        for (int i = 0; i + (1 << k) - 1 < n; ++i) {
            m[i][k] = min(m[i][k - 1], m[i + (1 << (k - 1))][k - 1]);
        }
    }
    // answering query
    int q;
    cin >> q;
    while (q--) {
        int L, R;
        cin >> L >> R;
        cout << query(L, R) << '\n';
    }
    return 0;
}
```