Mahadi Hasan Fardin

mahadihasanfardin2015@gmail.com

Abstract

A compact guide to essential C++ libraries and functions commonly used in competitive programming, including fast I/O, data structures like strings, vectors, sets, and maps, key algorithms for sorting and searching, and mathematical utilities such as GCD, LCM, and power calculations.

CoU\_BitBusters

Code. Compete. Conquer.

|  |  |  |  |
| --- | --- | --- | --- |
| Category | Header | Function | Description |
| Input/Output | <iostream> | cin | Standard input. |
|  |  | cout | Standard output. |
|  |  | cerr | Standard error output. |
|  |  | endl | Line break with flush. |
|  | <iomanip> | setprecision(n) | Set decimal precision to n. |
|  |  | fixed | Fixed-point notation. |
|  |  | setw(n) | Set field width to n. |
|  | <cstdio> | printf() | Print formatted output. |
|  |  | scanf() | Read formatted input. |
| Strings | <string> | s.length()/s.size() | Get the length of the string. |
|  |  | s.substr(pos, len) | Extract substring starting from pos of length len. |
|  |  | s.find(str) | Find the first occurrence of str in the string. |
|  |  | s.compare(str) | Compare two strings. |
|  |  | s.append(str) | Append str to the string. |
|  |  | s.erase(pos, len) | Remove characters from position pos of length len. |
|  |  | s.insert(pos, str) | Insert str at position pos. |
|  |  | s.replace(pos, len, str) | Replace characters from pos with str |
|  |  | s.back() / s.front() | Get the last / first character. |
|  | <cstring> | strcmp(s1, s2) | Compare two C-strings. |
|  |  | strcpy(dest, src) | Copy C-string src to dest. |
|  |  | strlen(s) | Get the length of a C-string |
| Vectors | <vector> | v.push\_back(value) | Add an element to the end of the vector. |
|  |  | v.pop\_back() | Remove the last element of the vector. |
|  |  | v.size() | Get the number of elements in the vector. |
|  |  | v.clear() | Remove all elements from the vector. |
|  |  | v.begin() / v.end() | Iterators pointing to the beginning/end of the vector. |
|  |  | v.insert(pos, value) | Insert value at the position pos. |
|  |  | v.erase(pos) | Remove element at position pos. |
|  |  | v.front() / v.back() | Access the first / last element. |
|  |  | v.reserve(n) | Reserve space for n elements in advance. |
|  | <array> | a.fill(value) | Fill the entire array with value. |
|  |  | a.size() | Get the size of the array. |
| Algorithms | <algorithm> | sort(start, end) | Sort the range [start, end). |
|  |  | reverse(start, end) | Reverse the range [start, end). |
|  |  | max(a, b) / min(a, b) | Return the maximum/minimum of a and b. |
|  |  | binary\_search(start, end, value) | Check if value exists in the sorted range [start, end). |
|  |  | lower\_bound(start, end, value) | Return iterator to the first element >= value in the range [start, end). |
|  |  | upper\_bound(start, end, value) | Return iterator to the first element > value in the range [start, end). |
|  |  | next\_permutation(start, end) | Generate the next lexicographical permutation. |
|  |  | prev\_permutation(start, end) | Generate the previous lexicographical permutation. |
|  |  | rotate(start, mid, end) | Rotate the range [start, end) around mid. |
|  |  | accumulate(start, end, init) | Sum of the range [start, end) starting with init. |
| Mathematics | <cmath> | sqrt(x) | Square root of x. |
|  |  | pow(x, y) | x raised to the power y |
|  |  | ceil(x) / floor(x) | Ceiling / floor value of x |
|  |  | abs(x) | Absolute value of x |
|  |  | gcd(a, b) | Compute the greatest common divisor of a and b. |
|  |  | lcm(a, b) | Compute the least common multiple of a and b. |
|  |  | log(x) / log10(x) | Natural / base-10 logarithm of x |
|  |  | sin(x) / cos(x) / tan(x) | Trigonometric functions. |
|  |  | exp(x) | Exponential function e^x. |
|  |  | round(x) | Round x to the nearest integer. |
| Sets and Maps | <set> | s.insert(value) | Insert value into the set. |
|  |  | s.erase(value) | Erase value from the set |
|  |  | s.find(value) | Find if value exists in the set |
|  | <unordered\_set> | us.insert(value) | Insert value into an unordered set. |
|  | <map> | m[key] | Access or modify the value associated with key. |
|  |  | m.insert({key, value}) | Insert a key-value pair. |
|  |  | m.erase(key) | Erase the key-value pair with key |
|  |  | m.find(key) | Find the iterator for key. |
|  | <unordered\_map> | um[key] | Access or modify the value in an unordered map. |
| Deque | <deque> | dq.push\_back(value) | Add value to the end of the deque |
|  |  | dq.push\_front(value) | Add value to the front of the deque. |
|  |  | dq.pop\_back() | Remove the last element of the deque. |
|  |  | dq.pop\_front() | Remove the first element of the deque. |
|  |  | dq.front() / dq.back() | Access the first / last element of the deque. |
| Priority Queue | <queue> | pq.push(value) | Insert value into the priority queue. |
|  |  | pq.top() | Get the maximum element. |
|  |  | pq.pop() | Remove the maximum element. |
|  |  | q.push(value) | Enqueue value into the queue. |
|  |  | q.front() / q.back() | Access the first / last element. |
|  |  | q.pop() | Dequeue the first element |