Koji tip podataka koristiti?

- Ako nije potrebno aritmetičko računanje (ili je to potrebno samo na kraju obrade) a barata se s brojevima, razmotri da li bi bilo lakše uneti ih i obradjivati ih kao string.

- Ako u bilo kom trenutku racunanja postoji šansa da se iskoci iz tipa int koristi long long.

- Ako je potrebno baratati s decimalnim brojevima ili je rezultat decimalan, pazi na celobrojno deljenje. Ono ti može uništiti zadatak.

- Nizove smeštaj na pozicije od 1 do N a ne od 0 do N-1 (sem ako to ne donosi prednost u zadatku)

- Imena promenljivima davati tako da te asociraju na ono šta predstavljaju.

- Ako nema potrebe čuvati podatke (nisu kasnije potrebni), moguće ih je obraditi odmah nakon učitavanja i tako uštedeti na memoriji i brzini.

- Char i string uvek učitavaj i ispisuj sa **cin** i **cout**.

- Ako je velika količina podataka na ulazu ili izlazu za aritmetičke tipove koristi scanf() i printf().

- Ako je string sastavljen iz više reči a bitni su razmaci, mora se koristi getline().

- Kada koristiš direktni niz (adresa člana predstavlja njegovu vrednost) ako je samo bitno ima li ga ili nema, koristi bool, ako je bitna pozicija ili broj pojavljivanja, onda int.-

- Kada koristiš bilo kakav niz deklariši desetak članova više nego što ti realno treba. (Ogrizović)

- ugradjenu funkciju pow() je ZABRANJENO koristiti za celobrojne tipove.

Kako izabrati algoritam?

- Ako nemaš druge ideje, uradi tako da radi tačno bar za neki set primera.

- Ograničenja u zadatku mogu ti sugerisati kako rešiti zadatak.

N== do 24 ( 2^N kombinatorno pretraživanje, rekurzija)

N== do 100 ( N^4 četvorostruki ciklus)

N== do 600 ( N^3 trostruki ciklus)

N== do 5 000 ( N^2 dvostruki ciklus, matrica)

N== do 100 000 (N log N sortiranje, binarno traženje)

N== do 1 000 000 (N nešto linearno - par prolaza kroz niz, prekalkulacija, prefix suma)

Kako testirati program?

1.) Test primer koji je u zadatku, skoro nikad ne otkriva ideju kako rešiti zadatak. Taodje ne otkriva granične slučajeve a često pokriva veoma mali deo mogučih rešenja.

2.) Za razgranate zadatke tipa “mala matrica” za svaki scenario smisli test primer i probaj kako radi.

3.) Ne prelazi na sledeću celinu zadatka dok prethodnu nisi proverio.

4.) Ako je zadatak sa više testova, prazni sve promenljive koje su u prethodnom testu promenjene.

5.) Uvek testiraj granične slučajeve. N==1,2,3 ili N == max moguće.

Ovo poslednje mora s datotekama da se odradi (a generišeš primer za koji znaš rešenje tako možeš da otkriješ TLE, MLE, overflow … pa čak i WA)

Kako predati kod programa.

1.) Ime datoteke koju predajes je “imezadatka.cpp” (onako kako se u uputstvu traži)

2.) Posklanjaj sve pomoćne ispise.

3.) Posklanjaj upotrebu fajlova sa freopen ako si testirao sa datotekama.

4.) Uključi sve potrebne biblioteke (čak i ako kod tebe radi nešto bez uklj. biblioteka, tamo ne mora da radi)

#include <bits/stdc++.h>

Ovo ukljucuje sve biblioteke u C++