7 28. dubna – Co očekávat u zkoušky

Do tohoto týdne ještě přesahují grafové algoritmy z minulého týdne, proto přibude jen velmi stručný výklad.

Studenti se často ptají, jak bude vypadat zkouška z Programování II. Známo je, že bude mít písemnou a ústní část. Ačkoliv lze letos očekávat možnost vzdáleného zkoušení, jako dominantní variantu si ukážeme dosud lépe probádanou osobně prováděnou zkoušku. Ta elektronická proběhne podle možností podobně, jen za využití vhodných komunikačních prostředků (především Zoomu). Každá správná zkouška začíná zadáním. To u naší zkoušky bývá poměrně dlouhé a těžko přehledné. Jedním z úkolů je analýza problému (tedy odlišení, co je pro řešení problému důležité a co ne). Po zveřejnění zadání nastupuje písemná část zkoušky, kdy máte připravit řešení. V tom se má nacházet:

0. Upřesnění zadání
1. Postřehy
2. Zdůvodněná volba algoritmu
3. Representace dat
4. Dekomposice – moduly, funkce, objektový návrh
5. Diskuse

Vyhodnoceny budou všechny aspekty (podle možností), ovšem až za vaší přítomnosti na tzv. ústní části. Jelikož na řešení budete mit nejspíše 120 minut, řešení bude značně nepřehledné (samostatně neopravovatelné) a u ústní části bude vaším prvním úkolem vysvětlit nám je. Počítejte s tím, že v průměrném případě jsme Vaše řešení ještě vůbec neviděli (neptejte se zbytečně (a už vůbec ne opakovaně), jestli jsme si ho před ústní částí důkladně přečetli – nechcete-li nás zbytečně rozzlobit). Co nám k řešení řeknete, to se pokusíme vzít na vědomí. Co nám neřeknete, jako by nebylo. U ústní části budete jen *komentovat, co jste opravdu napsali,* nesmíte dodávat nic, co vás napadlo po konci písemné části zkoušky.

K čemu odkazují některé výše uvedené body: Upřesnění zadání se vás začne týkat ve chvíli, kdy dostatečně dlouho po začátku zjistíte, že jste něčemu neporozuměli. V takovém případě se pokusíte pochopit, co jsme asi měli na mysli (například jestli cena za jízdenku může být záporná, jestli rychlost větru může být záporná,...). Obvykle si odpovíte přesně tak, jak jsme úlohu zamýšleli.

Postřehy se týkají rozdělení položek na důležité a nedůležité. Zdůvodněná volba algoritmu bude sestávat z popisu vhodného algoritmu, přiměřeného popisu alternativ a vysvětlení, proč jste se v důležitých okamžicích rozhodli tak, jak uvidíme. Data musíte vhodným způsobem representovat (tak, aby se vešla do paměti, data, co se do paměti nevejdou, musíme vhodně uložit na disk,...). Je důležité, aby representace dat umožňovala efektivní řešení problému (ne aby napřed spadl hardware, co bude problém řešit, pak se delší dobu nebude nic

dít, pak slunce spálí zeměkouli, zanikne vesmír, ještě dlouho nic a pak Váš algoritmus slavnostně dopočítá).

Dekomposice je asi jasná (provedete objektový návrh, popíšete, do jakých souborů byste funkce implementující algoritmy v bodu o zdůvodněné volbě algoritmu rozdělili, jaké třídy a objekty – bude-li to mít smysl,...). Diskusí své snažení zakončíte. Okomentujete, co se kde povedlo, co se nepovedlo, co by se dalo řešit jinak a lépe za jakých okolností (tedy kdybyste věděli, že mince mají hodnotu nejvýše 50, mohli byste je reprezentovat typem `byte` namísto 32bitového integeru či dokonce doublu. A nyní již vzhůru k jednomu konkrétnímu příkladu

7.1 Zadání

Na Matematicko-fyzikální fakultě pravidelně probíhá studentská anketa. Její výsledky se tradičně více či méně pravidelně objevují po koncích zkouškových období souvisejících semestrů. Na vstupu tedy máte dva textové soubory. Jeden popisuje studentská hodnocení, druhý přiděluje pracovníky ke katedrám. Tedy:
Vstup:

1. soubor:
    - Každá řádka obsahuje jedno hodnocení (jedním studentem jednoho předmětu):
    - učitel – 40 znaků,
    - předmět – 40 znaků,
    - hodnocení – číslo $\{1, 2, 3, 4, 5\}$.
2. soubor:
    - učitel – 40 znaků,
    - katedra – 40 znaků.

Vstupní soubory nejsou nijak seřazené! Vypíšete dva soubory, jeden pro studenty, druhý pro vedoucí kateder. Obsahovat budou totéž, budou se lišit jen setříděním.
Výstup:

1. soubor:
    - učitel – předmět – průměr – střední kvadratická odchylka (takto bude vypadat každý řádek tohoto souboru),
    - setříděný podle dvojice učitel-předmět,
2. soubor:
    - obsahuje totéž jako první soubor,
    - setříděný podle trojice katedra-učitel-předmět.

Další omezení:

- učitelů ≤ 1000,
- předmětů ≤ 1000,
- hlasů ≤ 1000000,
- kateder ≤ 50,
- paměť 1 kB.

7.2 Řešení

0. *Upřesnění zadání:* Předpokládáme, že jeden učitel je jen na jedné katedře (ač to není realistické). Výsledky vypíšeme s přesností na 2 desetinná místa, jména učitelů, předmětů a kateder budou znaky UNICODE a jména (učitelů i předmětů) budou různá (což opět není plně realistické). Kdyby údaje nebyly jednoznačné, přidali bychom zjednoznačňující údaje (číslo vyučujícího či kód předmětu).

1. *Postřehy:* Abychom úlohu úspěšně vyřešili, stačí pro každého učitele a předmět zjistit počet hodnotitelů, součet hodnocení a součet čtverců hodnocení (z těchto údajů už průměr a střední kvadratickou odchylku spočítáme. Vypadá to, jako by šlo o úlohu na třídění. Na vstupu i výstupu jsou přibližně táž data (jen zpracovaná), bude tedy vhodné nakreslit Data Flow Diagram.

2. *Algoritmus:* Průměrný student začne úlohu řešit od prostředka a my uděláme totéž (z důvodů, které začnou být jasné za chvíli). Spočítáme-li součet hodnocení za předmět, počet hodnotitelů a součet čtverců hodnocení, průměr spočítáje jako součet hodnocení vydělený počtem, tedy

$$P = \frac{\sum_{i=1}^{n} H_i}{n},$$

střední kvadratickou odchylku budeme chvíli popisovat:

$$\sigma = \frac{1}{n}\sum_{i=1}^{n}(H_i - P)^2 = \frac{1}{n}\sum_{i=1}^{n}(H_i^2 - 2H_iP + P^2) =$$

$$\frac{1}{n}\sum_{i=1}^{n} H_i^2 - \frac{2P}{n} \cdot \sum_{i=1}^{n} H_i + \frac{n \cdot P^2}{n} = \frac{1}{n}\sum_{i=1}^{n} H_i^2 - P^2.$$

Hned vidíme, k čemu je dobrá matematika. Z potřehů známe n (počet hodnotitelů), P (průměr), spočítáme z něj tedy P^2 a také známe (z postřehů) $\sum_{i=1}^{n} H_i^2$.

Nyní tedy zbývá spočítat údaje zmíněné v postřezích a celé to dát dohromady.

Nyní se tedy hluboce zamyslíme, jak spočítat požadované veličiny. U zkoušky se stává, že má člověk plno blbých nápadů a ty správné nápady se mu zdají jako blbosti. Jelikož bude vyhodnoceno jen to, co bude (někde) napsané, i když vám připadá, že je něco zjevná pitomost, někam to raději napište. Může se vám to hodit.

My začneme několika zbloudilými nápady: Mohli bychom třeba vzít první řádek, podívat se, kterého předmětu a vyučujícího se týká. Data bychom zcela prolezli a spočítali údaje pro tento předmět (jak spočítat tři výše zmíněné údaje je snad jasné). Pak zjistíme, že do paměti se nám tyto údaje nevejdou a vyhazovat je ze vstupních souborů také příliš nemůžeme, takže bychom se ve vstupu nejspíše ztratili. Další možnost je zkusit si údaje zapsat do speciálních souborů parametrizovaných jménem vyučujícího a názvem kurzu. To je sice také pěkné, ale jen do chvíle, kdy se nám v v těch jménech začnou objevovat znaky, jimiž nelze pojmenovat soubor. Tyyto nápady moc daleko nevedou. Proto zkusíme nasadit něco úplně jiného.

(a) Vstupní soubor setřídíme podle dvojice učitel-předmět. Rozmyslete si, jaký třídicí algoritmus byste použili. Přednášející doporučuje lehce zoptimalizovaný mergesort. Tím nám vzniknou data v až směšně snadno zpracovatelném formátu.

(b) Setříděná data projdeme (jednou). Máme totiž u sebe ty údaje, co spolu souvisejí. Dokud koukáme na tutéž dvojici učitel-předmět, načítáme: Jedničku do počtu hlasujících, hodnocení do součtu hodnocení a druhou mocninu hodnocení do součtu druhých mocnin hodnocení. Až narazíme na novou dvojici, vypíšeme (opět do pomocného souboru) údaje o současném předmětu, ve formátu učitel-předmět-počet hodnocení-součet hodnocení-součet čtverců hodnocení a údaje si zinicializujeme pro novou dvojici učitel-předmět.

(c) Spočítáme požadované hodnoty (výše naznačeným způsobem).

(d) Hurá, máme první výstupní soubor.

(e) Setřídíme první výstupní soubor podle kateder (položky v rámci katedry necháme v tom pořadí, v jakém byly).

Jelikož jsme normální lidi, třídíme porovnáním v čase $\Theta(n \log n)$. Nic moc jiného (než třídění) v algoritmu není (kromě lineární režie). Složitost je tak v $O(n \log n)$ (a kdybychom chtěli machrovat, dokážeme $\Theta(n \log n)$).

3. *Representace dat:* Jelikož se data nevejdou do paměti, je třeba je udržovat v souborech. První (vstupní) soubor setřídíme (za využití třeba dvou pomocných souborů pro mergesort) do pomocného souboru. Ten přepočítáme na první výstupní soubor (všechny soubory budou textové) a (opět za pomoci dvou mergeovacích souborů) na druhý výstupní soubor. Jednotlivé prvky můžeme reprezentovat třeba jako jednotlivé objekty (řádek

vstupního souboru, řádek pomocného souboru,... – a aspoň nebude nouze o pěkné objekty přirozeně vstupující do řešení).

4. *Dekomposice:* Jednotlivé řádky souboru budou reprezentovány objekty (s vhodnými metodami jako načti a vypiš). Každý bod algoritmu vyřešíme jednou funkcí. Všimneme si, že problémem především protékají data, tak nakreslíme Data Flow Diagram (jehož vrcholy budou jednotlivé funkce a šipky budou říkat, odkud ty funkce budou brát data a kam se budou posílat). Taktéž pro třídění navrhneme vhodnou funkci.

5. *Diskuse:* Výpočet omezuje nedostatek paměti. Kdyby se nám data vešla do paměti a nám se podařilo jimi v konstantním čase indexovat (například vhodným hashováním), mohli bychom problém řešit třeba i lineárně (sublineární řešení nepřipadá v úvahu, jelikož data musíme alespoň přečíst).

7.3 Ústní část

Projdete-li úspěšně písemnou částí a jejím vysvětlením, následovat bude skutečná ústní část, kdy se budeme vyptávat na všechno, co bylo tento rok na hodinách Programování 1 a 2.