

Ivan Franko National University of Lviv

Stallions

Maksym Shcherba, Petro Tarnavskyi, Yarema Stiahar

Ivan Franko National University of Lviv, LNU Stallions

template compilation s hash

Contest (1)

```
template.hpp
                                                           28 lines
// hash = 9fd99b
#include <bits/stdc++.h>
using namespace std;
#define FOR(i, a, b) for(int i = (a); i < (b); i++)
#define RFOR(i, a, b) for(int i = (a) - 1; i >= (b); i--)
#define SZ(a) int(a.size())
#define ALL(a) a.begin(), a.end()
#define PB push_back
#define MP make_pair
#define F first
#define S second
typedef long long LL;
typedef vector<int> VI;
typedef vector<LL> VL;
typedef pair<int. int> PII:
typedef pair<LL, LL> PLL;
typedef double db;
int main()
  ios::sync_with_stdio(0);
  cin.tie(0);
  return 0;
```

compilation.txt

```
g++ -02 -std=c++20 -Wno-unused-result -Wshadow -Wall -o %e %e.
cpp
g++ -std=c++20 -Wshadow -Wall -o %e %e.cpp -fsanitize=address -
fsanitize=undefined -D_GLIBCXX_DEBUG -g
```

s.sh

```
for((i = 0; ; i++)) do
echo $i
./gen $i > in
diff -w <(./a < in) <(./brute < in) || break
[ $? == 0 ] || break
done
```

hash.sh

Rules

Reject incorrect solutions from your teammates. Try to find counterexamples.

Discuss implementation and try to simplify the solution.

Avoid getting stuck on the problem.

Regularly discuss how many problems need to be solved and what steps to take, starting from the middle of the contest.

At the end of the contest, try to find a problem with an easy implementation.

Troubleshoot

Pre-submit

F9. Create a few manual test cases. Calculate time and memory complexity. Check the limits. Be careful with overflows, constants, clearing mutitestcases, uninitialized variables.

Wrong answer

F9. Print your solution! Read your code. Check pre-submit. Are you sure your algorithm works? Think about precision errors and hash collisions. Have you understood the problem correctly? Write the brute and the generator.

Runtime error

F9. Print your solution! Read your code. F9 with generator. Memory limit exceeded.

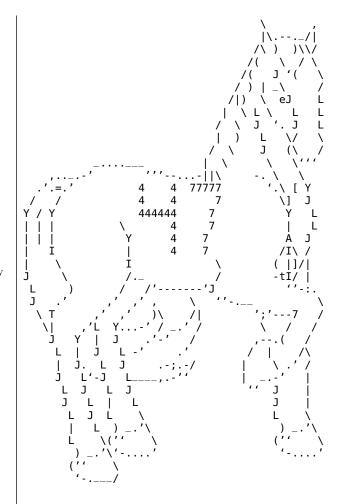
Time limit exceeded

What is the complexity of your algorithm? Are you copying a lot of unnecessary data? (References) Do you have any infinite loops? Use arrays, unordered maps instead of vectors and maps.

Pragmas

1 lines

- **#pragma** GCC optimize ("Ofast") will make GCC auto-vectorize loops and optimizes floating points better. It is not unexpected to see your floating-point error analysis go to waste.
- #pragma GCC target ("avx2") can double performance of vectorized code, but causes crashes on old machines.
- #pragma GCC optimize("unroll-loops") enables aggressive loop unrolling, which reduces the number of branches and optimizes parallel computation.



1

```
Data Structures (2)
dsu.hpp
                                                        f097c2, 31 lines
struct DSU
int n;
 VI p, sz;
  DSU(int _n = 0)
    n = _n;
    p.resize(n);
iota(ALL(p), 0);
sz.assign(n, 1);
  int find(int v)
    if (v == p[v])
    return v;
return p[v] = find(p[v]);
  bool unite(int u, int v)
    u = find(u);
v = find(v);
    if (u == v)
      return false;
    if (sz[u] > sz[v])
      swap(u, v);
    p[u] = v;
    sz[v] += sz[u];
    return true;
};
fenwick.hpp
                                                        6a7a21, 20 lines
struct Fenwick
  int n;
  VL t;
  Fenwick(int _n = 0): n(_n), t(n) {}
  void upd(int i, LL x)
    for (; i < n; i |= i + 1)
      t[i] += x;
  LL query(int i)
    LL ans = 0;
    for (; i \ge 0; i = (i \& (i + 1)) - 1)
     ans += t[i];
    return ans;
};
```