Team notebook

HCMUS-PenguinSpammers

January 26, 2022

Contents

1 A Contest

1.1 bashrc

```
// alias c='g++ -Wall -Wconversion
    -Wfatal-errors -g -std=c++17 \
//     -fsanitize=undefined,address'
// xmodmap -e 'clear lock' -e 'keycode
    66=less greater' #caps = <>
```

1.2 hash

```
// # Hashes a file, ignoring all whitespace
    and comments. Use for
// # verifying that code was correctly typed.
// cpp -dD -P -fpreprocessed | tr -d
    '[:space:]'| md5sum |cut -c-6
```

1.3 template

```
#include <bits/stdc++.h>
using namespace std;

#define rep(i, a, b) for(int i = a; i < (b);
    ++i)
#define all(x) begin(x), end(x)
#define sz(x) (int)(x).size()
typedef long long ll;
typedef pair<int, int> pii;
typedef vector<int> vi;

int main() {
      ios_base::sync_with_stdio(false);
    cin.tie(0);
    cout.tie(0);
}
```

1.4 troubleshoot

```
// 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?
```

1.5 vimrc

```
// set cin aw ai is ts=4 sw=4 tm=50 nu noeb
    bg=dark ru cul
// sy on  |  im jk <esc>  |  im kj <esc>  |
    no ; :
// " Select region and then type :Hash to
    hash your selection.
// " Useful for verifying that there aren't
    mistypes.
// ca Hash w !cpp -dD -P -fpreprocessed \| tr
    -d '[:space:]' \
//  \| md5sum \| cut -c-6
```

2 Combinatorial

2.1 Lucas Theorem

For non-negative integers m and n and a prime p, the following congruence relation holds: :

$$\binom{m}{n} \equiv \prod_{i=0}^{k} \binom{m_i}{n_i} \pmod p,$$

where :

$$m = m_k p^k + m_{k-1} p^{k-1} + \cdots + m_1 p + m_0,$$

and :

$$n = n_k p^k + n_{k-1} p^{k-1} + \cdots + n_1 p + n_0$$

are the base p expansions of m and n respectively. This uses the convention that $\binom{m}{n} = 0$ if $m \leq n$.