University of Copenhagen

# string & a long

Elias Rasmussen Lolck, Thor Vejen Eriksen, William Bille Meyling

ICPC European Championship 2024

July 30, 2024

# Setup

### hash.sh

5246ca

```
d41d8c# hashes a file, ignoring whitespaces and comments
d41d8c# use for verifying that code is copied correctly
d41d8ccpp -dD -P -fpreprocessed | tr -d '[:space:]' | md5sum |
       cut -c-6
```

### template

8065a4

```
d41d8c// #include <bits/stdc++.h>
d41d8cusing namespace std;
d41d8ctypedef long long ll;
d41d8c#define all(x) (x).begin(), (x).end()
d41d8c#define vi vector <int>
d41d8c#define vl vector <long long>
d41d8c#define vvi vector <vector <int>>
d41d8c#define pii pair <int, int>
d41d8c#define siz(v) (int) (v).size()
d41d8c
d41d8cint main() {
d41d8c  ios::sync_with_stdio(0); cin.tie(0);
d41d8c}
```

# Data_structures

### Disjoint Set Union

**Description**: Classic `DSU` using path compression and union by rank. `unite` returns true iff $u$ and $v$ were disjoint.
**Usage**: `Dsu d(n); d.unite(u, v); d.find(u);`
**Complexity**: `find()`, `unite()` are amortized $\mathcal{O}(\alpha(n))$, where $\alpha(n)$ is the inverse Ackermann function.

eaf77e

```
d41d8c// #include <something>
d41d8c// #include "something_else.h"
d41d8c
d41d8cstruct Dsu {
d41d8c  vi p, rank;
d41d8c  Dsu(int n) : p(n), rank(n, 0) {
d41d8c    iota(all(p), 0);
d41d8c  }
d41d8c  int find(int x) {
d41d8c    return p[x] == x ? x : p[x] = find(p[x]);
d41d8c  }
d41d8c  bool unite(int u, int v) {
d41d8c    if ((u = find(u)) == (v = find(v))) return false;
d41d8c    if (rank[u] < rank[v]) swap(u, v);
d41d8c    rank[u] += rank[p[v] = u] == rank[v];
d41d8c    return true;
d41d8c  }
d41d8c};
```