Gymnázium, Praha 6, Arabská 14

Obor programování

April 2023

Rubikova kostka

Adam Rubeš, Petr Soukop, Tomáš Vondra

Prohlašujeme, že jsme jedinými autory tohoto projektu, všechny citace jsou řádně označené a všechna použitá literatura a další zdroje jsou v práci uvedené. Tímto dle zákona 121/2000 Sb. (tzv. Autorský zákon) ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium, Praha 6, Arabská14 oprávnění k výkonu práva na rozmnožování díla (§ 13) a práva na sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

Anotace

Cílem projektu bylo vytvořit desktopovou aplikaci, ve které si uživatelé budou moci složit 3D model Rubikovy kostky. Aplikace by také zároveň uměla kostku zamíchat a složit jakékoli její zadání. Pro uživatele, kteří neumějí složit Rubikovu kostku, aplikace nabízí kompletní tutoriál, včetně procházení skládacích algoritmů krok po kroku.

Obsah

1 Úvod

Aplikace Rubikovy kostky byla napsána v jazyce Java a její grafické rozhraní v JavaFX. Program nabízí hned několik funkcí. V hlavním panelu s kostkou lze kostku skládat klávesovými zkratkami nebo táhnutím myši. Program obsahuje veškeré vysvětlení aplikace a algoritmů v sekci *Nápověda* a *Tutoriál*. Zároveň si uživatel také může přednastavit, jak bude kostka, kterou bude skládat, vypadat a následně ji skládat v hlavním panelu s 3D modelem kostky.

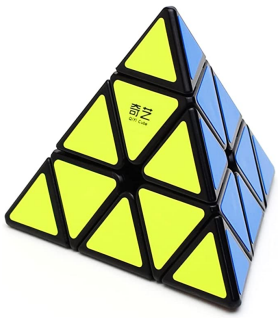
2 O Rubikově kostce

Rubikova kostka je hlavolam, ve své původní podobě tvořený krychlí složenou z dílčích barevných krychliček, jehož úkolem je pohyby stran přeuspořádat jednotlivé dílčí části tak, aby každá strana kostky byla obarvena jen jednou barvou. V atypických variantách se jedná o hranoly, jehlany, mnohostěny a další tělesa.

Nejběžnějším typem kostky je model 3×3×3. Tento má tři vrstvy, které dohromady tvoří 26 krychliček - 8 rohů, 12 hran, 6 středů a jádro. Každá z krychliček je nositelem jedné, dvou nebo tří jednobarevných nálepek, přičemž dohromady je na celém tělese zastoupeno tolik barev, kolik stěn má krychle, tedy šest. Počet barev na krychličce je dán jejím umístěním na tělese — na rohové krychličce se nachází tři nálepky, na hranové dvě a na středové jedna. Celá soustava je propojena pohyblivým mechanismem, který umožňuje kteroukoli vrstvu pootočit o libovolný celočíselný násobek pravého úhlu. Středy jsou jako jediné z částí nepohyblivé, tzn. zaujímají vůči sobě stále stejnou polohu. Barva středu tím pádem určuje, jaká má být výsledná barva celé stěny tento střed obsahující. Celkový počet kombinací pro kostku 3×3×3 je 43 252 003 274 489 856 000. V červenci 2010 bylo dokázáno, že jakoukoliv kombinaci lze vyřešit do 20 tahů.

Rubikovu kostku lze řešit mnoha způsoby. Existuje mnoho systematických metod, díky kterým kostku vyřeší i laik. Postupy se liší především průměrným počtem tahů potřebných na složení, principem a množstvím různých algoritmů, které je potřeba si zapamatovat. Nejznámější metoda spočívá ve skládání jedné vrstvy po druhé. Další známé a používané metody jsou např. tzv. CFOP (Jessica Fridrich's method), Lars Petrus method, Zborovski-Bruchem method či corners first; jejich společný prvek je to, že byly vyvinuty pro rychlé skládání kostky, tzv. speedcubing.

Na světě existuje několik nejrůznějších variant Rubikovy kostky. Princip verze 3×3×3 byl aplikován na větší i menší rozměry, načež se na trh dostaly typy 2×2×2, 4×4×4, 5×5×5 a podobně. Dále se vynalezla jiná tělesa, než jsou krychlového tvaru, což demonstrují např. hlavolamy Megaminx, Pyraminx nebo Square-1.



Obrázek 1: Pyraminx

2.1 Další Rubikovy hlavolamy

Ernö Rubik vynalezl více než jen skládací kostku, ale i skládacího hada. Rubikův had je hlavolam tvořený 24 rovnoramennými trojúhelníkovými hranoly, které jsou střídavě orientovány a seřazeny do řady. Hranoly lze libovolně otáčet, ale otáčením je nelze od sebe oddělit. Pomocí otáčení může člověk dosáhnout různých tvarů jak geometrických, tak napodobujících různé předměty nebo zvířata (obdélník, pes, kočka apod.) Ernö Rubik ho vynalezl v roce 1981.

2.2 O Ernöovi Rubikovi

Obrázek 2: Ernö Rubik

Kostka je pojmenována po jeho maďarském autorovi Ernövi Rubikovi, který se narodil v Budapešti roku 1944. Rubik vystudoval a odmaturoval na Gymnáziu krásného a užitečného umění v oboru sochařství. Pak se přesunul k architektuře a v roce 1967 promoval na Technické univerzitě v Budapešti. Poté pokračoval v doktorském studiu v oboru interiérové architektury na Akademii aplikovaných umění. Po absolvování tohoto studia byl v roce 1971 na Akademii jmenován docentem a začal vyučovat trojrozměrný design.

2.3 Historie Rubikovy kostky

Rubik si všiml, že jeho studenti, ačkoli doktorandi, mají velkou potíž s trojrozměrnou vizualizací. Proto přemýšlel o rozličných cvičeních, které by jim zadal. Řez krychlí různými rovinami je poměrně standardní cvičení a vede k dobře známým 2x2x2 a 3x3x3 polím. Když v květnu roku 1974 přemýšlel o 2x2x2 a 3x3x3 polích, tento zájem ho dovedl k otázce: "Jak mohu pootočit stěnami takových útvarů?"Do šesti týdnů měl řešení. Měsíc mu trvalo složit jeho první model kostky.

Původní nápad Ernö Rubika bylo zhotovení kostky jako učební pomůcky na vysoké škole při cvičení trojrozměrné představivosti. Prošla si svým vývojem, protože první prototyp se Ernő Rubik nejprve pokusil sestrojit jako kostku složenou z papíru, ale brzy zjistil, že papír se příliš rychle trhá a kostka se nedá správně posouvat. Nakonec se rozhodl vyrobit kostku z dřeva a s pomocí kuličkových ložisek zajistit, aby se dala snadno otáčet. V roce 1974 měl Rubik hotovou první verzi kostky a začal ji zkoumat a testovat. Ernön Rubikem byla kostka patentovaná 30. ledna roku 1975, avšak pod názvem Kouzelná kostka. Teprve později v roce 1980 dostala kostka název Rubikova kostka. První oficiální mistrovství světa ve skládání Rubikovy kostky se konalo v roce 1982 v Budapešti. Vítěz ji tehdy složil za 22,95 vteřiny. První úspěchy na sebe nenechaly dlouho čekat a již během následujících šesti let se Rubikova kostka, tehdy magická kostka, stala masově vyhledávanou hračkou či hlavolamem. Úspěch byl takový, že ač se jednalo o matematický hlavolam určený pro studenty univerzity, získala několikrát ocenění „Hračka roku“.

Rubikova kostka získala několik ocenění a brzy se proslavila. První ocenění dostala již v roce 1978, kdy získala kostka cenu na Mezinárodním strojírenském veletrhu v Budapešti, což vedlo k řadě zahraničních zakázek. V letech 1980 až 1981 byla vyhlášena hračkou roku ve třech evropských zemí (Německo, Francie a Velká Británie). Do své sbírky stavebnictví a designu ji zařadilo Muzeum moderního umění v New Yorku. Zahrála si v několika hollywoodských filmech – namátkou ve snímcích Zmizení nebo také třeba Armageddon. Po celém světě vznikly kluby a sdružení Rubikovy kostky. Na Youtube se dnes nachází více než čtyřicet

tisíc návodů na její složení. Bylo zaznamenáno několik dřívějších pokusů o výrobu Magické kostky, stejně tak jako pozdější nezávislé návrhy. Měřítkem Rubikova úspěchu je to, že žádný jiný design pro 3x3x3 nebyl po dobu uvedení Rubikovy kostky vyráběn.

Oficiální světový rekord pro složení Rubikovy kostky 3x3x3 je 3.47 vteřin. Rekord drží Číňan Yusheng Du. Softwaroví inženýři Ben Katz a Jared Di Carlo si postavili robota, který složí tento hlavolam za 0,38 sekundy, což z něj dělá nejrychlejšího řešitele Rubikovy kostky na světě. Dvě webkamery na snímání konfigurace rozmíchaného hlavolamu, 6 upravených motorů a jejich ovladačů měří čas skládání tomuto robotovi. V případě 2x2x2 je světový rekord roven 0,49 vteřiny, který drží Polák Maciej Czapiewski. Za dobu existence bylo pokořeno několik nejrůznějších rekordů s Rubikovou kostkou, jako je třeba složení Rubikovy kostky pod vodou, ve volném pádu a také třeba při jízdě na kole poslepu.

3 Použité technologie

Pro tvorbu naší aplikace jsme zvolili objektově orientovaný programovací jazyk Java vytvořený americkou společností Sun Microsystems a aktuálně vlastněnou technologickým gigantem Oracle Corporation. Pro tvorbu grafického rozhraní jsme zvolili nejnovější grafickou knihovnu v javě a to JavaFX. Jako build system aplikace používá Apache Maven. K tvorbě programu jsme využívali integrované vývojové prostředí InteliJ od společnosti JetBrains a k vyhotovení dokumentace jsme využily LaTeX editor OverLeaf.

V pořadí zleva nahoře: Obrázek 3: Java, Obrázek 4: JavaFX, Obrázek 5: Maven, Obrázek 6: SceneBuilder, Obrázek 7: InteliJ IDEA, Obrázek 8: LaTeX

4 Grafické rozhraní aplikace

Grafické rozhraní aplikace je vyhotoveno pomocí knihovny JavaFX. Každé okno aplikace je načítáno ze souboru XML, ve kterém jsou uložené informace o jednotlivých grafických prvcích. S každým XML souborem je následně spojena Controller třída, která obsahuje řídící prvky a část logiky programu.

Pro načítání XML souborů jejich Controllerů a měnění obsahu na obrazovce je vytvořena třída ContentManager. Mimo jiné třída ContentManager také zajištuje načítání uživatelem zvoleného jazyka ze souboru .yml, viz v následující kapitole.

K tvorbě jednotlivých souborů XML byla využita aplikace od firmy Gluon SceneBuilder, která výrazně zjednodušuje práci se soubory XML a finálním stylingem. Zároveň bylo využito pro normalizaci stylu skrz aplikaci a dynamické měnění kaskádových stylů, které jsou do jisté míry kompatibilní s JavaFX.

4.1 Hlavní okno

Když otevřeme aplikaci, vyskočí na nás úvodní okno. Zde se nachází několik tlačítek, které odkazují na další podokna. Největší z nich je tlačítko *Spustit*, které spustí 3D model Rubikovy kostky. Pod ním jsou další tři tlačítka, která jsou spíše vedlejší. Tlačítko *Nápověda*, odkáže uživatele na kompletní vysvětlení, které slouží uživatelům pro lepší orientaci v našem programu. Tlačítko *Tutoriál* obsahuje kompletní návod, jak sestavit Rubikovu kostku 3x3x3. Na hlavním panelu se nachází ještě tlačítko *Vlastní rozložení*, jehož funkce se vysvětluje níže v dokumentaci.

4.2 Vlastní rozložení

Po stisknutí tlačítka *Vlastní rozložení* se uživatelům objeví okno, kde si budou moct vytvořit vlastní rozložení Rubikovy kostky na předem připravené síti kostky 3x3x3. Středy stran kostky jsou předem předvyplněny a nelze je přenastavit (stejně jako u normální Rubikovy kostky nelze středy vyměnit). Uživatel má po levé straně na výběr ze šesti barev. Uživatel si smí zvolit jen jednu. Výběr je nastaven jako *radio buttons*, to znamená, že výběr

více barev v jeden moment není možný. Každá barva se na kostce musí vyskytovat právě na devíti políčkách (jako tomu je i na běžné Rubikově kostce). Po obarvení celé kostky stačí kliknout na tlačítko *Uložit* a kostka se převede do 3D modelu, kde je možné kostku skládat nebo ji může složit i program podle svého algoritmu.

4.3 Tutoriál

Po stisknutí tlačítka *Tutorial* se uživatelům objeví okno, kde se dozví základní informace o tom, jak složit Rubikovu kostku a seznámí se i s notací pohybů kostky. Tutorial, který je zde uživateli prezentován, popisuje kombinaci CFOP algoritmu a beginner metody, které jsou vhodné pro začátečníky i mírně pokročilé, protože slouží jako základní odrazový můstek pro pochopení časově méně náročných metod. Tento postup je popsán krok po kroku a doprovázen obrázky, které usnadňují pochopení postupu. Podrobnější informace se o postupu dozvíte v následujících kapitolách.

4.4 Výběr jazyků

V aplikaci je také možné vybrat si jeden ze dvou zatím podporovaných jazyků. Data se načítají ze souboru YAML, který je načítán ve třídě ContentManager za pomocí importované knihovny yamlsnake. Výstupem načítání dat je Mapa s klíčem ve formátu String a Objektem jako hodnotou. Tato mapa je pak předána knihovní třídě FXML loader (pomocí které se načítají XML soubory), aby bylo možné dané hodnoty referovat ze souboru XML.

Do budoucna by mohlo být zajímavé rozšíření vytvořit uživatelsky přístupnou možnost vytvářet vlastní překlady bez nutnosti přímé práce se souborem YAML.

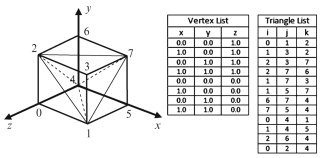
5 3D skládání kostky

Při stisknutí tlačítka *Spustit* se uživateli zobrazí okno obsahující 3D model kostky a množství ovládacích prvků sloužících k interakci s modelem. S kostkou může uživatel otáčet v prostoru jako s celkem nebo také s jejími jednotlivými stranami za použití myši nebo tlačítek na klávesnici. Zároveň je možné využít funkce automatického zamíchání nebo automatického vyřešení, kdy se spustí algoritmem vygenerovaná sekvence tahů.

Zároveň je možné zamíchat si sám nebo náhodně programem kostku a poté si pomocí timeru změřit čas, jak dlouho uživateli trvá složit kostku a kolik tahů vám to zabralo.

5.1 TriangleMesh

3D model Rubikovy kostky se skládá z 27 samostatných kostiček, které jsou tvořeny za pomoci tříd z knihovny JavaFX. Především třídy TriangleMesh a MeshView. TriangleMesh (česky síť trojúhelníků) je 3D struktura hojně využívaná k modelování 3D objektů. Pro vytvoření TriangleMeshe musíte nadefinovat body v prostoru, body na textuře, kterou chcete namapovat na model a z tak zvaných obličejů, které reprezentují samotné trojúhelníky. Obličej je definován pomocí 3 bodů v prostoru a 3 bodů na textuře.



Vertex List		
x	y	z
0.0	0.0	1.0
1.0	0.0	1.0
0.0	1.0	1.0
1.0	1.0	1.0
0.0	0.0	0.0
1.0	0.0	0.0
0.0	1.0	0.0
1.0	1.0	0.0

Triangle List		
i	j	k
0	1	2
1	3	2
2	3	7
2	7	6
1	7	3
1	5	7
6	7	4
7	5	4
0	4	1
1	4	5
2	6	4
0	2	4

Obrázek 9: Kostka v prostoru

Pro vytvoření krychle je zapotřebí zadat 8 bodů v prostoru (jeden pro každý vrchol krychle) a pak podle potřeby maximálně 8 bodů na textuře. Následně budeme uvažovat o každé straně krychle jako o dvou rovnoramenných, pravoúhlých trojúhelnících, které mají společnou přeponu a body, které na ní leží. Takto vyhotovený TriangleMesh je následně předán třídě MeshView, které ho vykreslí do prostoru.

5.2 Animace

Animace kostky jsou vytvořeny za pomoci další knihovní třídy JavyFX AnimationTimer. V naší implementaci extendujeme tuto třídu a vytváříme abstraktní třídu Animacem, jejíž potomci potom reprezentují jednotlivé pohyby kostky. Tyto třídy dědí z AnimationTimer metodu handle, která je volána při každém nově vykresleném snímku a pokaždé pohne kostkou o daný počet stupňů.

Takto robustní systém potomků byl vytvořen v prvních verzích programu, kdy jsem plánoval napsat metody pro animace kompletně sám za použití pouze základních funkcí JavaFX na měnění polohy objektů v prostoru. Později se ukázalo, že nezbytné kalkulace pro pohyb kostek v prostoru, zahrnující trigonometrické funkce a čísla s často nekonečným desetinným rozvojem, ztrácejí na přesnosti se zvyšujícím se množstvím tahů. Nakonec jsem tedy zvolil využití knihovních funkcí pro spolehlivý pohyb objektů v prostoru, a proto se nyní může zdát celková struktura poněkud redundantní. S vizuální reprezentací kostky je taktéž spojené trojrozměrné pole, které udržuje informace o pohybu jednotlivých kostek pomocí operací s maticemi.

5.3 Automatické provádění tahů

Pro vykonání sekvence tahů, která byla například vypsaná algoritmem využíváme třídu CasovacAnimací, která je potomkem třídy Thread. Tato třída je spouštěna jako nové vláko z hlavního grafického vlákna a sama si následně časuje tahy, které mají být vykonané voláním třídy VláknaAnimací, která je taktéž potomkem třídy Thread.

Obě tyto třídy fungují jako samostatná vlákna, což aplikaci přidává stabilitu a celkovou plynulost. V případě nečekaného problému se čtením nebo vykonáváním pohybů, dojde většinou pouze k ukončení sekvence tahů, ale hlavní grafické vlákno zůstane běžet. Tato implementace byla vymyšlena s myšlenkou toho, že by uživatelé mohli vkládat vlastní sekvence tahů, které by pak program vykonal. Na realizaci této funkce se pracuje.

5.4 Timer a ukládání tahů

V aplikaci je také možné využít timer, který uživateli změří čas, za který dokáže složit kostku a kolik pohybů k tomu potřeboval. Tato funkce otevírá dveře pro budoucí vylepšení aplikace o síťové propojení, které by umožnilo, aby hráči mezi sebou sdíleli své výsledky a soutěžili navzájem mezi sebou.

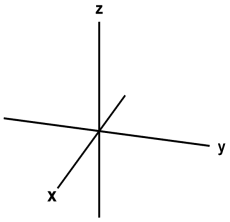
K ukládání počtu tahů potřebných ke složení kostky se váže i mechanismus ukládající každý vykonaný tah nehledě na to, jestli je timer spuštěný. Ukládání těchto tahů je užitečné pro řešící algoritmus, který právě ze sekvence rozložení kostky zvládne vygenerovat postup pro složení. Více však o fungování algoritmu najdete níže v dokumentace.

6 Algoritmus řešící rozložení kostky

6.1 Datové zobrazení kostky

Samotná kostka je v programu ukládána jako trojrozměrné pole objektů "Kosticka", tento objekt představuje jednu malou kostku, kterých má celá kostka 27. "Kosticka"je definována svými souřadnicemi, pravdivostními hodnotami určující, zda se jedná o roh či hrana kostky a polem objektů KostickaBarva.

Kostka používá kartézský systém souřadnic, tedy X značí v kostce šířku, Y značí hloubku a Z značí výšku. Například, kostka se souřadnicemi x=2; y=1; z=0 bude hrana horní a pravé strany, s barvami žlutá a oranžová.



Obrázek 10: Souřadnicová soustava

Co se týče barevného rozložení, kostka používá vcelku standardní rozpoložení barev. Horní strana má žlutou barvu, přední strana má zelenou barvu a pravá strana má oranžovou barvu. Zbytek barev je automaticky spjat s předchozími jmenovanými barvami.

Obrázek 11: Barevné rozložení kostky

Objekt "KostickaBarva"by se dal považovat za stavební kámen řešících postupů. Objekt totiž zobrazuje barvu a směr, v jakém je barva položená na konkrétní kostičce. Ve finální kostičce je tedy potom uloženo více těchto objektů v poli a každý z nich určuje jednu z jejích stran. Pro ukládání barev a směrů byla použita již mezinárodně používaná notace, která používá první písmena z anglických slov. Pro strany tedy z front, back, up, down, right, left (přední, zadní, horní, spodní, pravá, levá) stane F, B, U, D, R, L a pro barvy se zkrátí white, yellow, green, orange, blue, red (bílá, žlutá, zelená, oranžová, modrá, červená) na W, Y, G, O, B, R.

KostickaBarva má poté gettery a settery na získávání potřebných hodnot. Užívány jsou ale spíše speciální metody v třídě Kosticka, které získávají směr či barvu na základě druhé známé

hodnoty tedy barvy či směru. Například getSmerBarvy získá směr zadané barvy v kostičce tím, že projde pole KostickaBarva objektů a porovná je se zadanou barvou. Třída Kosticka má ještě gettery na pravdivostní hodnoty hrana, roh a metody isBilyRoh a orientHrana. Metoda isBilyRoh vrátí "true", pokud je daná kostička rohem a zároveň její pole KostickaBarva objektů má informaci o bílé barvě. Metoda orientHrana slouží k orientaci, na které straně se nachází daná hrana na základě souřadnic X a Y. Třída Kosticka má v neposlední řadě také setter a getter na pole objektů KostickaBarva pojmenované barvy.

6.2 Zpracování pohybů

Zpracování řetězce s pohyby dle základní notaci kostky probíhá v metodě provestPohyby. Ta dostane jakožto parametr dané pohyby, ty projde a určí, zda se nejedná o speciální pohyby, tedy opačné či dvojité pohyby, které se značí doplňujícím znakem za písmenem pohybu. Na základě tohoto rozboru zavolá metodu otoceni, která dostane pouze řetězec s jedním pohybem.

Metoda otoceni poté stanoví podle daného pohybu, jak vypadá pole nyní, jak má vypadat po otočení, a kterým směrem se má pole otočit ve stupních. Následně dočasnému poli předá současný stav kostky. Všechny tyto proměnné předá třídě otoceniPole, jakožto parametry. V případě, že metoda otoceni dostane jako parametr pohyb x, y, z nebo jejich opačné možnosti, zavolá metodu provestPohyby pro konkrétní pohyby, do kterých se toto celkové otočení osy dá rozdělit. Například pro x bychom mohli vytvořit náhradní řetězec R M‘ L‘.

Metoda otoceniPole po získání všech potřebných parametrů provede na základě zadaného směru zkopírování původní kostky do nové a tu následně otočí, neboli změní souřadnice jednotlivých kostiček. Otočení ostatních informací proběhne pouze změnou směrů barev v jednotlivých kostičkách, a to na základě uvedených tvarů v char polích pre a post.

6.3 Zamíchání kostky

Kostka má i vlastní náhodný generátor zamíchání, který je v metodě generatorZamichani. Ta si v char poli možných polí náhodně vezme určitý pohyb a k tomu na základě dalšího náhodně generovaného čísla přiloží k tahu buď mezeru, apostrof nebo číslo dva. Tím se vytvoří normální tah nebo jeden ze speciálních. Šance na jeden z těchto tří znaků je třetinová. Metoda po zavolání vrátí konečný řetězec.

Samotné zamíchání kostky je zprostředkováno pomocí metody zamichat, která dostane jako parametr řetězec pohybů. Tyto pohyby poté zadá metodě provestPohyby jako parametr.

6.4 Řešení

Program používá kombinaci různých metod na skládání Rubikovy kostky, ale nejvíce používána je metoda CFOP. Ta spočívá ve složení nejdříve bílého kříže, poté druhé vrstvy, následně žluté strany a pak zbytku kostky. Tento program ale užívá jiné metody, co se týče složení druhé vrstvy, kdy namísto přímého dokončení bílé strany společně se složením druhé vrstvy, dokončí nejprve bílou stranu a až poté složí druhou vrstvu.

Řešení se do částí programu zařizující vykreslování 3D kostky a ostatní vizuální prvky dostává pomocí metody vypsat v třídě Test. Ta dostane jako parametr řetězec pohybů, kterými byla kostka zamíchána. Metoda postupně vytvoří datovou kostku, kterou poté zamíchá řetězcem v parametru a vygeneruje řešení pro jednotlivé části řešení, ty celou dobu ukládá do řetězce, který metoda vrátí jako svou návratovou hodnotu.

6.4.1 Vytvoření bílého kříže

První částí řešení je metoda bileHrany, která upraví kostku tak, aby všechny bílé hrany byly kolem žlutého středu. Metoda si nejprve projde všechny hrany ve spodní straně, pokud nějaké najde, tak se ujistí, že jejich posunutím žádné správně položené hrany z horní strany neodstraní pomocí metody slot. Metoda slot dostává jako parametry souřadnice X, Y, Z a barvu a na jejich

základě generuje tahy k případné záchraně dané kostičky s barvou z parametru ve vrstvě vyhovující souřadnici Z.

Metoda bileHrany po zavolání metody slot zavolá metodu orientRoh na vygenerování tahů na umístění rohu nalezeného ve spodní vrstvě do vrstvy horní. Poté projde metoda všechny hrany ve spodní vrstvě, které se nenachází ve spodní straně a zjistí, zda nejsou bílé. Pokud ano, zjistí se pomocí metody orientHrana, v jaké se hrana nachází straně, následně se pomocí metody slot vygenerují pohyby k případné záchraně již správně orientovaných hran v horní straně, a pak se podle strany hrany provede předem daný řetězec pohybů. Poté se provedou podobné procesy s druhou a horní vrstvou. Jakmile je vše hotovo, tak se ještě rekurzivně zavolá metoda bileHrany a všechny bílé hrany jsou na svém místě. Složený řetězec je následně metodou vrácen jako její návratová hodnota.

Druhou částí je metoda bilyKriz, která upraví kostku tak, aby vznikl na bílé straně obrazec kříže. Tato metoda poběží v cyklu, dokud počítadlo bílých rohů ve žluté straně pocetBilychHran nebude mít návratovou hodnotu rovnu nule. Metoda projde všechny hrany ve žluté vrstvě, a pokud se jedná o bílou hranu, která má zároveň ve správném směru správnou barvu podle obecného rozložení kostky, tedy hrana je u středu správné strany ze čtyř vedlejších stran – modré, červené, zelené, oranžové – tak provede pohyb pro správné umístění hrany do bílé strany vygenerovaný podle metody orientHrana. Pokud ovšem není splněna podmínka o vedlejším středu, provede se pohyb U, aby se případně našla správná hrana, která by se měla otočit. Metoda následně vrátí pohyby jako svou návratovou hodnotu.

6.4.2 Dokončení bílé strany

Třetí částí je metoda bilaVrstva, která dokončí složení bílé strany kostky. Ta nejprve zavolá metodu bileRohy, která slouží ke generování pohybů na správnou konstelaci bílých rohů do bílé strany s pomocí metody bilyRohSpravne, která zjišťuje, zda je bílý roh na správné pozici pro tyto pohyby. Následně zavolá metodu spatneBileRohy, která zjistí, zda není nějaký roh špatně orientován, pokud ano, tak tento roh vytlačí ven a zavolá metodu

bileRohy pro jeho správné umístění. Poté je zahájen cyklus, který volá metody bileRohy a spatneBileRohy, dokud platí pravdivostní hodnota metody bilyRohU, která kontroluje, zda jsou v horní vrstvě nějaké bílé rohy.

6.4.3 Vkládání hran kostky

Čtvrtou částí je metoda vlozeniKraju, která dokončí druhou vrstvu kostky. Ta nejprve zavolá metodu krajeU, která na podobném principu jako metoda bileRohy si upraví kostku tak, aby měla kostka na přední straně hranu, kterou bude třeba posunout. Následně vrátí řetězec předem daných pohybů podle toho, zda je třeba hranu posunout na levou stranu či na pravou stranu. Poté je zavolána metoda spatneKraje, která zjistí, zda nejsou některé hrany v prostřední vrstvě špatně orientovány, pokud ano, tak se provedou podobné pohyby jako v krajeU, aby byla daná hrana vytlačena. Následně se spustí cyklus, který volá metody krajeU a spatneKraje znovu, dokud jsou v horní vrstvě rohy, které tam nepatří, tedy nejsou žluté.

6.4.4 Složení žluté strany

Pátou částí je metoda zlutyKriz, která upraví kostku tak, aby na žluté straně vznikl útvar kříže. Ta nejprve spočítá počet žlutých hran na horní straně, což v tomto stádiu skládání může být pouze nula nebo dva, pokud není kříž již vytvořen. Pokud je počet roven nule, tak metoda do své návratové hodnoty dostane již předepsaný řetězec pohybů, kterým dosáhne žlutého kříže. Pokud je počet roven dvojce, tak může nastat jeden ze dvou případů. Proto metoda zjistí X souřadnice těchto hran, a pokud je absolutní hodnota jejich rozdílu sudá, pak je zde vytvořená čára ze tří žlutých kostiček a metoda se ujistí, že tato čára je vůči přední straně kostky rovnoběžná. Pokud ne, bude provádět pohyb U, dokud tomu tak nebude stále dokola. Poté metoda vrátí předem dané pohyby, jakožto řešení daného problému.

Pokud je ovšem absolutní hodnota rozdílu X souřadnic lichá, pak metoda zkontroluje, zda je na žluté straně útvar ve tvaru písmena L složeného ze tří žlutých kostiček. Pokud ne, bude provádět pohyb U, dokud tomu tak nebude stále dokola. Následně

metoda vrátí jiné předem dané pohyby pro řešení tohoto problému.

Šestou částí je metoda zlutaVrstva, která dokončí složení žluté vrstvy. Ta do dočasné proměnné uloží počet žlutých rohů, ten zjistí zavoláním metody zluteRohy, která projde horní stranu kostky zjistí, kolik jejích rohů je žlutých. Následný cyklus bude pokračovat, dokud nebudou rohy čtyři. V něm se nachází podmínky, které rozřadí dané řetězce pohybů na základě počtu žlutých rohů, předtím ale musí kostka vyhovovat určité konstelaci různých žlutých kostiček, ta je docílena opakováním U pohybu. Když jsou provedeny a uloženy dané pohyby, tak se zjistí nový počet žlutých rohů a na základě toho buď cyklus končí nebo se opakuje.

6.4.5 Dokončení kostky a optimalizace

Poslední částí je metoda zbytek, která dokončí kostku. Tato metoda už v zásadě spočívá jen v přidávání již předem daných pohybů na základě stavu kostky. Tudíž se jedná pouze o sbor podmínek a cyklů s průchody dat kostky.

Na konci je výstup ještě upraven metodou optimalizovat, která projde zadaný řetězec a smaže případné zbytečné tahy nebo zkrátí výstup z například "U U"na "U2". A pak je už řetězec nastaven jako návratová hodnota a řešení je hotové.

7 Vlastní tutoriál

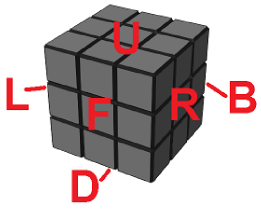
V programu je implementován vlastní tutoriál, který má za úkol uvést začátečníky do skládání Rubikovy kostky. Jeho rozdělení do kroků však neodpovídá krokovému dělení v řešícím algoritmu. První a druhá část algoritmu je shrnuta prvním krokem tutoriálu. Zbytek je už ovšem odpovídající, třetí část algoritmu odpovídá druhému kroku, čtvrtá část třetímu kroku a tak dále.

Zde je originální znění tutoriálu uvedené v programu:

„Vítejte v tutoriálu na skládání Rubikovy kostky. Pokud jste nováček nejprve doporučujeme obeznámit se s obecnou notací pohybů kostky a až poté překročit k samotnému skládání.

NOTACE POHYBŮ:

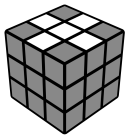
Pro notaci pohybů se používají první písmena jednotlivých stran v angličtině, tedy F jako front – přední strana a podobně. Obecné pohyby jsou brány otočení po směru hodinových ručiček, v protisměru se k písmenu strany přidá apostrof. Poté je tedy F otočení přední strany po směru hodinových ručiček a F‘ je otočení přední strany v protisměru. Dále se používají písmena x, y, z pro otočení kostky celé v její ose. X je osa strany R a L, Y stran U a D a Z je osa stran F a B.



Obrázek 12: Notace kostky

POSTUP:

Tento postup je pouze jeden z mnoha používaných, většina z nich je však v základech velmi podobná. Prvními kroky se skládá první ze stran, která je nejčastěji vybírána bílá, ale můžete si vybrat, jakou chcete, tutoriál je napsán pro začátek s bílou stranou jakožto strana D.

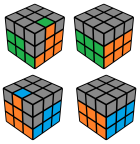


Postup se dělí do 6 kroků. Prvním je složení bílého kříže, tedy musíte dojít do stádia, kdy na bílé straně budete mít obrazec na obrázku.

Toho docílíte tím, že bílé hrany dostanete kolem žlutého středu a poté podle postranních různě barevných středů (modrá, zelená, oranžová, červená) budete bílé hrany dosazovat k bílému středu pomocí otáčení žluté strany. Pokud tedy chcete bílou hranu s například červenou hranou dostat na správné místo, budete točit se žlutou stranou, dokud nebudete mít pod bílou hranou dvě červená pole a poté hranu přesunete na bílou stranu dvojitým otočením červené strany.

Druhým krokem je dokončení bílé strany, tedy vložení bílých rohů. To se provádí za pomocí pohybů R U R' U', které slouží k otáčení příslušného rohu, pokud je špatně orientovaný, nebo k dosazení rohu na místo. Na konci tohoto kroku by kostka měla vypadat jako na obrázku.

Třetím krokem je složení druhé vrstvy. Ta se dokončí vložením správných krajů mezi správné strany. Pro tento krok existují dva případy a dva konkrétní postupy na to, abyste dostali konkrétní kraj z horní strany na své místo.



U' L' U L U F U' F'

U R U' R' U' F' U F

Pokud se dostanete do situace, kdy máte kraj správně umístěný, ale špatně orientovaný, použijte jeden z postupů k vložení jiného rohu, čímž vysvobodíte požadovanou kostičku.

Čtvrtým krokem je složení žlutého kříže. V tuto chvíli byste již měli mít složené obě patra a na žluté straně byste měli mít jeden z následujících obrazců, přičemž ke každému z nich je jednoduchý postup, jak se dostat ke žlutému kříži.

F R U R' U' F' U2 F U R U' R' F'

F U R U' R' F'

U R U' R' U' F' U F

Nyní by vaše kostka měla vypadat jako na obrázku. Pátým krokem je dokončení žluté strany. Pokud v tuto chvíli už nemáte složenou žlutou stranu, tak byste měli být v jednom z následujících stavů.

1) Máte 0 žlutých rohů na svém místě, takže si kostku upravte tak, aby první pole směrem k vám na levé straně bylo žluté, šedivá strana na obrázku by měla být vaše přední.

2) Máte 1 žlutý roh na svém místě, takže si kostku upravte tak, aby pole v levém dolním rohu na horní straně bylo žluté.

3) Máte 2 žluté rohy na svém místě, takže si kostku upravte tak, aby pole směrem vlevo na přední stěně bylo žluté.

1) 

2) 

3) 

Pokud máte kostku správně upravenou, tak proveďte následující postup: R U R' U R U2 R'. Pokud nemáte složenou žlutou stranu, opakujte celý pátý krok.

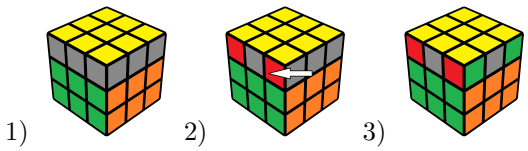
Posledním krokem je dokončení celé kostky. Nyní když máte složenou žlutou stranu, tak záleží už pouze na třetí vrstvě vedlejších stran (modrá, červená, zelená, oranžová). Tu dokončíte pomocí dvou kroků. V prvním musí nastat jeden ze tří

následujících stavů:

1) Žádná z vedlejších stran nemá stejně barevné rohy ve třetí vrstvě.

2) Jedna z vedlejších stran má stejně barevné rohy ve třetí vrstvě. Kostku si tedy upravte tak, abyste tuto stranu měli směrem od vás, například pohybem U.

3) Více z vedlejších stran mají stejně barevné rohy.

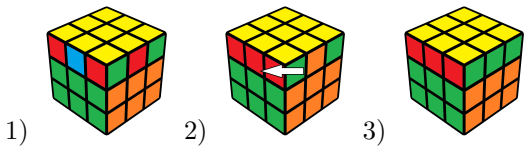


Pokud jste se ocitli v případu 1 nebo 2, proveďte následující pohyby: R' F R' B2 R F' R' B2 R2. A zkontrolujte, že nastala možnost 3, pokud ne, opakujte pohyby. Nyní by měl nastat jeden ze tří stavů:

1) U žádné ze stran není střed shodné barvy s rohy.

2) U jedné ze stran je střed shodné barvy s rohy. Kostku si upravte tak, aby tato strana byla směrem od vás například pohybem U.

3) U více stran je střed shodné barvy s rohy.



Pokud je vaše kostka ve stavu 1 nebo prostřední pole směrem k vám a první pole na levé straně shodují v barvě a kostka je ve stavu 2 (na obrázku).

Tak proveďte následující pohyby: R2 U R U R' U' R' U' R' U R'. Pokud jste se dostali do některého z předchozích tří stavů, postupujte podle nich. Pokud jste ve stavu 2 a nenastala předchozí popisovaná situace, tak proveďte následující: R U' R U R U R U' R' U' R2. Nyní byste se měli ocitnout ve třetím stavu, opakujte tedy pohyb U, dokud není vaše kostka složená."

8 Závěr

V závěru bychom chtěli říct, že aplikace splnila vše, co bylo v zadání. Na rozdíl od ostatních aplikací s Rubikovou kostkou na internetu má naše o několik funkcí více, jako je například vlastní rozložení kostky a tutoriál. Do budoucích let by náš tým chtěl do aplikace přidat více variant Rubikových kostek, které byly již v této dokumentaci zmíněny, třeba například 2x2x2 nebo jiné tvary odlišné od kostky. Zároveň by náš tým chtěl přidat možnost přidání vlastního motivu na kostku.

9 Bibliografie

https://hlavolam.maweb.eu/historie-rubikovy-kostky
https://www.novinky.cz/clanek/
koktejl-rubikova-kostka-bavi-svet-uz-45-let-40312053
https://cdr.cz/clanek/
historie-rubikovy-kostky-jak-vznikla-kdo-ji-vynalezl-jak-se-stala-popularni-p
https://cs.wikipedia.org/wiki/Rubikova_kostka
https://en.wikipedia.org/wiki/Rubiks_Cube#Solutions
https://en.wikipedia.org/wiki/CFOP_method
https://ruwix.com/the-rubiks-cube/
how-to-solve-the-rubiks-cube-beginners-method/
https://cubesolve.com/
https://ryanstutorials.net/rubiks-cube-tutorial/
https:
//ruwix.com/the-rubiks-cube/advanced-cfop-fridrich/
https://what-when-how.com/wp-content/uploads/2012/07/
tmpc2f9555.png
https://docs.oracle.com/javase/8/javafx/api/javafx/
scene/shape/TriangleMesh.html
https://www.youtube.com/watch?v=lJvHSCOdnCY&ab_channel=
OracleDevelopers
https://www.youtube.com/watch?v=EBKHdV-_rIc&ab_channel=
OracleLearning
https://www.youtube.com/watch?v=rFJXRFT_B0c&ab_channel=
OracleDevelopers
https://stackabuse.com/
reading-and-writing-yaml-files-in-java-with-snakeyaml
https://stackoverflow.com/questions/40124238/
javafx-rotate-an-object-without-turning-the-axis
https://stackoverflow.com/questions/19960368/
how-to-make-sense-of-javafx-triangle-mesh
https://stackoverflow.com/questions/58709048/
how-to-pass-complex-objects-into-the-fxml-via-namespace-and-fxmlloader
https://what-when-how.com/wp-content/uploads/2012/07/
tmpc2f9555

10 Seznam obrázků

1. Pyraminx - https://www.amazon.com/CuberSpeed-Qiming-Pyramid-Black-MoFangGe/dp/B07811MJCP
2. Ernö Rubik - https://cs.wikipedia.org/wiki/Ern%91_Rubik#/media/Soubor:Erno_Rubik_Genius_Gala_2014
3. Java - https://logos-world.net/java-logo)
4. JavaFX - https://en.wikipedia.org/wiki/JavaFX)
5. Maven - https://commons.wikimedia.org/wiki/File:Apache_Maven_logo)
6. SceneBuilder - https://gluonhq.com/products/scene-builder)
7. InteliJ IDEA - https://commons.wikimedia.org/wiki/File:IntelliJ_IDEA_Icon)
8. LaTeX - https://freebiesupply.com/logos/latex-logo)
9. Kostka v prostoru - https://what-when-how.com/wp-content/uploads/2012/07/tmpc2f9555.png)
10. Souřadnicová soustava - https://mathinsight.org/media/applet/image/large/cartesian_coordinate_axes_3d.png)
11. Barevné rozložení kostky - https://cubesolve.com/img/yellow-cross-dot.svg)
12. Notace kostky - https://content.instructables.com/FPW/FT89/IANRUS8P/FPWFT89IANRUS8P.png?auto=webp&fit=bounds&frame=1)

Všechny obrázky reprezentující stav kostky v tutorialu jsou upravenými verzemi obrázku z https://cubesolve.com/img/yellow-cross-dot.svg)