Korespondenční Seminář z Programování

ZAČÁTEČNICKÁ KATEGORIE

29. ročník | KSP-Z | Červen 2017

Poslední letošní série skončila a nyní nás všechny čekají zasloužené prázdniny. Po prázdninách ale na vás (nebo na vaše kamarády či sourozence) bude čekat KSP s novou várkou úloh. A pokud se již po tomto ročníku ročníku KSP-Z cítíte zkušenější, můžete zkusit příští rok nahlédnout třeba i do KSP-H.

A jako obvykle se nás nebojte zeptat, pokud vám cokoliv není úplně jasné. Obrátit se na nás můžete přes fórum na našich stránkách nebo e-mailem na *ksp@mff.cuni.cz*.

Řešení čtvrté série začátečnické kategorie 29. ročníku KSP

29-Z4-1 Šíření viru

Úloha se šířícím se počítačovým virem byla hezká simulace na úvod letošní poslední série. V zadání jsme slíbili, že v každém vstupu se všechny počítače nakazí, takže stačilo jen počítat, za jak dlouho to nastane. Jak na to?

U každého počítače si budeme pamatovat, v jakém kroku byl nakažen (maximum z těchto čísel pak bude finální odpovědí) a kolik má nakažených sousedů (na počátku u všech počítačů nula). Zatím nezpracované nakažené počítače si budeme držet ve frontě, do které na počátku vložíme nakažené počítače ze vstupu (a nastavíme jim krok, ve kterém byly nakaženy, na nulu).

Pak nám stačí postupně vybírat počítače z fronty a zpracovávat je. Každému zatím nenakaženému sousedovi zpracovávaného počítače zvedneme počítadlo nakažených sousedů o jedna a pokud tím soused přesáhne kritickou mez (jeho počítadlo nakažených sousedů dosáhne alespoň poloviny počtu jeho sousedů), nakazíme ho taky – vložíme ho na konec fronty nakažených počítačů a nastavíme mu krok nakažení o jedna vyšší, než má aktuálně zpracovávaný počítač.

Tímto postupem postupně nakazíme všechny počítače (zastavíme se ve chvíli, kdy už ve frontě nebude žádný nezpracovaný nakažený počítač) a díky tomu, že na ukládání nezpracovaných nakažených počítačů používáme frontu, spočítáme správně i krok nakažení – počítače zpracováváme jakoby „ve vlnách", nejprve zpracujeme všechny počítače, které se nakazí v kroku k (což může vést k nakažení nějakých počítačů v kroku $k+1$), než postoupíme dál. Pro nalezení odpovědi si tak stačí průběžně pamatovat maximální krok, ve kterém nakazíme nějaký z počítačů.

Přidáme ještě drobný implementační tip: Lepší, než porovnávat, že $A \geq B/2$ je porovnávat $2A \geq B$, vyhnete se tak hloupým zaokrouhlovacím chybám.

Program (Python 3):
`http://ksp.mff.cuni.cz/viz/29-Z4-1.py`

Jirka Setnička

29-Z4-2 Vybírání atrakcí

Tato úloha šla efektivně řešit *hladovým algoritmem*. Hlavní trik spočívá v tom, že před samotným přiřazováním atrakcí si vše setřídíme. Atrakce setřídíme jednoduše podle jejich rychlosti. Osoby setřídíme vzestupně podle nejvyšší snesitelné rychlosti a v případě rovnosti je dále setřídíme podle snesitelné rychlosti nejnižší.

Setřízené atrakce nyní přesuneme do seznamu nepoužitých atrakcí. Dále pro každou osobu nalezneme pozici nejlehčí nepoužité atrakce, kterou již zvládne. Od této pozice vybereme K za sebou jdoucích atrakcí a ty osobě přidělíme. Přidělené atrakce poté odstraníme ze seznamu nepoužitých atrakcí. Takto postupujeme, dokud nevyužijeme všechny atrakce.

Proč tohle může fungovat? Vezměme prvního člověka v popsaném pořadí, pojmenujme ho A. Nemáme příliš volnosti, nějakých K atrakcí mu přidělit prostě musíme. Každopádně má smysl uvažovat pouze ty, které snese. Mezi nimi náš algoritmus zvolí ty, jejichž rychlost je nejmenší. Klíčové v tuto chvíli je, že horní mez osob už pouze poroste – mezi lidmi, pro které tyto atrakci nejsou příliš pomalé, má osoba A nejnižší horní mez. Pokud bychom tyto atrakce přidělili komukoli jinému, mohlo by se stát, že všechny další atrakce budou na A příliš těžké. Naopak nikoli, tedy můžeme atrakce přidělit právě osobě A.

V algoritmu bychom dvakrát tvrdě narazili, pokud bychom neměli zaručeno, že řešení existuje. Poprvé tehdy, kdy přidělujeme K po sobě jdoucích atrakcí osobě, o které víme, že zvládne tu nejpomalejší z nich. Díky způsobu přidělování atrakcí ale víme, že pokud by osoba na nějakou atrakci nestačila, znamená to, že v seznamu nepoužitých už není více atrakcí, na které by osoba stačila – řešení nemůže existovat. Podruhé pak když předpokládáme, že s dokončením seznamu atrakcí jsme přiřadili K atrakcí každému člověku.

Nyní se zaměřme na časovou složitost. Počet atrakcí si označíme $M = NK$. Jelikož třídíme osoby i atrakce, nedostaneme se na složitost lepší, než $\mathcal{O}(M \log M)$.

Jak to je ale s vybíráním atrakcí? Většina operací závisí na správném výběru datové struktury pro ukládání zatím nepoužitých atrakcí.

Mohli bychom použít pole a v něm vždy vyhledat první použitelnou atrakci binárně. Bohužel, jedno odebrání (i více najednou) prvků v poli trvá $\mathcal{O}(M)$ – po odebrání musíme zacpat vzniklou díru přesunutím všech prvků napravo. Složitost bychom si tím pohoršili na $\mathcal{O}(NM + M \log M)$.

Každopádně, na plný počet bodů tento postup už stačil. Komu to ale nestačí, a věří že to musí jít lépe, nechť čte dál.

Na reprezentaci seznamu místo pole použijeme vyhledávací strom.[1] Po setřídění atrakcí z nich sestavíme vyvážený strom, to nám stačí i v $\mathcal{O}(M \log M)$. Vyhledání

[1] `http://ksp.mff.cuni.cz/viz/kucharky/stromy`

prvku i jeho odstranění je rovněž rychlé, obě tyto operace vyhledávací strom zvládne v $\mathcal{O}(\log M)$. Prvních K prvků od daného místa vybereme jednoduše pomocí hledání následníka, které trvá rovněž $\mathcal{O}(\log M)$.

Podotkněme, že není potřeba žádný speciální samovyvažovací strom. Stačí strom na začátku programu postavit rozumně vyvážený a později z něj jen odebírat. Tím se strom sice znevyváží, pořád ale bude mít hloubku nejvýše $\mathcal{O}(\log M)$.

Celkem v algoritmu provedeme M vložení a odstranění, to nám celkově zabere $\mathcal{O}(M \log M)$ času. N atrakcí jsme vyhledali, zbylých $M - N$ jsme našli jako následníky. Po sečtení všech operací nám tudíž vyjde, že časovou složitost udržíme na $\mathcal{O}(M \log M)$.

Program (Python 3):
`http://ksp.mff.cuni.cz/viz/29-Z4-2.py`

Václav Končický

29-Z4-3 Želva v akváriu

Želva nás provázela všechny čtyři série a doufáme, že pro vás byla stejně dobrým společníkem jako pro Kevina. Tentokrát jste si za ni mohli vysloužit dokonce o dva body více, protože orientace v 3D prostoru může být náročná.

Jak jsme psali v zadání, tentokrát nestačí pamatovat si jen směr, kterým se želva dívá. Je potřeba si pamatovat ještě alespoň jeden. Například, kterým směrem má natočený krunýř, nebo kam míří její pravý bok.

Z těchto tří směrů si stačí pamatovat jen libovolné dva – třetí se dá vždy dopočítat. Úplně nejsnáze si ale řešení úlohy pořídíte, pokud si budete pamatovat všechny tři směry. Přepočítávat totiž stačí vždy pouze dva – směr v ose, okolo které se želva otáčí, se nikdy nemění. Jinými slovy, při každém otočení se změní pouze dva směry.

Všechny směry si budeme pamatovat jako *vektory* jednotkové délky. Takový vektor má tři složky, každou v jedné ze tří souřadných os. Každá složka obsahuje buď 0, 1 nebo -1, navíc je právě jedna složka nenulová. Například vektor ukazující na sever (směr pohledu želvy na začátku programu) si zapamatujeme jako $[1, 0, 0]$ – stejně jako v zadání.

Představme si nejprve jednoduché otočení želvy doprava. Směr nahoru, tedy kam míří krunýř, se touto rotací nezmění. Směr, kam se želva dívá, stačí nahradit za ten, kam byl předtím natočen pravý bok. Zbývá už jen spočítat nový směr pravého boku. Ten je rovnoběžný s tím, kam se předtím želva dívala – jen je opačným směrem.

Pro celou operaci otočení doprava tedy stačí prohodit směr pohledu a pravého boku, a následně pravý bok obrátit – každou složku vektoru vynásobit -1. Ostatní rotace jsou už jen variací na stejné téma. Vždy stačí jen některé dva směry prohodit a jeden z nich obrátit.

Program (Python 3):
`http://ksp.mff.cuni.cz/viz/29-Z4-3.py`

Ondra Hlavatý

29-Z4-4 Hledání součtu

Zadání úlohy po nás chtělo najít v posloupnosti N čísel takovou její souvislou podposloupnost, že součet prvků této podposloupnosti bude co nejblíže zadanému číslu C. Pro vyřešení několika prvních vstupů postačil jednoduchý přístup k řešení – stačilo vyzkoušet všechny možné podposloupnosti a z nich vybrat tu s nejmenším rozdílem jejího součtu a čísla C. Úloha se ale dá snadno vyřešit v lineárním čase, jak si za chvíli ukážeme.

Zkusme si nejprve úlohu trochu zjednodušit. Nebudeme hledat podposloupnost s nejbližším součtem k číslu C, ale podposloupnost, jejíž součet je roven právě C. Každá souvislá podposloupnost začíná a končí na nějakých indexech původní posloupnosti. Označme si tato místa jako zac a kon, přičemž vždy platí $zac <= kon$. Na začátku položíme $zac = kon = 0$. Tyto indexy budeme při běhu programu postupně zvyšovat. Zavedeme si dále proměnnou S, ve které budeme mít uložen součet prvků zadané posloupnosti mezi místy zac a kon.

Provedeme nyní několik pozorování. Pokud je $S > C$, nemá cenu zvyšovat kon. Raději zvýšíme zac. Zadání předpokládá pouze kladná čísla, pokud bychom kon o 1 zvýšili, zvýšilo by se nám i S o nově přidaný prvek. My ale chceme, aby $S = C$. Pokud naopak zvýšíme o 1 index zac, S se nám sníží, což je v této situaci žádoucí. Obdobně, pokud je $S < C$, nemá cenu zvyšovat zac. Raději zvýšíme kon o 1. A nakonec, pokud je $S = C$, máme nalezeno řešení a můžeme skončit.

Tato pozorování nám dávají přímý návod na sestavení algoritmu pro zjednodušenou verzi úlohy. Jak toto řešení rozšířit tak, aby zvládlo i úlohu ze zadání? Velmi jednoduše! Stačí si při běhu ukládat dosud nejlepší nalezené řešení a v každé iteraci pak kontrolovat, jestli aktuální řešení není náhodou lepší, než to doposud nejlepší nalezené. O tom, které řešení je lepší, lze rozhodnout velmi jednoduše – z absolutní hodnoty rozdílu S a C. Máme-li dvě řešení A, B a k nim příslušné S_A respektive S_B, porovnáme $|S_A - C|$ s $|S_B - C|$ a vybereme menší z nich. Výpočet můžeme ukončit ve chvíli, kdy máme $S = C$ nebo kdybychom zac zvýšili na N. (pozn. indexujeme od 0, poslední prvek posloupnosti má index $(N - 1)$.)

Povšimněme si, že způsob manipulace s indexy v podstatě odpovídá přidávání prvků zadané posloupnosti do fronty a jejich odebírání z ní. Každý prvek se přitom může do fronty dostat maximálně jednou. Odtud nám plyne i časová složitost řešení $\mathcal{O}(N)$.

Program (C++):
`http://ksp.mff.cuni.cz/viz/29-Z4-4.cpp`

Jan 'Toman' Tománek

29-Z4-5 Hanojské věže jinak

Dostali jsme nějaké rozestavení, jak máme vlastně začít? Dobrým prvním krokem by bylo rozmyslet si na jaké tyči chceme věž postavit. Všimněte si, že je vždy nejlepší věž postavit na tyči, kde už je největší disk. Proč tomu tak je? Kdybychom chtěli věž postavit jinde, musíme uvolnit největší disk (nesmí na něm být žádné další disky) a celou tyč, kde chceme věž postavit, na poslední tyč tedy musíme vyskládat všechny zbylé disky. Pak přesuneme největší disk a pokračujeme ve stavbě. Nicméně krok přesunu největšího disku si můžeme odpustit a a rovnou stavět na největším disku, nijak si nepohoršíme.

Nyní jsme si vybrali tyč, kde budeme stavět věž a navíc tam máme už první, největší disk. Dřív než než na něj začneme skládat menší disky musíme na něj nějak dostat druhý největší. Představme si, že největší disk neexistuje. Snažíme se tedy vystavět věž s výškou o jeden disk menší než původně, tentokrát ale na konkrétní tyč – na tu, kde se nachází disk největší. Poznamenejme, že tím, že je největší disk největší,

	řešitel	škola	ročník	sérií	Z4-1	Z4-2	Z4-3	Z4-4	Z4-5	Z4-6	série	celkem
114.	Tomáš Dostál	MendelG_OP	2	1							0,0	23,0
115.–116.	Alexandra Géciová	GJHroncaBA	1	3							0,0	22,0
	Marek Zelený	GVoděraPH	3	1							0,0	22,0
117.–119.	Ondřej Buček	GJarošeBO	3	1							0,0	20,0
	Matouš Bílek	GJŠkodyPŘ	2	1							0,0	20,0
	Ondřej Gajda	GTěš	0	3							0,0	20,0
120.–127.	Dávid Daubner	GVaršŽilina	2	1							0,0	18,0
	Tereza Hladíková	JazG HK	3	1							0,0	18,0
	Štěpán Košan	GKlatovy	4	2							0,0	18,0
	Václav Luňák	GDašickáPA	4	1							0,0	18,0
	Ondřej Mašek	GEBenešeKL	3	1							0,0	18,0
	Tomáš Novotný	G BO-Řeč	3	2							0,0	18,0
	Pavel Turinský	G Brandýs	4	5							0,0	18,0
	Matěj Šmíd	SPŠÚžlabPH	3	4							0,0	18,0
128.–130.	Martin Hubata	GMikulášPL	1	3							0,0	16,0
	Arian Adam Ott	GSOŠRok	0	2							0,0	16,0
	David Rajchman	MasG_Plzeň	0	2							0,0	16,0
131.	Petr Doubravský	AkademG_PH	1	2							0,0	15,0
132.	Dominik Tulak	SPŠMasarLI	1	1							0,0	14,0
133.–134.	Ondřej Hráček	GOlgHavl	0	1							0,0	12,0
	Vojtěch Lengál	GZborovPH	3	1							0,0	12,0
135.	Jindřich Dítě	VOSPŠŽďár	1	5							0,0	11,0
136.–137.	Mária Ďuračková	GJHroncaBA	2	1							0,0	10,0
	Václav Šraier	GČeskoliPH	4	3							0,0	10,0
138.–144.	Matúš Ferech	GJHroncaBA	4	1							0,0	8,0
	Frantisek Hanzlik	ZŠ Elem	−1	1							0,0	8,0
	Jan Hřebenář	20. ZŠ	0	1							0,0	8,0
	Filip Kastl	GKepleraPH	1	1	8						8,0	8,0
	Jan Kutálek	GSOŠNovJicin	3	1							0,0	8,0
	Šimon Prokop	21. ZŠ	0	1							0,0	8,0
	Natálie Volfová	GJírovcČB	1	1							0,0	8,0
145.–148.	Petr Chotěborský	GSla	0	1							0,0	6,0
	Jakub Hroník	GJiříPoděb	3	1							0,0	6,0
	Daniel Rozehnal	GJWolkraPV	1	1							0,0	6,0
	Jakub Švojgr	GČeskáČB	−2	1							0,0	6,0
149.	Petr Kabourek	G BO-Řeč	1	1							0,0	4,0
150.	Tomáš Husák	GLitoměřPH	3	1							0,0	3,0
151.–152.	Filip Bouda	SŠStrŠvejc	1	1							0,0	2,0
	Matej Nižník	GPošKošice	2	1	2						2,0	2,0
153.	Štěpán Henrych	GŽat	1	1							0,0	0,3

KSP pro vás připravují studenti Matematicko-fyzikální fakulty Univerzity Karlovy.

Webové stránky: *https://ksp.mff.cuni.cz/* **E-mail:** *ksp@mff.cuni.cz* **Diskusní fórum:** *https://ksp.mff.cuni.cz/forum/*

Chcete-li s námi komunikovat bezpečně, můžete si ověřit náš HTTPS certifikát – jeho SHA1 fingerprint je: E9:DB:EE:C6:62:BC:14:DE:09:E4:E8:97:DC:36:0E:87:B3:50:B0:01.

se nám nic nezmění tím, že ho budeme ignorovat – můžeme na něj dát libovolný disk, stejně jako na prázdnou tyč.

Pomohli jsme si tedy vůbec? Stále máme za úkol postavit věž, byť o kousek menší, navíc na konkrétní tyč. Ukážeme si, že ano. Označme si druhý největší disk (popř. největší, který neignorujeme) jako D. Jako první musíme dostat D na cílovou tyč. Můžeme mít štěstí a D tam už bude, pak můžeme tento krok přeskočit, pokud máme ale smůlu, chce to trochu úsilí, pojďme se na to podívat.

Abychom mohli přesunout D na cílovou tyč, musí být cílová tyč prázdná (až na ignorovaný disk) a D volná (nesmí na něm být žádný další disk). Všechny ostatní disky tedy musí být vyskládané do věže na poslední tyči. Poté můžeme přesunout disk D a poté celou vyskládanou věž na cílovou tyč.

To vypadá, jako bychom se zase nikam nedostali. Abychom vyřešili problém přeskládání věže, musíme vyřešit problém přeskládání dvou věží (nejprve na uvolnění cílové tyče a D, poté na přeskládání zbylých disků na cílovou tyč). Nicméně klíčové je, že tyto věže jsou opět o jeden disk nižší. Můžeme je vyřešit právě tím samým algoritmem! Abychom uvolnili cílovou věž a D, můžeme na chvíli ignorovat D a jako D dočasně označit další největší disk. Pak necháme proběhnout celý algoritmus, poté přesuneme naše původní D a pak znova necháme náš algoritmus pracovat.

Jenže skončí toto někdy? Vždyť v každém průběhu algoritmu vytváříme dva další. Musíme si uvědomit, že až budeme pracovat s jedním diskem, nepotřebujeme vůbec spouštět nějaký složitý algoritmus, stačí přesunout disk na správné místo, pokud tam ještě není. Ti zkušenější z vás v tomto postupu zcela správně vidí analogii s tzv. *rekurzí*.

Nakonec ještě připomeneme, proč toto vyskládání je nejrychlejší možné. Na začátku jsme opatrně vybrali tu nejlepší tyč, na které věž stavět. Poté už všechny kroky byly nutné (uvolnit disk a cílovou tyč, přeskládat zbylé disky), takže jsme tento postup nemohli vůbec urychlit.

Dominik Smrž

29-Z4-6 Tajná síť taháků

Nejprve si zkusíme problém trochu formalizovat. Místo tajné sítě taháků si můžeme rovnou představit graf. Jeho vrcholy budou Kevinovi spolužáci, hrany povedou mezi těmi spolužáky, kteří si mezi sebou předávají taháky napřímo.

Nebude to ale jen tak ledajaký graf – v zadání jsme vám slíbili, že nebude obsahovat cykly. Navíc, protože jde o jednu síť, můžeme předpokládat, že jde o graf souvislý. Graf, který je souvislý a neobsahuje cykly, je strom.

Bez újmy na obecnosti si strom libovolně zakořeníme. Nyní se podívejme, co se stane, pokud ze stromu odtrhneme libovolný vrchol v. Strom se rozpadne na několik podstromů: jeden za každého syna v, a jeden navíc za otce v. Tyto části mají dohromady $|V| - 1$ vrcholů. Jak také jinak, že :)

Na pomoc si přizveme obyčejné prohledávání do hloubky. Budeme postupovat stejně, jako bychom chtěli spočítat počet vrcholů ve stromě: z rekurze do listů vrátíme 1, jinak součet hodnot vrácených ze synů zvýšený o jedna.

Tento algoritmus jednoduše modifikujeme. Během návratu z rekurze víme, jaká je velikost všech podstromů. Pokud odečteme jejich součet od celkového počtu vrcholů (a odečteme ještě jedničku navíc), získáme velikost zbylé části grafu. Není-li žádná z nich větší než polovina, vyhráli jsme, hledaným vrcholem je ten aktuální.

Takto najdeme toho správného spolužáka v lineárním čase. Najdeme ho ale vždy?

Tak, jak byla úloha zadána, se může stát, že žádný takový neexistuje. Vezměte si třeba strom tvořený dvěma vrcholy spojenými hranou. Ať odebereme libovolný z nich, zbylý podstrom bude mít vždy alespoň polovinu původního počtu vrcholů.

Protože ale náš algoritmus projde nakonec všechny vrcholy, tak si můžeme být jistí, že pokud ten správný vrchol existuje, tak jej najdeme.

Kdybychom ale požadovali ostře více než polovinu původního počtu vrcholů, aby síť zůstala v provozu, situace by byla zajímavější. O tom zas ale někdy příště…

Ondra Hlavatý

Výsledková listina čtvrté série začátečnické kategorie 29. ročníku KSP

	řešitel	*škola*	*ročník*	*sérií*	*Z4-1*	*Z4-2*	*Z4-3*	*Z4-4*	*Z4-5*	*Z4-6*	*série*	*celkem*
0.					8	10	10	12	12	14	66,0	264,0
1.	Jakub Šťastný	G BO-Řeč	2	8	8	10	10	12	12	8	60,0	258,0
2.	Lucia Krajčoviechová	GJHroncaBA	1	4	8	10	10	12			40,0	238,0
3.	Karel Balej	G_Rokycany	2	4	8	10	10	12	10	14	64,0	236,0
4.	Daniel Skýpala	GTomkovaOL	−1	8	4,7	2,7	3,4	12	2	8	32,7	222,7
5.	Petr Borňás	G_Roudnice	1	4	8	3,3	10	12	4	2	39,3	219,3
6.	Petr Aubrecht	GHeyrovPH	2	4	8		10	12			30,0	218,0
7.	Petr Šimůnek	G_Hořice	4	4	6		10	4			20,0	205,0
8.	Michaela Bobeničová	GPošKošice	2	4	8	4		12			24,0	189,0
9.	Vladimír Chudý	ZŠRonov	0	4	8		0				8,0	180,0
10.	Ondřej Jamelský	G Cheb	−1	4	8	0	10	12	7	13	50,0	179,4
11.	Terézia Strišovská	GJHroncaBA	1	4	8	3,3	10	12			33,3	179,3
12.	Ondřej Gonzor	G Brandýs	0	7				2			2,0	165,0
13.	Petr Budai	G JGJ PH	0	4	8						8,0	162,0
14.	Jiří Löffelmann	GLitoměřPH	3	8	8	10	10	12			40,0	160,0
15.	Michal Kodad	SPŠ_Smíchov	1	3							0,0	149,0
16.	Jan Kotovský	GPísnickáPH	−2	4	8	1		2			11,0	145,0
17.	Andrej Pajtaš	GLitoměřPH	1	3							0,0	140,5
18.	Jakub Šuráň	GStrážnice	2	4	8						8,0	140,0
19.	Erik Kučák	GHorMichal	4	3							0,0	139,0
20.	Vojtěch Březina	GCoubTábor	0	8	8			2			10,0	134,0
21.	Dávid Šutor	GTerVans	2	2							0,0	127,5
22.	Erik Berta	GAlejKošice	2	4	1						1,0	123,7
23.	Robert Jaworski	GÚstavníPH	−1	7	8	0	10	12			30,0	123,0
24.–25.	Jan Kaifer	GČesBrod	1	10	8	0					8,0	120,0
	Jan Vodstrčil	G VMýto	0	3							0,0	120,0
26.–27.	Martin Bencko	GOhradníPH	0	8		0		12	1	2	15,0	119,0
	Kateřina Čížková	G_Rokycany	3	3							0,0	119,0
28.	Anna Hollmannová	GSRandyJN	0	7							0,0	116,0
29.	Tomáš Domes	MendelG_OP	4	5							0,0	108,0
30.	Jakub Jirkal	GJungmanLT	2	12	8	1					9,0	107,0
31.	Jakub Ucháč	ŠMaVVzt	1	3							0,0	103,0
32.	Vincent Orlovský	GTerVans	2	2							0,0	99,0
33.	Dalibor Kramář	G BO-Řeč	2	3							0,0	98,0
34.	Zuzana Urbanová	GFXŠaldyLI	3	3							0,0	97,0
35.	Jozef Mikuláš	CZŠJBosca	0	3							0,0	95,0
36.	Vojtěch Hudec	G_ČTřebová	3	5							0,0	94,0
37.	Jan Kučera	GFKřižíka	0	4	2						2,0	93,0
38.	Jaroslav Knápek	GLesníZlín	0	3							0,0	92,3
39.	Ondřej Wrzecionko	GTěš	2	4	2						2,0	92,0
40.	Lucie Vomelová	GŠpitálsPH	1	3							0,0	88,0
41.	Radoslav Hašek	G_Čáslav	3	8			10	2			12,0	84,0
42.	Jakub Brož	PČGKarVary	3	2							0,0	83,0
43.	Andrej Tomči	GHorMichal	4	2							0,0	82,0
44.	David Nápravník	GLitoměřPH	4	5							0,0	81,7
45.	Lukáš Caha	GZborovPH	3	2							0,0	81,0
46.–47.	Vojtěch Brož	GBudějovPH	2	2							0,0	78,0
	Gabriela Pachlová	G_ČTřebová	2	2							0,0	78,0
48.	Petr Macháček	G_TýnNVlt	1	2							0,0	77,0
49.	Vojtěch Káně	G Brandýs	1	8	0						0,0	76,0
50.	Kateřina Nová	G_Vimperk	4	4	2,7	0					2,7	73,7
51.	Karel Tomanec	SPŠBruntál	4	3			0				0,0	72,0
52.	Dominik Pilný	G_Ostrov	2	3							0,0	69,0
53.	Martin Sobotka	GLitoměřPH	1	2							0,0	68,0
54.	Dávid Oravec	G DubNVáh	2	3							0,0	66,0
55.–56.	Jaroslav Paidar	SPŠMasarLI	3	2							0,0	65,0
	Rajmund Hruška	GPošKošice	4	3							0,0	65,0
57.	Martin Zmitko	G FrýdlNOs	1	4	8	0		9			17,0	64,0
58.	Josef Polášek	GKeplerаPH	1	3							0,0	63,7
59.–60.	František Kmječ	G Brandýs	1	2							0,0	62,0
	Barbora Plačková	GHlu	−1	2							0,0	62,0
61.	Ondřej Krsička	GJarošeBO	1	5							0,0	61,0
62.	Michal Mlčoch	G UherBrod	2	5		0					0,0	60,0
63.–65.	Daniela Hrbáčová	G Wicht	3	3							0,0	58,0
	Pavel Martinec	GLesníZlín	3	4							0,0	58,0
	Filip Masár	PiarGNitra	3	1							0,0	58,0
66.	Ondřej Cach	SPSE_Pard	1	5							0,0	54,0
67.–68.	Tomáš Sládek	GJHroncaBA	2	3							0,0	49,0
	Ladislav Töpfer	G DrJPekMB	2	2							0,0	49,0
69.	Janek Hlavatý	GJirsíkaČB	−2	12							0,0	48,0
70.–71.	Jan Chybík	SPŠMasarLI	2	2							0,0	46,0
	Vojtěch Kuchař	ZŠ Sobotka	0	7				0			0,0	46,0
72.–73.	Martin Hofbauer	G BO-Řeč	2	3							0,0	44,0
	Tomáš Vítek	G_Břeclav	0	3	8						8,0	44,0
74.	Jan Koška	GJírovcČB	−3	3				12			12,0	43,0
75.–77.	Timea Szöllősová	G_Gröss_BA	1	1							0,0	41,0
	Matouš Vondrášek	GJírovcČB	1	1							0,0	41,0
	Jan Štěch	GJirsíkaČB	0	4		0					0,0	41,0
78.–82.	Martin Horáček	GŠumperk	4	1							0,0	40,0
	Martin Melicher	GPošKošice	2	1							0,0	40,0
	Adam Perinay	GJHroncaBA	3	1							0,0	40,0
	Dennis Pražák	GJirsíkaČB	2	5							0,0	40,0
	Jakub Szymsza	CmGy PV	1	1							0,0	40,0
83.	Vít Beran	MasG_Plzeň	3	2							0,0	38,0
84.	Radim Buráň	G UherBrod	2	3							0,0	37,0
85.–87.	Jan Juračka	GBystnPern	2	1							0,0	36,0
	Michael Kozel	GZborovPH	3	7							0,0	36,0
	Magdaléna Rýdlová	GLesníZlín	3	2							0,0	36,0
88.	Václav Čermák	GKlatovy	4	1							0,0	34,0
89.	Tomáš Chabada	SPŠMasarLI	1	2							0,0	32,0
90.–95.	Jan Bíl	GDašickáPA	4	1							0,0	30,0
	Dominik Dinh	GNVPláníPH	2	3	2						2,0	30,0
	Tomáš Dulava	GMatOS	3	1							0,0	30,0
	Vít Gaďurek	Neuvedená	2	7							0,0	30,0
	Vojtěch Krupka	GJungmanLT	3	1							0,0	30,0
	Erik Řehulka	ŠPMNDaGB	1	2							0,0	30,0
96.–98.	Tuan Anh Hoang	GZborovPH	3	4							0,0	29,0
	Vladimír Holý	CírkG Plzeň	2	2							0,0	29,0
	Jiří Kvapil	GTomkovaOL	−1	4	2	0					2,0	29,0
99.–108.	Michal Grňo	BiGyBBHK	4	1							0,0	28,0
	Jaroslav Horáček	GymVyš	3	1							0,0	28,0
	Lucie Kubíčková	GFXŠaldyLI	3	1							0,0	28,0
	Tomáš Nguyen	SPŠÚžlabPH	2	2							0,0	28,0
	Adéla Návratová	ZŠ MTyrš	−1	1							0,0	28,0
	Ondrej Potúček	G_GolNitra	3	3							0,0	28,0
	Vojtěch Poupa	CírkG Plzeň	−1	1							0,0	28,0
	Adrián Rošinec	GHorMichal	4	1							0,0	28,0
	Eliška Vlčinská	GHladnov	2	2							0,0	28,0
	Benedikt Žour	G UherBrod	2	8							0,0	28,0
109.	Jiří Janoušek	GBudějovPH	1	2							0,0	27,0
110.	Jan Martínek	GTomkovaOL	2	1							0,0	26,0
111.	Evgeniya Knyazeva	GNVPláníPH	3	3							0,0	25,0
112.–113.	Roman Ondráček	GBoskovice	3	7							0,0	24,0
	Lukáš Vavřík	GNeumannŽR	3	2							0,0	24,0