WeakTrichotomy

Contents

1 Mathematics

1.1 Divisibility

Theorem 1 (The division algorithm)**.** *Given any integers a and b, with $a > 0$, there exist unique integers q and r such that $b = qa + r$, $0 \le r < a$. If $a \nmid b$, then $0 < r < a$.*

Theorem 2. *If $g = (b, c)$, then there exist integers x, y such that $g = bx + cy$.*

Theorem 3. *For any positive integer m, $(ma, mb) = m(a, b)$.*

Theorem 4. *If $d \mid a$ and $d \mid b$ and $d > 0$, then $(a/d, b/d) = 1/d \cdot (a, b)$. If $(a, b) = g$, then $(a/g, b/g) = 1$.*

Theorem 5. *If $(a, m) = (b, m) = 1$, then $(ab, m) = 1$.*

Theorem 6. *For any integer x, $(a, b) = (b, a) = (a, -b) = (a, b + ax)$.*

Theorem 7. *If $c \mid ab$ and $(b, c) = 1$, then $c \mid a$.*

Theorem 8. *If $m > 0$, $[ma, mb] = m[a, b]$. Also $[a, b] \cdot (a, b) = |ab|$.*

1.2 Congruences

Definition 9. Given integers a, b, m with $m > 0$. We say that a is congruent to b modulo m and we write $a \equiv b \pmod m$ if m divides the difference $a - b$.

Theorem 10. *Congruence is an equivalence relation.*

Theorem 11. *If $a \equiv b \pmod m$ and $\alpha \equiv \beta \pmod m$, then*

- *$ax + \alpha y \equiv bx + \beta y \pmod m$ for all integers x and y.*
- *$a\alpha \equiv b\beta \pmod m$.*
- *$a^n \equiv b^n \pmod m$ for every positive integer n.*
- *$f(a) \equiv f(b) \pmod m$ for every polynomial f with integer coefficients.*

Theorem 12. *If $c > 0$ then $a \equiv b \pmod m$ if and only if $ac \equiv bc \pmod{mc}$.*

Theorem 13 (Cancellation law)**.** *If $ac \equiv bc \pmod m$ and if $d = (m, c)$, then $a \equiv b \pmod{m/d}$.*

Theorem 14. *Assume $a \equiv b \pmod m$. If $d \mid m$ and $d \mid a$ then $d \mid b$.*

Theorem 15. *If $a \equiv b \pmod m$ then $(a, m) = (b, m)$.*

Theorem 16. *If $a \equiv b \pmod m$ and if $0 \le |b - a| < m$, then $a = b$.*

Theorem 17. *We have $a \equiv b \pmod{m}$ if and only if a and b give the same remainder when divided by m.*

Theorem 18. *If $a \equiv b \pmod{m}$ and $a \equiv b \pmod{n}$ where $(m, n) = 1$, then $a \equiv b \pmod{mn}$.*

Theorem 19. *Assume $(a, m) = 1$. Then the linear congruence $ax \equiv b \pmod{m}$ has exactly one solution.*

Theorem 20. *Assume $(a, m) = d$. Then the linear congruence $ax \equiv b \pmod{m}$ has solutions if and only if $d \mid b$.*

Theorem 21. *Assume $(a, m) = d$ and suppose that $d \mid b$. Then the linear congruence $ax \equiv b \pmod{m}$ has exactly d solutions modulo m. These are given by $t, t + m/d, t + 2m/d, \ldots, t + (d-1)m/d$, where t is the solution, unique modulo m/d of the linear congruence $ax/d \equiv b/d \pmod{m/d}$.*

Theorem 22 (Euler-Fermat theorem)**.** *Assume $(a, m) = 1$. Then we have $a^{\varphi(m)} \equiv 1 \pmod{m}$.*

Theorem 23. *If a prime p does not divide a then $a^{p-1} \equiv 1 \pmod{p}$.*

Theorem 24 (Little Fermat theorem)**.** *For any integer a and any prime p we have $a^p \equiv a \pmod{p}$.*

Theorem 25. *If $(a, m) = 1$ the solution (unique $\pmod{m}$) of the linear congruence $ax \equiv b \pmod{m}$ is given by $x \equiv ba^{\varphi(m)-1} \pmod{m}$.*

Theorem 26 (Lagrange)**.** *Give a prime p, let $f(x) = c_0 + c_1x + \cdots + c_nx^n$ be a polynomial of degree n with integers coefficients such that $c_n \not\equiv 0 \pmod{p}$. Then the polynomial congruence $f(x) \equiv 0 \pmod{p}$ has at most n solutions.*

Theorem 27 (Wilson's theorem)**.** *For any prime p we have $(p-1)! \equiv -1 \pmod{p}$.*

Theorem 28 (Chinese remainder theorem)**.** *Assume $m_1, \ldots, m_r$ are positive integers, relatively prime in pairs. Let $b_1, \ldots, b_r$ be arbitrary integers. Then the system of congruences $x \equiv b_1 \pmod{m_1}, \ldots, x \equiv b_r \pmod{m_r}$ has exactly one solution modulo the product $m_1 \cdots m_r$.*

2 Generators

2.1 Python

```
from random import randint
from os import system
import sys

def compile_cpp(file):
    """Compile specified file and create executable"""
    system(f"echo␣Compiling␣{file}.cpp")
    system(f"g++␣{file}.cpp␣-o␣{file}")

def generate_input():
    """Create input for a single test case"""
    sys.stdout = open("in", "w")
    print(randint(1, 100))
    sys.stdout = sys.__stdout__

def generate_output(file):
    """Generate output from specified executable"""
    system(f'./{file}␣<␣in␣>␣{file}.out')
    output = open(f'{file}.out', "r").read().strip()
    return output

def generate_outputs(files):
    outputs = []
    for file in files:
        output = generate_output(file)
        outputs.append(output)
    return outputs

def clear(files):
    """Remove specified files in current folder"""
    statement = "rm"
    for file in files:
        statement = statement + "␣" + file
    system(statement)

def are_equal(outputs):
    return len(set(outputs)) == 1

def wrong_answer(files):
    print("Wrong␣Answer")
    clear(files)
    exit(0)

def main():
    # GIVEN the number of test cases and the file names
    n_test_cases = 100
```

```
    files = ["wa", "ac"]
    executables = [f'{file}.out' for file in files]
    for file in files:
        compile_cpp(file)

    # WHEN the outputs generated by both executables on
    # each test case are compared
    for test_case in range(1, n_test_cases + 1):
        print(f'Test␣Case␣{test_case}:␣', end="")
        generate_input()
        outputs = generate_outputs(files)
        if not are_equal(outputs):
            wrong_answer(files)
        print("Correct")

    # THEN they should coincide and
    # the final verdict must be Accepted
    print("Accepted")
    clear([*files, *executables, "in"])

if __name__ == "__main__":
    main()
```

3 Utils

3.1 Bitwise operations

```
Long GetBit(Long mask, Long bit) { return (mask >> bit) & 1LL; }
void TurnOn(Long& mask, Long bit) { mask = mask | (1LL << bit); }
void TurnOff(Long& mask, Long bit) { mask = mask & (~(1 << bit)); }
```

3.2 Coordinate compression

```
void Compress(vector<int>& a) {
  int n = a.size();
  vector<pair<int, int>> pairs(n);
  for (int i = 0; i < n; i++) pairs[i] = {a[i], i};
  sort(pairs.begin(), pairs.end());
  int nxt = 0;
  for (int i = 0; i < n; i++) {
    if (i > 0 && pairs[i - 1].first != pairs[i].first) nxt++;
    a[pairs[i].second] = nxt;
  }
}
```

3.3 Mersenne Twister

```
mt19937_64 rng(chrono::steady_clock::now().time_since_epoch().count());
Long GetRandom(Long l, Long r) {
  return uniform_int_distribution<Long>(l, r)(rng);
}
vector<Long> GetPermutation(int n) {
  vector<Long> permutation(n);
  for (int i = 0; i < n; i++) permutation[i] = i;
  shuffle(permutation.begin(), permutation.end(), rng);
  return permutation;
}
```

3.4 Vim configuration

```
syntax on

:set tabstop=2 softtabstop=2
:set shiftwidth=2
:set expandtab
:set smartindent
:set nu
:set nowrap
:set incsearch
:set relativenumber

:set colorcolumn=80
highlight ColorColumn ctermbg=0 guibg=lightgrey
```

4 Templates

4.1 Pragmas

```
#pragma GCC optimize ("Ofast,unroll-loops")
#pragma GCC target ("sse,sse2,sse3,ssse3,sse4,popcnt,abm,mmx,avx,tune=
    native")
```

4.2 Template

```
#include <bits/stdc++.h>
#define debug(x) cout << #x << " = " << x << endl
using namespace std;
```

```
typedef long long Long;
typedef long double Double;
typedef unsigned long long ULong;
typedef pair<Long, Long> Pair;
const int N = 1e6;
const Long INF = 1e18;
const Double EPS = 1e-9;
int main(void) {
  ios::sync_with_stdio(false);
  cin.tie(0);
  return 0;
}
```