Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Grafové algoritmy vo vyučovaní informatiky na SŠ

Záverečná práca doplňujúceho pedagogického štúdia

2016
Bc. Paula Budzáková

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Grafové algoritmy vo vyučovaní informatiky na SŠ

Záverečná práca doplňujúceho pedagogického štúdia

Študijný program:	Informatika
Študijný odbor:	7656 Učiteľstvo akademických predmetov
Školiace pracovisko:	Katedra základov a vyučovania informatiky
Školiteľ:	RNDr. Michal Winczer, PhD.

Bratislava, 2016
Bc. Paula Budzáková

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Paula Budzáková
Študijný program: informatika (Certifikovaný interdisciplinárny program, iné N st., denná forma)
Študijný odbor: učiteľstvo akademických predmetov
Typ záverečnej práce: Záverečná práca doplňujúceho pedagogického štúdia
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Grafové algoritmy vo vyučovaní informatiky na SŠ
Graph algorithms in informatics education in upper secondary school

Cieľ:

1. Preskúmať existujúce prostredia na vyučovanie grafovych algoritmov .
2. Preskúmať náročnosť úloh používaných v týchto prostrediach a v informatických súťažiach,ktoré sú zamerane na grafove algoritmy.
3. Navrhnúť program a sadu úloh na testovanie grafovych algoritmov.
4. Otestovať úlohy na žiakoch strednej školy a následne vyhodnotiť výsledky.

Kľúčové slová: stredná škola, informatika, grafové algoritmy

Vedúci: RNDr. Michal Winczer, PhD.
Katedra: FMFI.KZVI - Katedra základov a vyučovania informatiky
Vedúci katedry: doc. RNDr. Zuzana Kubincová, PhD.

Spôsob sprístupnenia elektronickej verzie práce:
bez obmedzenia

Dátum zadania: 08.12.2015

Dátum schválenia: 26.02.2016 doc. RNDr. Zuzana Kubincová, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie: Ďakujem svojmu školiteľovi RNDr. Michalovi Winczerovi, PhD. za odborné vedenie pri písaní záverečnej práce.

Abstrakt

Záverečná práca sa zaoberá grafovými algoritmami vo vyučovaní informatiky na strednej škole, presnejšie prehľadávacími algoritmami na grafovej štruktúre. V práci sme implementovali program, ktorý je možné využiť pri výklade jednotlivých algoritmov, alebo ako overovací nástroj pre žiakov.
V praktickej časti tejto práce sme využitie implementovaného programu testovali na vybranej strednej škole.

Kľúčové slová: Grafové algoritmy, Prehľadávanie do šírky, Prehľadávanie do hĺbky, Informatika, Stredná škola

Abstract

The diploma thesis deals with teaching graph algoritms at a secondary school. More precisely it's aimed at algorithms used for traversing graph structures. As the part of the thesis, we implemented program which can be used to present graph algorithms or as a validation tool for students.
In the practical part of this work we tested our program at selected secondary school.

Keywords: Graph Algorithms, Breadth First Search, Depth First Search, Informatics, Secondary school

Obsah

Zoznam obrázkov

Úvod

Vo vyučovaní informatiky sa stretávame s rôznymi témami. Najčastejšie je to práca s textovým alebo grafickým editorom. Ak by sme sa zúčastnili hodín informatiky na rôznych slovenských školách, zistili by sme, že úroveň vyučovania sa výrazne diametrálne odlišuje. Rôznu úroveň vyučovacieho procesu informatiky môžeme odôvodniť viacerými možnými faktormi, ktoré ju môžu ovplyvňovať. No algoritmické myslenie u jednotlivých žiakov sa dá rozvíjať aj bez vysokokvalitných zariadení. Ak by sme mali k dispozícii dostatok edukačných softvérov, vedeli by sme ich využiť nielen vo vyučovaní informatiky, ale výrazne by nám pomohli vo vyučovaní rôznych predmetov.

Pojem graf by sme v štátnom vzdelávacom programe hľadali márne. Keď sa už ale pozrieme na inovovaný štátny vzdelávací program, nájdeme časť okruhu, ktorá je zameraná na štruktúry ako také, a tu už slovíčko graf bije do očí. My sme sa rozhodli zamerať na grafové štruktúry a následné ich prehľadávacie algoritmy. Navrhli sme program, ktorý je možné využiť pri samotnom výklade alebo pri overovaní riešenia. Počas zisťovania informácií o vyučovaní grafov na rôznych stredných školách sme zistili, že túto tému môžeme nájsť na seminároch pre budúcich maturantov z informatiky. Práve takýto seminár sme využili na testovanie programu, ktorý sme testovali s pomocou úloh z pripraveného pracovného listu.

V testovaní sme zistili, že takáto pomôcka pri vyučovaní žiakom pomohla a veľmi ju ocenili. Vyučovanie témy grafových algoritmov pomocou programu bolo pre nich zaujímavejšie a viac prístupnejšie. Takáto pomôcka pri vyučovaní by bola dobrým nástrojom, ako sprístupniť jednoduchšie témy z tém obsiahnutých v seminári aj do vyučovania hodín informatiky pre všetkých žiakov.

Kapitola 1

Východiská práce

Súčasný stav vyučovania predmetu informatika by sme mohli priradiť k predškolskému veku dieťaťa, či už v základných alebo stredných školách. Na Slovensku sme ešte nedospeli do takého štádia, kedy by sa informatika nevyučovala ako len informačné a komunikačné technológie, ale hlavne ako skutočná informatika. Informatika by nemala byť len o grafických editoroch a o zvládnutí ovládania rôznych programov na úpravu textu alebo tabuliek. Je veľmi dôležité žiakov viesť aj k programovaniu a k správnemu algoritmickému mysleniu. Samotná výučba informatiky by mala byť správnou kombináciou algortimického myslenia a praktickej zručnosti.

1.1 Štátny vzdelávací program

„Cieľom informatiky na strednej škole je naučiť žiakov základné pojmy, postupy a prostriedky informatiky. Vychovávať ich k efektívnemu využívaniu prostriedkov informačnej civilizácie, s rešpektovaním právnych a etických zásad používania informačných technológií a produktov.“[12] Presne takto formuluje cieľ informatiky **Štátny vzdelávací program ISCED 3A** (ďalej len ŠVP) [12]. Výchovno-vzdelávací proces stanovený ŠVP je daný tak, aby žiaci :

- sa naučili pracovať v prostrediach bežných aplikácií a efektívne vyhľadávať informácie uložené na pamäťových kartách alebo na sieti,
- si rozvíjali logické myslenie, svoju osobnosť, tvorivosť, sebakritickosť a snažili sa o vzdelávanie,
- si rozvíjali svoje schopnosti kooperácie a komunikácie,
- sa naučili komunikovať cez sieť,
- nadobudli schopnosti pre výskumnú prácu,

- si rozvíjali algoritmické myslenie a programátorské zručnosti.

V ŠVP je vzdelávací obsah informatiky rozdelený na 5 základných tematických okruhov:

1. *Informácie okolo nás* - okruh, ktorý rozširuje učivo základnej školy. Žiaci si budujú základy informatiky ako vednej disciplíny, vedia rozlíšiť jednotlivé typy informácií, vedia ovládať aplikácie na spracovávanie špecifických informácií. Žiaci dokážu prezentovať získané informácie, uchovávať ich a prenášať medzi aplikáciami.

2. *Komunikácia prostredníctvom IKT* - Žiaci sa oboznamujú so základnými pojmami internetu, webovým prehliadačom, počítačovou sieťou a bezpečnosťou na internete.

3. *Postupy, riešenia problémov, algoritmické myslenie* - Žiaci v tomto okruhu získajú schopnosť uvažovať nad riešením problému pomocou IKT a algoritmické myslenie. Naučia sa rôzne postupy pri riešení úloh z rôznych oblastí.

4. *Princípy fungovania IKT* - Žiaci by sa v tomto tematickom okruhu mali zoznámiť s princípmi práce počítača a jeho jednotlivými časťami. Mali by zvládať funkcie a základné vlastnosti operačného systému a poznať rôzne softvérové druhy.

5. *Informačná spoločnosť* - posledný okruh sa zaoberá etickými, morálnymi a spoločenskými aspektami informatiky. Žiaci by sa mali oboznámiť s rôznymi možnosťami vzdelávania prostredníctvom IKT.

Každý okruh je rozdelený na obsahový a výkonový štandard. Oba štandardy sú písané formou osnovy, čo všetko by sa v danom tematickom okruhu malo prebrať a čo všetko by žiaci mali ovládať.
Od začiatku školského roka 2015/2016 vstúpil do platnosti **inovovaný štátny vzdelávací program** (ďalej len inovovaný ŠVP) [13], ktorý predstavuje modifikáciu pôvodného ŠVP. Ciele predmetu a počet tematických okruhov zostal zachovaný, zmenili sa, okrem posledného okruhu, ich názvy. Každý tematický okruh je rozdelený na menšie podokruhy. Každý podokruh obsahuje výkonový a obsahový štandard. Obsahový štandard podokruhu obsahuje vytýčené pojmy, vlastnosti a procesy, ktoré by mal po danej časti žiak ovládať:

1. *Reprezentácie a nástroje* - podokruhy: práca s grafikou, textom, prezentáciami, multimédiami, tabuľkami, informáciami a štruktúrami.

2. *Komunikácia a spolupráca* - podokruhy: prezentovanie informácie prostredníctvom webovej stránky, vyhľadávanie na webe, práca s nástrojmi na spoluprácu a zdieľanie informácií, práca s nástrojmi na komunikáciu.

3. *Algoritmické riešenie problémov* - podokruhy: analýza problému, jazyk na zápis riešenia, pomocou postupnosti príkazov, pomocou nástrojov na interakciu, pomocou premenných, pomocou cyklov, pomocou vetvenia, interpretácia zápisu riešenia, hľadanie a opravovanie chýb.

4. *Softvér a hárdver* - podokruhy: práca so súbormi a priečinkami, práca v operačnom systéme, počítač a prídavné zariadenia, práca v počítačovej sieti a na internete, práca proti vírusom a špehovaniu.

5. *Informačná spoločnosť* - podokruhy: bezpečnosť a riziká, digitálne technológie v spoločnosti, legálnosť používania.

Inovovaný ŠVP [13] má najviac rozdelený okruh algoritmického riešenia problémov, čo je veľmi dobré hlavne pre začínajúceho učiteľa alebo učiteľa, ktorý nemá veľmi dobrú prax s učením programovania. Podokruhy v tomto okruhu predstavujú presnú postupnosť, kde jednotlivé témy gradujú a správne nadväzujú. Keďže v súčasnej praxi je používaný inovovaný ŠVP, v našej práci budeme pracovať už len s touto verziou.
Grafové algoritmy vo vyučovaní informatiky priamo, či už v ŠVP alebo v inovovanom ŠVP, nikde nenájdeme. Je to téma, ktorá je zatiaľ na strednej škole aktuálna hlavne pre tých, ktorí majú záujem maturovať z informatiky a navštevovať informatické semináre. Je to téma, ktorej sa v súčasnosti venuje veľmi málo stredoškolských učiteľov a skôr sa s ňou môžeme stretnúť až na vysokej škole. To nám ale nebráni venovať sa tejto tematike v stredoškolskom prostredí. Grafové algoritmy v inovovanom ŠVP môžeme nájsť v podobe kombinácie dvoch tematických okruhov, a to okruhu *Reprezentácie a nástroje* a *Algoritmické riešenie problémov*.
Okruh Reprezentácie a nástroje je pri grafoch veľmi dôležitý. Aj keď žiaci môžu určité vedomosti z oblastí grafov nadobudnúť na matematike, je podstatné, aby o nich uvažovali aj s informatického hľadiska. Práve na to nám v inovovanom ŠVP slúži časť okruhu **Reprezentácie a nástroje - štruktúry**. Hlavné body výkonového obsahu okruhu sú:

- organizovať informácie do štruktúr – vytvárať a manipulovať so štruktúrami, ktoré obsahujú údaje a vzťahy (tabuľky, grafy, postupnosti obrázkov, čísel, ...),

- posudzovať vlastnosti operácií s rôznymi štruktúrami (napr. možnosť mazania, vkladania, vyhľadávania, ...),

- interpretovať údaje zo štruktúr – odvodzovať vzťahy zo zadaných údajov v štruktúre, prerozprávať informácie uložené v štruktúre vlastnými slovami.

Druhým, veľmi podstatným okruhom, je okruh **Algoritmické riešenie problémov**. V tomto okruhu sú pre nás podstatné všetky jeho časti od analýzy problému až po

hľadanie a opravovanie chýb. Je veľmi dôležité, aby žiak vedel interpretovať grafový algoritmus aj pomocou zápisu programovacím jazykom, kde je potrebné, aby žiak vedel správne používať základné programovacie témy. Musí vedieť analyzovať problém, aby si zvolil správnu metódu na jeho vyriešenie a rozmyslel si predbežnú postupnosť príkazov. Pomocou vetvenia, premenných a cyklov naprogramovať funkčný grafový algoritmus. Cieľom tejto práce nie je žiakov naučiť programovať grafové algoritmy. Cieľom je im pomôcť pri výučbe a upevňovaní informácií pomocou edukačného softvéru, ktorý im vizualizuje grafový algoritmus na nimi zadanom grafe. Aj keď priamo nepracujú s programovacím jazykom, je potrebné, aby vedomosti z vyššie spomenutého tematického okruhu mali aspoň sčasti nadobudnuté.

1.2 Bloomova taxonómia

Taxonómia vzdelávacích cieľov predstavuje veľmi užitočnú pomôcku pre učiteľa. Proces zámernej zámeny osobnosti žiaka a štrukturálne poňatie osobnosti stali za základom pre členenie taxonómií na kognitívne, afektívnne a psychomotorické vzdelávacie ciele.

Bloomova taxonómia je vo svete najznámejším pokusom o klasifikáciu kognitívnych vzdelávacích cieľov do jednotlivých úrovní. Je pomenovaná podľa amerického psychológia Benjamína Blooma [8]. Popisuje vzdelávanie ako proces, ktorý pomáha žiakom vo vývoji od nižších foriem myslenia k vyšším formám. Pôvodná Bloomova taxonómia, z roku 1956, má jednoduchú štruktúru, učivo neklasifikuje a nezaoberá sa jednotlivými fázami vyučovacieho procesu. Slavín [9] zdôrazňuje usporiadanie od jednoduchších úrovní ku komplexnejším. V Bloomovej taxonómii nájdeme v okruhu kognitívnych - poznávacích cieľov, postupne od jednoduchších k zložitejším, tieto úrovne:

- **Znalosť** - Na tejto úrovni si má žiak vybaviť, alebo znovu spoznať konkrétne poznatky, fakty, termín, a podobne z pamäte. Ide o reprodukovanie zapamätaného učiva.
- **Porozumenie** - Na tejto úrovni je žiak schopný porozumieť významu obsahu učiva, ktorý mu bol sprostredkovaný slovne alebo obrazom. Dokáže vysvetliť obsah vlastnými slovami, odfiltrovať nepodstatné a formulovať to, čo je v texte obsiahnuté explicitne.
- **Aplikácia** - na základe teoretických poznatkov a porozumenia vzťahov a pravidiel vie žiak na tejto úrovni aplikovať získané vedomosti do konkrétnej situácie.
- **Analýza** - na úrovni analýzy vie žiak vytvoriť rozbor informácie na jednotlivé menšie prvky, vie určiť vzťahy a ich vzájomnú interakciu a hierarchiu.
- **Syntéza** - na úrovni syntézy je žiak schopný zložiť jednotlivé menšie prvky do jedného uceleného celku.

- **Hodnotiace posudzovanie** - žiak v úrovni hodnotiaceho posudzovania dokáže posúdiť správnosť, presnosť, efektívnosť svojho riešenia

Myšlienkové a kognitívne operácie a aj vzdelávacie ciele vyšších úrovní sú podmienené dosiahnutím vzdelávacích cieľov na jednoduchších úrovniach. Nezvládnutie alebo neúplne zvládnutie jednej úrovne obvykle zakladá problémy pri dosahovaní vyšších úrovní.

Taxonómia Blooma pomáha učiteľom zistiť, či sú ich požiadavky na žiaka vyvážené, či požiadavky smerom k učivu nie sú len na nižších úrovniach na úkor zložitejších výkonov žiakov.

V druhej polovici 90. storočia bola Bloomova taxonómia významne revidovaná tímom, pod vedením D. R. Krathwola [1]. Výstupom sa stala nová dvojdimenzionálna taxonómia vzdelávacích cieľov, ktorá zahrňuje dimenziu kognitívnych poznatkov a dimenziu kognitívnych procesov. Reevidovaná Bloomova taxonómia je znázornená na obrázku 1.1.

	DIMENZIA KOGNITÍVNEHO PROCESU					
ZNALOSTNÁ DIMENZIA	**1. Zapamätať si**	**2. Porozumieť**	**3. Aplikovať**	**4. Analyzovať**	**5. Hodnotiť**	**6.Tvoriť**
A. faktické poznatky	vymenovať, uviesť	stručne vyjadriť, zhrnúť	roztriediť, klasifikovať	usporiadať	zatriediť vybrať	kombinovať
B. konceptuálne poznatky	opísať	interpretovať rozoznať	experimentovať	vysvetliť porovnať	odhadnúť určiť	plánovať načrtnúť
C. procedurálne poznatky	usporiadať	Predpokladať	vypočítať riešiť	rozlišovať znázorniť	vyvodiť, usúdiť	vytvoriť poskladať navrhnúť
D. metakognitívne poznatky	použiť	Spracovať	skonštruovať	vytvoriť	vykonať vyjadriť	aktualizovať zdokonaliť

Obr. 1.1: Revidovaná Bloomova taxonómia [7]

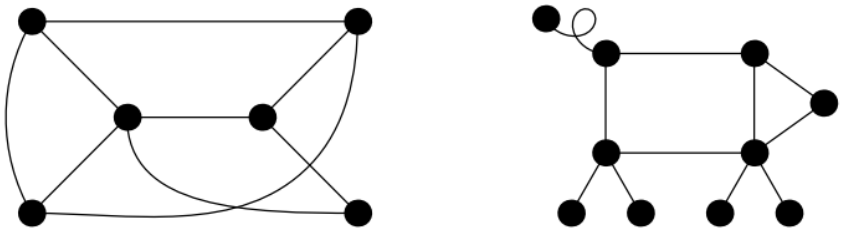
1.3 Graf a grafové algoritmy

Grafové algoritmy sú veľmi dôležitým nástrojom abstrakcie pri riešení mnohých problémov v informatickej praxi, preto sme sa rozhodli v našom výskume zamerať práve na ne. No v tematike grafových algoritmov je potrebné ovládať základné znalosti o grafových reprezentáciách. Ak žiaci takéto vedomosti a znalosti neovládajú, odporúčame im graf a grafové reprezentácie priblížiť. V tejto časti kapitoly v podkapitole 1.3.1 sme spísali a vysvetlili základné grafové znalosti, ktoré sú v jednoduchosti vysvetlené pre čitateľnosť a porozumenie bez predchádzajúcej vedomosti o teórii grafov. Podkapitola 1.3.2 opisuje stromové grafové štruktúry, ktoré predstavujú špeciálne grafy často využívané v informatickej oblasti.

1.3.1 Graf

Graf [11] v teórii grafov predstavuje niečo iné, ako graf funkcie alebo stĺpcový graf, ktorý môžeme použiť napríklad v aplikácii Excel. Graf predstavuje dvojicu vrcholov a medzi nimi existujúcich hrán. Existenciu hrany často vyjadrujeme ako „ dvojica vrcholov je spojená hranou“ alebo „z prvého vrchola vedie hrana do druhého vrchola“. Ďalšie vrcholy, s ktorými je vrchol spojený hranou, sa nazývajú *susediace vrcholy*.

Graf je abstrakný pojem a často ho kreslíme na papier. Nielen na papieri, ale aj v programových vizualizáciách vrcholy grafu kreslíme ako „guličky“ a hrany ako čiary, ktoré tieto „guličky“ spájajú, ako môžeme vidieť na obrázku 1.2. Ak by sme pomocou grafu chceli znázorniť vzťahy medzi ľuďmi, namiesto plných vrcholov použijeme vrcholy obsahujúce mená ľudí a hrany medzi nimi by predstavovali vzťah. Pomocou grafu



Obr. 1.2: Grafická reprezentácia dvoch rôznych grafov[11]

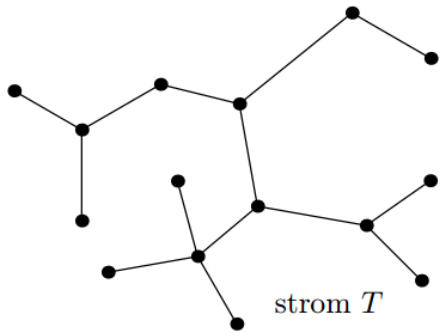
môžeme znázorniť napríklad aj vlakovú sieť koľajníc na mape Slovenska. Každý vlakový úsek prechádza cez rôzne mestá, ktoré predstavujú vlakové stanice. Jednotlivé vlakové stanice v našom grafe znázorníme ako vrcholy a vlakové koľajnice medzi nimi ako hrany grafu. Výsledný graf bude mať rovnakú grafickú formu ako grafy na obrázku 1.2. Každá časť koľajníc medzi stanicami má svoju dĺžku. Túto dĺžku môžeme v grafe zaznamenať tak, že hodnoty úsekov pripíšeme ku hrane, ktorá predstavuje daný úsek. Ak hrany v grafe majú nejakú hodnotu, dostávame **ohodnotený graf**.

Niektoré vlakové úseky nemusia podporovať obojsmernú dopravu. Niektorými vieme ísť zo západu na východ, opačne alebo obojsmerne. Na grafovej reprezentácii to ale nevieme rozoznať. Aby sme v grafe zaznamenali správnu štruktúru vlakovej siete, každú hranu v grafe označíme šípkou, podľa toho, ktorým smerom vedie. Hrany, ktoré vedú obojsmerne, označíme šípkami na oboch jej koncoch. Takýmto označením hrán v grafe dostaneme **orientovaný graf**. Graf, v ktorom nie je určená orientácia hrán, je graf **neorientovaný**.

1.3.2 Strom

Strom [3] je veľmi dôležitou triedou grafu. Má veľmi jednoduchú štruktúru a veľa vecí sa v ňom počíta až triviálne. Základnou vlastnosťou stromu na obrázku 1.3, je:

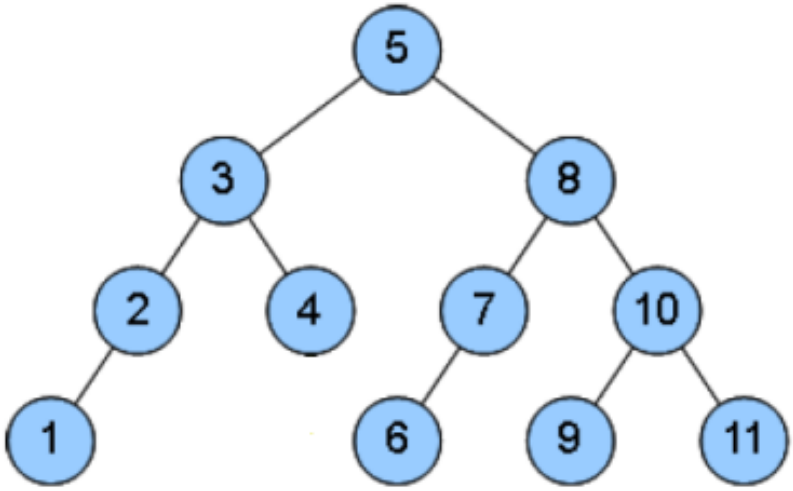
- Strom je graf, ktorého celkový počet hrán je o 1 menší ako celkový počet vrcholov



Obr. 1.3: strom [3]

Každý strom môžeme vytvoriť z jedného vrcholu postupným prilepovaním ďalších vrcholov k nemu. V algoritmoch sa častejšie stretávame so stromami, ktoré majú jeden hlavný vrchol, nazývaný **koreň**. Pri práci so stromami začíname v hlavnom vrchole - koreni - a z neho pokračujeme do ďalších. Predstavme si, že hrany stromu sú orientované smerom od koreňa. Vrcholy, z ktorých nevychádza žiadna orientovaná hrana, sa nazývajú *listy*. Ostatné vrcholy sú vnútorné vrcholy stromu.

Do obrázku stromu orientáciu hrán zvyčajne nekreslíme. Hrany sú spravidla orientované z vrchola ležiaceho vyššie do vrchola ležiaceho nižšie. Preto kreslíme vrcholy idúce z vyššieho vrchola pod vyšší vrchol a zarovnávame zľava doprava. Vrchol, ktorý vychádza z predchádzajúceho vyššieho vrchola je jeho synom. Ak počet synov je rovný dvom, hovoríme o **binárnom strome** [3]. Typickým príkladom binárneho stromu je rodokmeň človeka. Každý vrchol stromu obsahuje vlastnú informáciu, ktorá môže byť číselného alebo textového obsahu. Každý vrchol môže mať dvoch synov, ktorí sa označujú ľavý a pravý. Binárne stromy sa často používajú pre prezentáciu čiastočného usporiadania množiny údajov, ktorej prvky sa majú vyberať na základe obsiahnutej informácie vo vrchole. Strom, v ktorom všetky informácie ľavého podstromu sú menšie ako informácie vrchola a informácie pravého podstromu sú väčšie ako informácie vrchola, sa nazýva **binárny vyhľadávací strom**, znázornený na obrázku 1.4.



Obr. 1.4: Binárny vyhľadávací strom. V ľavých synoch je hodnota kľúča menšia ako v koreni/otcovi, v pravých synoch je hodnota kľúča väčšia. [3]

1.4 Prehľadávacie algoritmy

Ak by sme mali vyriešiť úlohu, v ktorej by sme sa mali dostať z bodu A do bodu B pomocou bludiska, jedným možným riešením by bolo vyskúšať všetky možné cesty, až by sme sa dostali do cieľa. Správne riešenie môžeme dostať aj pomocou grafového algoritmu tak, že si bludisko reprezentujeme ako graf. Ak by sme sa do cieľa mali dostať za krátky stanovený čas, možnosť prejsť všetky možné cesty, čiže v grafe hrany, by nebola veľmi efektívna. Efektívny prechod grafu musí spĺňať nasledujúce body:

- každú hranu prejdeme maximálne jedenkrát, čiže raz tam a raz späť,
- tou istou hranou sa vrátime, až keď z vrchola nevedie ďalšia cesta,
- hranou vedúcou do už navštíveného vrchola sa hneď vrátime.

Efektívny prechod grafom má dve implementácie:

1. *Prehľadávanie do hĺbky (Depth First Search)*
2. *Prehľadávanie do šírky (Breadth First Search)*

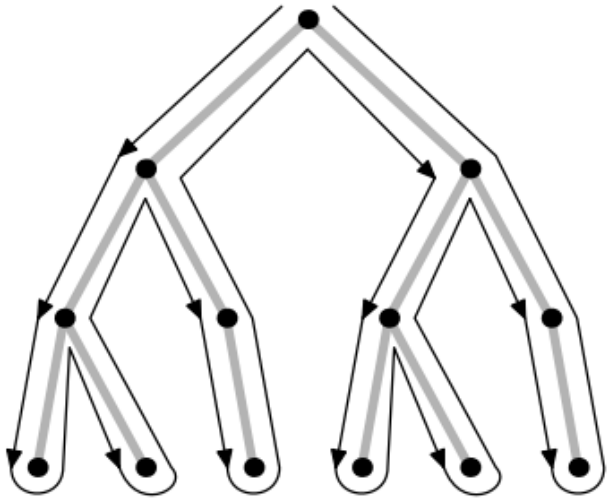
1.4.1 Prehľadávanie do hĺbky

Použijeme znova úlohu s bludiskom. Pred vstupom do bludiska si uviažeme koniec nite o strom a vstúpime do vnútra. Niť držíme počas celej cesty bludiskom. Na prvom rázcestí, čiže v prvom vrchole grafu, si vyberieme jednu možnú cestu, cez ktorú prejdeme do ďalšieho vrchola. Aby sme nemali zmätok v tom, ktoré hrany sme už prešli, každú

hranu, ktorou prechádzame, si označíme kriedou, a to po celej jej dĺžke. Hrana, ktorou sme prešli, je označená od jej začiatku až po jej koniec. V každom vrchole, do ktorého vstúpime, urobíme nasledujúce kroky:

- Pokiaľ na zemi nájdeme položenú niť, tak vieme, že vrchol sme už navštívili. Keď sa budeme vracať späť pomocou namotávania nite na klbko, určite sa do neho zase vrátime. Vrchol neprehľadávame, ale vrátime sa späť do predchádzajúceho vrchola.
- Pokiaľ ale vo vrchole žiadnu niť nenájdeme, tak sa vydáme prvou, ešte neprejdenou hranou, čiže neoznačenou kriedou. Prechádzaním neoznačených hrán vychádzajúceho vrchola vrchol prehľadávame. Ak takáto hrana neexistuje, znamená to, že vrchol sme úplne prehľadali, namotávame niť a vraciame sa do predchádzajúceho vrchola.

Týmto spôsobom prehľadáme celé bludisko, až sa postupným namotávaním nite dostaneme úplne na začiatok, čiže do vychádzajúceho vrchola.
Označovanie prejdených hrán kriedou slúži na to, aby sme sa v bludisku nezacyklili, čiže aby sme nechodili stále dookola cez tie isté hrany. Ťahajúca niť počas celej cesty slúži k tomu, aby sme našli potrebnú cestu späť k vrcholu, ktorým sme vstupovali do bludiska. Obrázok 1.5 znázorňuje prehľadávanie do hĺbky [3].

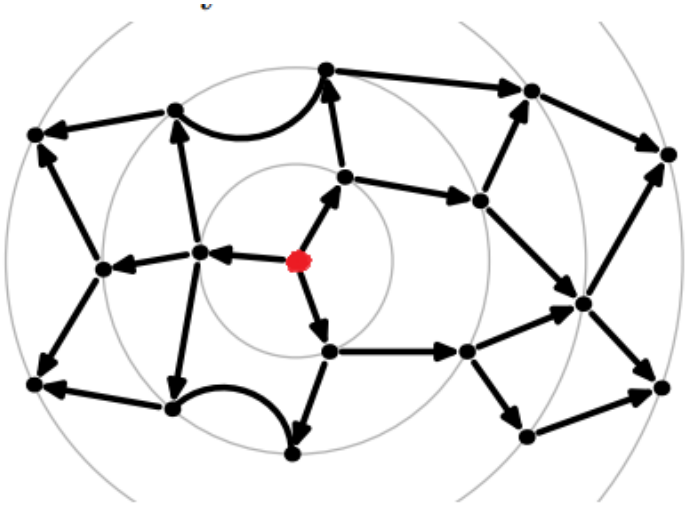


Obr. 1.5: Prehľadávanie do hĺbky[3]

1.4.2 Prehľadávanie do šírky

Predstavme si, že sa na vychádzajúci vrchol postaví tisíc ľudí, ktorí chcú urobiť rekord. Všetci naraz začnú graf prehľadávať. Keď sa cesta rozdelí, tak sa rozdelí aj dav utekajúci hranou. Predpokladáme, že všetky hrany v grafe sú rovnako dlhé. Graf prehľadávame všetci spoločne po rovnakých úrovniach. V prvej úrovni sa všetci dostaneme do

vrcholu, do ktorého vedie z vychádzajúceho vrchola len jedna hrana. V druhej úrovni sa všetci dostaneme do vrchola, ktorý je od vychádzajúceho vrchola vzdialený dve hrany, a tak ďalej, až kým neprídeme k vrcholu, z ktorého už nevedie žiadna neprejdená hrana. V takomto prípade je graf celý prehľadaný. Princíp algoritmu prehľadávania do šírky [3] je znázornený na obrázku 1.6. Červená bodka predstavuje vychádzajúci vrchol.



Obr. 1.6: Prehľadávanie do šírky[3]

1.5 Edukačný softvér

Edukačný softvér [2] je vyvinutý na poznávanie, rozvoj informačnej gramotnosti a na podporu učenia sa. Je vytvorený ako nástroj pre učiteľa alebo žiaka na učenie. Ponúka prostredie na poznávanie a modelovanie rôznych situácií. Edukačný alebo didaktický softvér vo všeobecnosti:

- rozvíja informačnú gramotnosť,
- podporuje individuálny prístup k tomu, kto sa na ňom učí,
- poskytuje spätnú väzbu.

V školskom prostredí sa stretávame s rôznymi softvérami slúžiacimi na edukačné účely, ktoré na to primárne ale nie sú určené. Skutočný edukačný softvér sa v našich školách vyskytuje najmenej. Existuje mnoho vlastností, vďaka ktorým vieme edukačný softvér klasifikovať. Najčastejšie klasifikujeme podľa vyučovacieho predmetu, na ktorý je určený, podľa vzdelávacej paradigmy, podľa použitia počítača a pod. Vzdelávacie paradigmy, podľa ktorých klasifikujeme, môžeme rozdeľovať na:

1. *Inštruktívna paradigma* - založená na behaviorálnom učení. Edukačný softvér obsahujúci túto paradigmu vykonáva „programové učenie“. Učebný materiál je

pomocou menších častí vedomostí spojených s úlohou predkladaný študentovi. Študent odpovedá na úlohy a dostáva spätnú väzbu o správnosti odpovede. Pri nesprávnom zodpovedaní softvér preberá úlohu doučovateľa. To sa opakuje, až kým študent neprejde danú časť úspešne a prejde na ďalšiu.

2. *Objaviteľská paradigma* - základom je u študenta podporiť skúmanie modelu alebo simulácie. Prebieha učenie sa objavovaním a skúsenostné učenie.

3. *Paradigma hypotéz* - založená na teóriách Piageta, Paperta a Poppera. Softvér poskytuje študentovi prostredie laboratória, v ktorom môže testovať a vyjadrovať svoje hypotézy.

4. *Oslobodzujúca paradigma* - cieľom je získať viac času pre samotné učenie redukovaním pracovného úsilia pri spracovávaní informácií.

Edukačný softvér pre stredné školy je určený žiakom, ktorí sa podľa *Piagetovej teórie kognitívneho vývinu* nachádzajú v *štádiu formálnych operácií*, pre ktoré je typické abstraktné myslenie a logické uvažovanie. Dieťa v tomto štádiu je schopné hypoteticky-induktívneho usudzovania, pri experimentovaní systematicky obmieňa premenné a hľadá pravidlá. Myšlienkové operácie sa spájajú do zložitejšej štruktúry a dieťa s nimi dokáže pracovať priamo alebo nepriamo.
Z pohľadu používateľa, či už učiteľa alebo žiaka, by mal edukačný softvér spĺňať určité kritériá, ktoré sa netýkajú len používateľského prostredia. Softvér je pre používateľa zaujímavý, ak je :

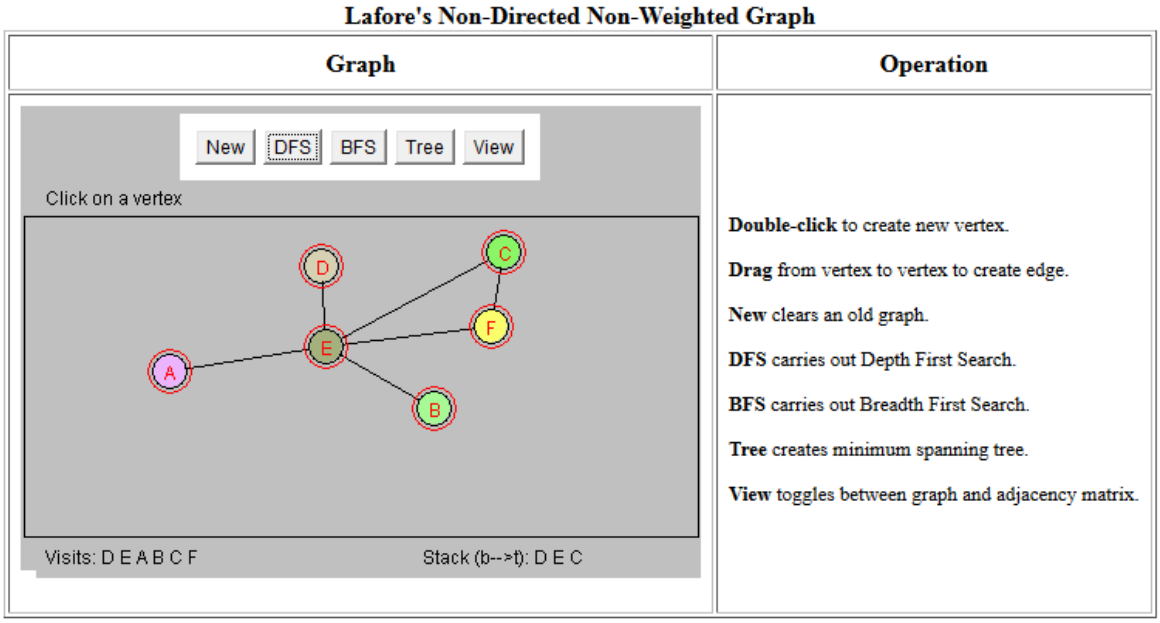
- jednoducho a intuitívne ovládateľný,
- podporuje individuálny prístup používateľa,
- má príťažlivý vzhľad,
- obrázky, animácie, zvuk sú dostatočne kvalitné,
- obsahuje najnovšie technológie.

1.5.1 Existujúce programy

Väčšina existujúcich edukačných programov na grafové algoritmy bola primárne vytvorená pre používanie na vysokých školách. Autormi softvérov sú najčastejšie vysokoškolskí učitelia, ktorí program navrhli a vytvorili pre potreby svojich študentov.
Edukačný softvér určený priamo pre potreby strednej školy, ktorý už existuje a je k dispozícii, sme nenašli. Viacerí stredoškolskí učitelia disponujú s veľmi jednoduchou verziou softvéru tohto typu, ktoré boli najčastejšie vytvorené ako záverečný projekt rozširujúceho seminára z informatiky.

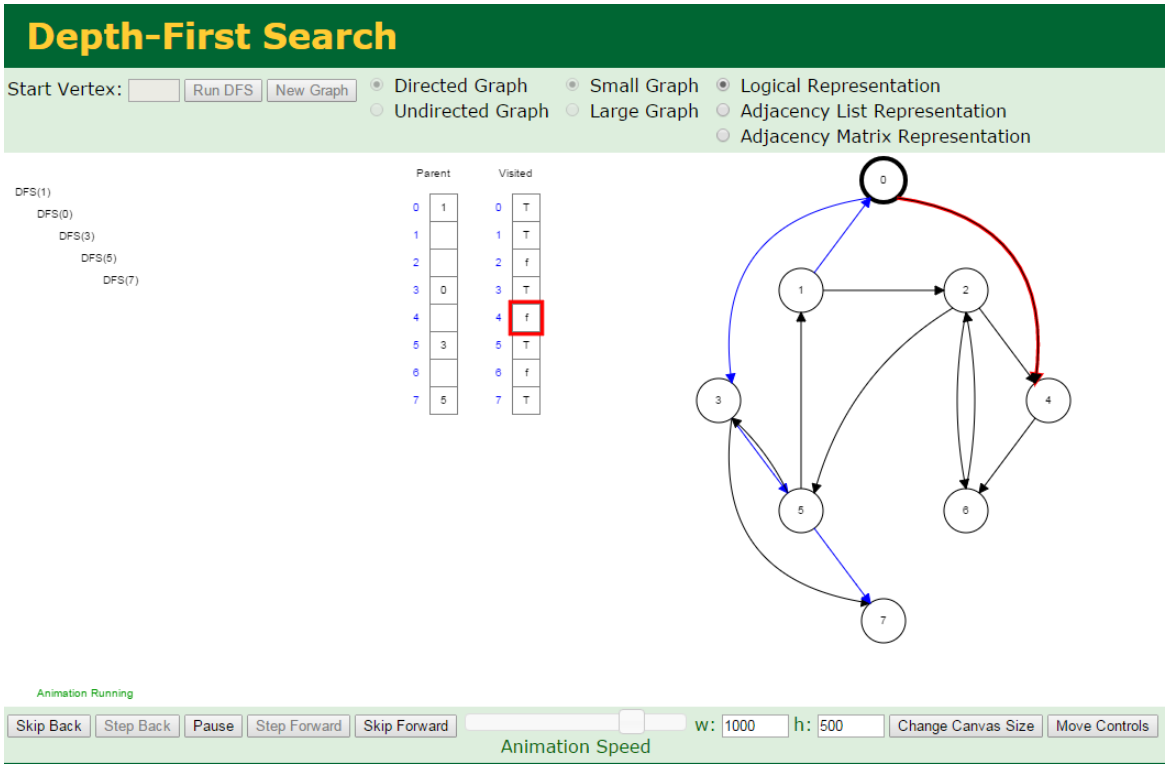
Edukačný softvér, znázornený na obrázku 1.7, ktorý vytvoril Dr. Chokchai "Box" Leangsuksun na Technickej Univerzite v Louisiane [6], ponúka žiakovi precvičovanie prehľadávania do šírky a do hĺbky na grafe. Primárne bol určený ako pomôcka predmetu študentom na tejto univerzite. Vzhľad softvéru je celkom strohý, na prvý pohľad nie veľmi príťažlivý. Ovládanie je celkom jednoduché a intuitívne, aj keď sa niekedy objaví komplikácia. Okno softvéru je rozdelené na dve časti. Pravá časť obsahuje potrebné informácie k ovládaniu softvéru, ľavá strana obsahuje vykresľovaciu časť grafu s tlačidlami funkčnosti.

Študent má možnosť naklikať si ľubovoľný neorientovaný graf, na ktorom spustí ním vybrané prehľadávanie. Vytvorený vrchol je farebne odlišný od ostatných a obsahuje v sebe automaticky vygenerovaný kľúč vo forme postupnosti písmen abecedy tak, v akom poradí boli vrcholy vytvorené. Nie je možná dodatočná úprava kľúča alebo úplná zmena jeho obsahu. Softvér neponúka študentovi možnosť si graf uložiť ani načítať. Upravovanie grafu je vyriešené veľmi jednoducho, nie je možné vymazávanie samostatných vrcholov alebo hrán. Graf vieme upraviť jeho vymazaním a opätovným vytvorením. Nie je možná ani úprava grafu posunutím vrcholov ťahaním myši. Softvér nevyužíva animáciu ani zvuk. Študent má na výber medzi vizuálnym grafom, ktorý si naklikal, alebo medzi reprezentáciou grafu pomocou matice susednosti. Individuálny prístup k používateľovi ponúka len vo vytváraní grafu. Softvér neobsahuje žiadne úlohy, neponúka ani žiadnu kontrolu študentovej odpovede, pretože študent si môže len skontrolovať svoju postupnosť prehľadávania, ktorú má niekde na papieri, s postupnosťou prehľadávania, ktorú vygeneroval softvér.



Obr. 1.7: Softvér vytvorený na Technickej univerzite v Louisiane.[6]

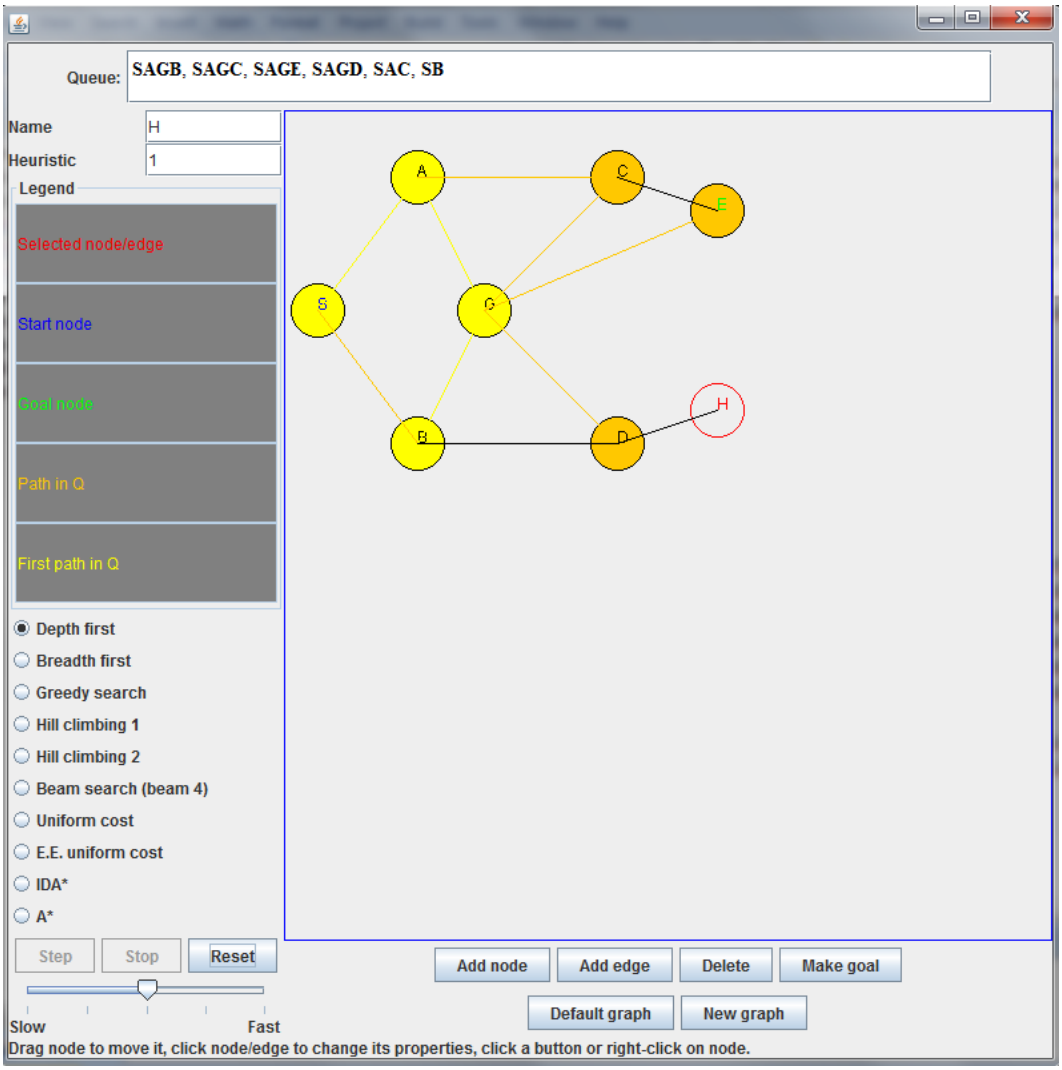
Profesor David Galles z Univerzity San Francisko [4] vytvoril veľmi pekný edukačný softvér na vizualizácie binárnych vyhľadávacích stromov, prehľadávanie do šírky a do hĺbky, ktorý aj môžeme vidieť na obrázku 1.8. Softvér síce neponúka študentovi naklikať si vlastný graf, ale ponúka mu na výber z množiny už vytvorených grafov. Grafy sú rozdelené na dve množiny - veľké a malé grafy, ktoré môžu byť orientované alebo neorientované. Grafy sa v jednotlivých množinách náhodne generujú. Rozmiestnenie a kľúče vrcholov sú v danej množine stále rovnaké, menia sa len hrany, čím vzniká nový graf. Každý vytvorený graf má dve varianty, a to buď s ohodnotenými alebo neohodnotenými hranami. Graf je možné zobraziť v logickej reprezentácii, v reprezentácii maticou susednosti alebo pomocou zreťazeného zoznamu. Študent si vpísaním kľúča



Obr. 1.8: Softvér vizualizujúci prehľadávanie do hĺbky.[4]

vrcholu do editu zvolí počiatočný vrchol prehľadávania a tlačidlom spustí prehľadávanie. Začne sa vizualizovať veľmi dobrá animácia prehľadávania. Počas animácie sa zvýrazňujú vrcholy, aktuálne hrany, cez ktoré prehľadáva a na ľavú stranu od grafu sa zapisuje postupnosť vrcholov, v ktorých sa prehľadávanie už uskutočnilo. Postupnosť zapisuje vo forme DFS (hodnota vrcholu), čo znamená, že v danom vrchole sa uskutočnilo celé prehľadávanie. Študent má k dispozícii nastavenie samotnej animácie. Animáciu môže zastaviť, spustiť, alebo spúšťať len po krokoch. Taktiež si môže upraviť aj rýchlosť prehrávania animácie. K dispozícii má aj možnosť upraviť si veľkosť animačného plátna. Keďže si študent graf v softvéri nevytvára, nemá možnosť úpravy, ukladania ani načítania vygenerovaných grafov.

Za veľké plus považujeme kvalitnú animáciu a veľmi dobre vytvorené grafy, ktoré študentovi ponúkajú dobrý spôsob pochopenia samotného prehľadávania. Softvér vizualizuje prehľadávanie aj s využitím dátovej štruktúry zásobníka, ktorý je zobrazený po celý čas behu programu na ľavej strane grafu. Počas priebehu algoritmu sa vkladajú a menia jeho hodnoty, čo študentovi ponúka možnosť pochopiť algoritmus aj priamo z programátorského hľadiska. JSearch Demo je softvér na vizualizáciu vyhľadávacích al-



Obr. 1.9: JSearch Demo softvér .[10]

goritmov, ktorý vytvorili autori El Tramo a Joost Vennekens [10]. Tento softvér, ktorý je znázornený na obrázku 1.9, nie je určený pre žiakov stredoškolského vzdelania ani pre žiakov prvých ročníkov vysokej škole. Softvér je vytvorený na vysokej úrovni, obsahuje viac ako desať prehľadávacích algoritmov, ktoré nie sú primárne určené len na prehľadávanie grafov.

Softvér sa dá veľmi ťažko ovládať intuitívne, prostredie obsahuje veľa podobných farieb, čo nepôsobí veľmi dobre, skôr chaoticky a neprehľadne. Študent si vie graf vytvárať sám, vie ho upravovať a meniť obsah vrcholov aj dodatočne, nielen pri vytváraní.

K dispozícii má aj prednastavený graf od autorov, ktorý si môže tiež upravovať a meniť. Pri vymazávaní vrcholu vymaže s ním aj všetky hrany, ktoré z neho vedú. Načítavanie a ukladanie grafov nie je k dispozícii.
Vybraný algoritmus na grafe vizualizuje postupne, aj keď s tým niekedy mal softvér problém, odvizualizoval tri kroky a prestal. Legenda je umiestnená na dosť nešťastnom mieste. Pôsobí ako tlačidlá, ktoré slúžia k vizualizácii. Taktiež farby jednotlivých krokov vizualizácie nie sú dobre zvolené. Farby vnútra vrcholu prekrývajú v niektorých prípadoch jeho obsah, ktorý je potom ťažko rozoznať. Podobne je to aj u hrán, niektoré majú rovnakú farbu ako vrchol a sú ťažko rozoznateľné.
Samotný softvér nie je najhorší, aj keď ani najlepší. Obsahuje určité prvky edukačného softvéru, no je dosť neprehľadný a neintuitívny aj pre vysokoškoláka.

Existuje viacero softvérov na grafové algoritmy okrem vyššie spomenutých. No nie každému môžeme dať prívlastok edukačný pre stredné školy. Môže to byť z viacerých dôvodov, no za hlavný dôvod by sme určili slabý záujem o túto tematiku vo vyučovaní informatiky na stredných školách. Ani jeden softvér neobsahuje navrhnutú sadu úloh, či už pre učiteľa alebo žiaka, neobsahuje ani administrátorskú a používateľskú časť, pre doplnenie takýchto úloh pre žiakov učiteľom. Veľká časť programov obsahuje len chabé vizualizácie na grafe. Zo spomenutých softvérov by sme vyzdvihli softvér profesora Davida Gallesa, ktorý má príťažlivé prostredie, pekné a odborné vizualizácie, a hlavne je študentom veľmi nápomocný. Podobnú predstavu máme o našom navrhovanom edukačnom softvéri, v ktorom si ale žiaci dokážu graf vytvoriť, uložiť a môžu pracovať so sadou úloh v pripravenom pracovnom liste.

Kapitola 2

Program

V predchádzajúcej kapitole sme ukázali niekoľko existujúcich programov na prácu s grafmi a grafovými algoritmami. Už ako sme zhodnotili, ani jeden nebol primárne určený pre používanie na strednej škole, všetky boli radené buď skôr k domácemu preopakovaniu alebo pre potreby vysokoškolských študentov. V tejto kapitole predstavujeme nami navrhnutý program, jeho postupný návrh, dizajn, prostredie a ovládanie.

2.1 Softvérové rozhranie

Program na prácu s prehľadávacími algoritmami je implementovaný ako okenná aplikácia v programovacom jazyku JAVA pomocou vývojového prostredia NetBeans IDE s využitím najnovšej softvérovej platformy jazyka Java - JavaFX, ktorá zastrešuje tvorbu komplexnejších internetových aplikácií, ale taktiež aj okenných a mobilných aplikácií či appletov. Platformu JavaFX sme vybrali na základe jednoduchého vytvárania farebných užívateľských prostredí, ale aj jej úplného nahradenia nadstavby starších grafických knižníc, ktoré už nebudú ďalej rozvíjané.
Implementovaný program korektne funguje na počítačoch s operačným systémom Windows s nainštalovaním minimálne Java SE 8 JRE. Pri nižšej verzii nie je možné program používať.

2.2 Dizajn prostredia

Pri navrhnutí dizajnu sme sa snažili z pohľadu používateľa splniť očakávané kritériá. Naším cieľom bolo čo najlepšie pokryť dve podstatné otázky pri hodnotení edukačného softvéru z pohľadu používateľa:

- Má prostredie programu príťažlivý vzhľad?
- Je kvalita farieb, obrázkov a animácií dostatočná?

Farebnosť je jednou zo zásad tvorby edukačného softvéru. Farby sú základnou vlastnosťou obrazovej a súčasťou mimorečovej komunikácie. Pri výbere správnych farieb sme sa snažili dosiahnuť vyzdvihnutie podstatných prvkov do popredia.
Dizajn prostredia je ladený do málo kontrastných farieb, pre vyhnutie sa nechceného rozptyľovania a zbytočnej očnej záťaže pri dlhšom používaní programu. Pri výbere jednotlivých farieb sme brali do úvahy aj ich jednotlivý vplyv na človeka. Hlavné farby programu tvoria primárne farby - červená, zelená a modrá a sekundárna farba - žltá. Žltá farba predstavuje farbu optimizmu, aktivuje a oslobodzuje pred strachom. Pomáha udržovať bdelosť, podporuje koncentráciu a symbolizuje tvorivosť. Taktiež má veľký účinok aj na pozitívnu náladu. Vzhľadom k tomu, že sa práve ku žltej farbe neviažu žiadne negatívne konotácie, je práve žltá farba zvolená na pozadí programu. S kombináciou primárnych farieb využitých pri vytváraní grafu dokonale dopĺňa celý jeho koncept.

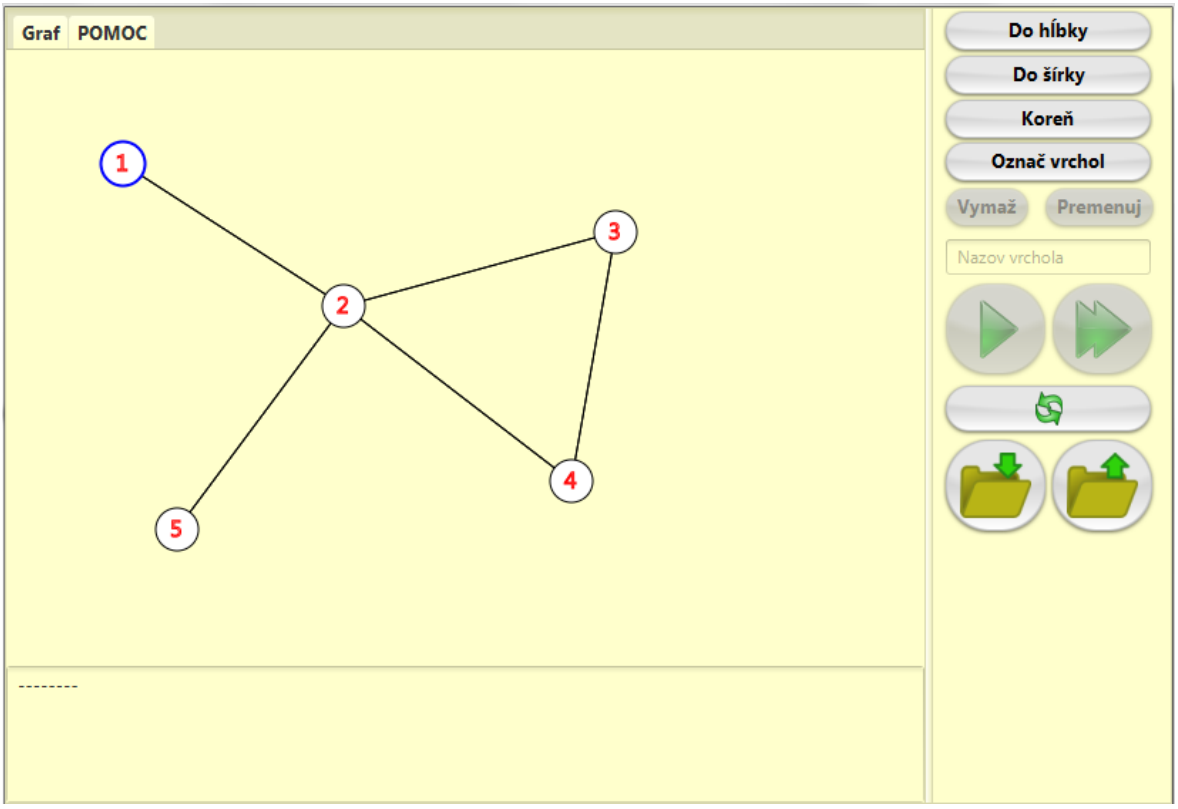
Pri vytváraní grafu využívame dve primárne farby a neutrálnu bielu. Vrchol grafu je kombináciou bielej farby, ktorá vypĺňa jeho vnútro a červenej farby, ktorá zvýrazňuje index vrcholu. Červená farba patrí, podobne ako farba žltá, medzi optimistické farby a zároveň je ľudsky najvýraznejšia. Index vrcholu pri jej použití je pre používateľa výrazný a jasne zreteľný. Vrchol, ktorý je po obvode zvýraznený modrou farbou, signalizuje vrchol, ktorý predstavuje koreň prehľadávania. Modrá farba je oponentná farba k farbe žltej. Rovnako ako červená farba, jej použitie na žltom pozadí dáva používateľovi jasne najavo, kde sa koreň prehľadávania nachádza a aj pri zmene koreňa v grafe nedochádza k žiadnym nejasnostiam.

Počas behu prehľadávania sa jednotlivé aktuálne prehľadávané vrcholy prefarbujú pre lepšiu vizualizáciu a spätnú väzbu používateľovi. Práve pre tento úkon bola zvolená farba zelená, ktorá zvýrazní aktuálne prehľadávaný vrchol po jeho obvode, a taktiež prefarbí index vrcholu obsiahnutý v jeho vnútri. Pozadie vrcholu sme aj počas behu prehľadávania nechali biele.
Jednotlivé tlačidlá obsiahnuté v programe boli do prostredia zvolené tak, aby vystihovali svoju podstatu a ladili so zvoleným dizajnom programu. Na obrázku 2.1 je znázornený dizajn programu s vytvoreným jednoduchým grafom.

2.3 Prostredie programu

Program ihneď po spustení obsahuje okno pre realizáciu grafu s názvom GRAF a druhé okno s názvom POMOC, ktoré obsahuje stručný súhrn algoritmov prehľadávania do šírky a do hĺbky. Medzi jednotlivými oknami sa jednoducho preklikáva. Na pravej strane programu sú umiestnené tlačidlá:

- Do hĺbky - zvolí prehľadávací algoritmus grafu do hĺbky.



Obr. 2.1: Dizajn navrhnutého programu

- Do šírky - zvolí prehľadávací algoritmus grafu do šírky.
- Koreň - po kliknutí na toto tlačidlo je možné zvoliť nový koreň prehľadávania v grafe.
- Označ vrchol - slúži na označenie vrcholu, ktorý sa bude upravovať. Toto tlačidlo je nadradené tlačidlo k tlačidlám Vymaž a Premenuj.
- Vymaž - po označení vrcholu pomocou tohto tlačidla používateľ označený vrchol vymaže.
- Premenuj - po označení vrcholu sa pomocou tohto tlačidla vrchol premenuje na používateľom vložený názov do editu, ktorý sa nachádza pod tlačidlom.
- Jedna zelená šípka - pomocou tohto tlačidla sa vizualizuje prehľadávanie grafu postupne po krokoch.
- Dve zelené šípky - po stlačení tohto tlačidla sa vizualizuje priamo celé prehľadávania grafu.
- Dve zelené šípky idúce do kruhu - vymaže vytvorený graf a vráti program do stavu po spustení.

- Uloženie a načítanie - tieto tlačidlá slúžia na uloženie vytvoreného grafu v programe, alebo na jeho načítanie do programu. Pre možnosť načítania grafu ihneď po spustení programu je toto tlačidlo prístupné ihneď po spustení programu.

Obr. 2.2: Ukážka programu ihneď po spustení.

2.3.1 Vytváranie grafu

Používateľ si vie vytvárať graf do pripraveného okna s názvom GRAF. Vytváranie grafu je veľmi jednoduché. Jednotlivé časti grafu sa vytvárajú nasledovne:

- Vytvorenie vrcholu - kliknutie do okna ľavým tlačidlom myši.
- Vytvorenie hrany - kliknutie pravým tlačidlom myši na vybraný počiatočný a konečný vrchol vytvorí medzi vybranými vrcholmi hranu.

Vytvorené vrcholy grafu sa dajú jednoducho presúvať. Po kliknutí ľavým tlačidlom myši na vrchol a jeho podržaním vie používateľ presunúť vybraný vrchol na požadované miesto prislúchajúceho okna. Pre vymazanie vrcholu musí použiť tlačidlá Označ vrchol a Vymaž. Najprv je potrebné kliknúť na tlačidlo Označ vrchol, ktorý dovolí používateľovi označiť vrchol, ktorý chce upravovať. Ako sme spomenuli vyššie, tlačidlo Označ vrchol je nadradeným tlačidlom pre tlačidlá Vymaž a Premenuj. Po jeho kliknutí sa tieto tlačidlá sprístupnia používateľovi, inak sú zablokované a ich použitie nie

je možné. Po kliknutí na tlačidlo Vymaž označený vrchol vymaže.
Rovnakým spôsobom sa postupuje pri premenovaní vrcholu. Po označení vrcholu si používateľ vyplní požadovaný názov vrcholu do pripraveného editu a následne po stlačení tlačidla Premenuj sa označený vrchol premenuje. Z dôvodu lepšieho porozumenia jednotlivých algoritmov, bola počas testovania programu zavedená menšia zmena. Indexy vrcholov, ktoré vrcholy obsahujú už pri ich samotnom vytváraní, pri ich premenovaní ostávajú. Nový názov vrcholu sa zobrazí vo vrchole vedľa tohto indexu. Takýto prístup premenovania bol zvolený z veľmi podstatného dôvodu. Ak sa vrcholy premenujú na krátke slovné spojenia, pre žiakov by bolo komplikované určiť prioritu vyberania vrcholov v prehľadávaní grafu. Aby sa predišlo nejasnostiam a komplikáciám pri používaní programu vo vyučovaní, algoritmus prehľadávania si bude vždy vyberať vrcholy na základe ich indexov.

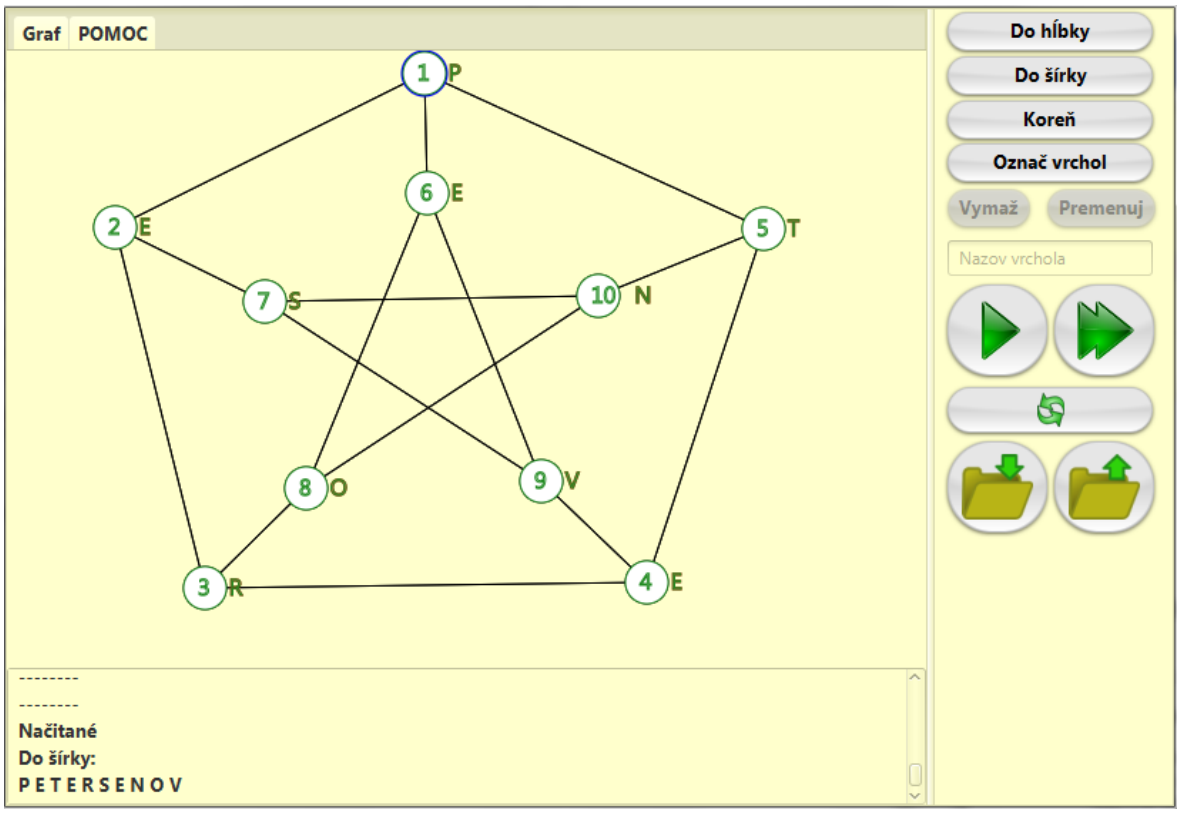
Pri presúvaní vrcholu sa s ním presúvajú aj jemu prislúchajúce hrany. Ak by užívateľ potreboval hranu medzi vrcholmi vymazať, postupuje rovnakým spôsobom ako pri jej vytváraní. Kliknutím pravým tlačidlom na počiatočný a konečný vrchol hrany sa existujúca hrana medzi týmito dvoma vrcholmi vymaže. Iné úkony úpravy pri hranách v programe implementované nie sú.
Vytvorený program si používateľ môže pre jeho opätovné použitie uložiť. Pri jeho načítaní sa graf načíta so všetkými úpravami, ktoré používateľ pri jeho vytváraní na neho aplikoval. Čiže sa načíta v takom stave, v akom ho používateľ uložil.

2.3.2 Vizualizácia algoritmov

Po vytvorení alebo načítaní grafu môžeme prejsť k samotnej vizualizácii prehľadávania. Koreň prehľadávania je primárne nastavený vždy na vrchol s indexom 1. Pre zmenu koreňa si vie užívateľ pomocou tlačidla Koreň vybrať z grafu taký vrchol, z ktorého chce, aby jeho prehľadávanie vychádzalo. Následne si zvolí pomocou tlačidiel algoritmus prehľadávania, ktorý chce na graf aplikovať.
Po zvolení algoritmu je teraz na používateľovi, či chce, aby sa prehľadávanie vizualizovalo až do konca (tlačidlo s dvomi zelenými šípkami), alebo si ho bude postupne púšťať po jednotlivých krokoch (tlačidlo s jednou zelenou šípkou).

Ako vidíme na obrázku 2.3, každá zmena je zaznamená v spodnej časti okna, ktorá predstavuje „textové pole“. Graf v programe na obrázku 2.3 bol načítaný, následne bol vybraný algoritmus prehľadávania do šírky, ktoré bolo programom na grafe aj vizualizované[1]. Postupnosť jednotlivých vrcholov prehľadávania sa postupne počas vizualizácie zapisuje do pripraveného okna. Ako je vidieť, pri premenovaní vrcholov indexy vo vrcholoch grafu ostávajú, no pri výpise postupnosti sa vypisujú len nové názvy vrcholov. Takéto riešenie bolo zvolené pre prehľadnosť postupnosti, ktorá pri kombinácii čísel a

[1]Vizualizáciu môžeme rozpoznať na základe prefarbenia vrcholov na zelené.



Obr. 2.3: Načítaný graf, na ktorom bolo aplikované prehľadávanie do šírky, ktoré bolo programom vizualizované.

písmen by mohla byť chaotická. Používateľ si vie jednotlivé indexy k číslam priradiť na základe vytvoreného grafu v okne. Textové okno sa nevymazáva, počas celého behu programu v ňom ostáva všetko zachované. Pomocou posuvnej lišty sa vie používateľ v nej orientovať a pozrieť si aj predchádzajúce riešenie, resp. zmeny, ktoré vykonal.

Kapitola 3

Testovanie

Edukačný softvér na prehľadávanie grafov bol testovaný na Gymnáziu Jura Hronca v Bratislave. Testovanie prebiehalo na seminári z informatiky, ktorého sa zúčastnilo 11 žiakov 3. a 4 ročníka gymnázia. Testovanie prebehlo v dvoch etapách – dva týždne na dvoch vyučovacích hodinách. Testovanie sa uskutočňovalo na žiakoch jednotlivo a bolo rozdelené na tri etapy:

1. Vysvetlenie grafovej štruktúry a prehľadávacích algoritmov
2. Práca s pracovným listom bez použitia programu
3. Práca s pracovným listom s použitím programu pre overenie a kontrolu

Testovaní žiaci disponovali s vedomosťami o binárnych stromoch, čo im veľmi pomohlo k pochopeniu danej problematiky. Grafová štruktúra bola vysvetlená s použitím úplných základov, ktoré boli priblížené v predchádzajúcej kapitole. Počas prvých dvoch hodín bola žiakom vysvetlená potrebná problematika. V druhej polovici im boli rozdané pracovné listy, ktoré obsahovali zatiaľ len prvé dve z piatich pripravených úloh. Počas druhých dvoch hodín žiaci vypracovali ďalšie tri úlohy z pracovného listu a pokračovali programovaním samotných prehľadávacích algoritmov, čím si utvrdili správne pochopenie nadobudnutých vedomostí.

3.1 Grafová štruktúra a prehľadávacie algoritmy

Na začiatku hodiny sme sa žiakov opýtali, či vedia, čo je to vlastne graf. Reakcie žiakov boli viac menej také, aké sme očakávali. Skoro všetci žiaci reagovali na otázku odpoveďou spojenou s grafom funkcie, grafom v programe Excel a podobne.
Žiakom sme priblížili grafovú štruktúru a následne sme viedli krátku diskusiu o tom, čo je možné pomocou takejto štruktúry riešiť. Počas diskusie sme im odkrývali rôzne vlastnosti grafu, ako orientácia, cesta a podobne. Vlastnosti grafu boli spomenuté len

povrchne, aby žiaci mali vedomosť o ich existenciách. Ako sme spomenuli, žiaci disponovali s vedomosťami o binárnych stromoch, ktoré im rýchlejšie pomohli pochopiť problematiku grafovej štruktúry.
Už počas diskusie sme zaregistrovali nadpriemernú aktivitu dvoch študentov, ktorí potrebné vedomosti už mali nadobudnuté. Zistili sme, že navštevujú korešpondenčné semináre z informatiky občianskeho združenia TROJSTEN, ktoré organizujú z veľkej časti študenti našej fakulty. Využili sme to a s ich pomocou sme na pripravenom grafe na tabuli ostatným žiakom predstavili prehľadávací algoritmus do hĺbky. Žiak postupne prehľadával graf na tabuli pomocou prehľadávania do hĺbky. Spolu s ním sme graficky prechádzali graf, podľa jeho inštrukcií a na tabuľu sme zapisovali postupnosť vrcholov. Po skončení prehľadávania sme algoritmus ešte raz zhrnuli pre ostatných, ktorí tento algoritmus prehľadávania do hĺbky nepoznali. Pre ich lepšie porozumenie sme im ukázali prehľadávací algoritmus aj na binárnom strome.
Po preopakovaní algoritmu sme náhodne vybrali žiaka, ktorý znova prehľadával pripravený graf na tabuli do hĺbky, no s rozdielnym koreňom prehľadávania od predošlého. Pre overenie ním vytvorenej postupnosti vrcholov prehľadávania sme zapli pripravený program a pomocou dataprojektoru sme ho použili pre overenie tejto postupnosti tak, aby to reálne sledovali aj žiaci. Po skontrolovaní postupnosti sme prešli k algoritmu prehľadávania do šírky.
Prehľadávanie do šírky sme žiakom vysvetlili na praktickom príklade rozliatej vody z vedra, ktorá sa roztečie všetkými smermi. Vyzvali sme jedného z dvoch študentov, ktorí tento algoritmus už ovládali, aby pripravený graf na tabuli prehľadal algoritmom prehľadávania do šírky. Následne sme sa opýtali žiakov, či by vedeli pomocou tohto algoritmu nájsť najkratšiu cestu a ako by to vedeli využiť. V pripravenom grafe na tabuli sme vymazali jednotlivé hodnoty z vrcholov, do ktorých žiaci dopĺňali hĺbku vrcholov, na základe ktorej by následne mohli určiť najkratšiu cestu. Pôvodne sme hľadanie najkratšej cesty v grafe nemali v pláne žiakom vysvetľovať, ale na podnety od žiakov sme to zahrnuli do výkladu.

V druhej časti seminára sme žiakom priblížili pripravený program. Rozdali sme im prvú časť pracovných listov, ktorá obsahovala len prvé dve úlohy.

3.2 Pracovný list

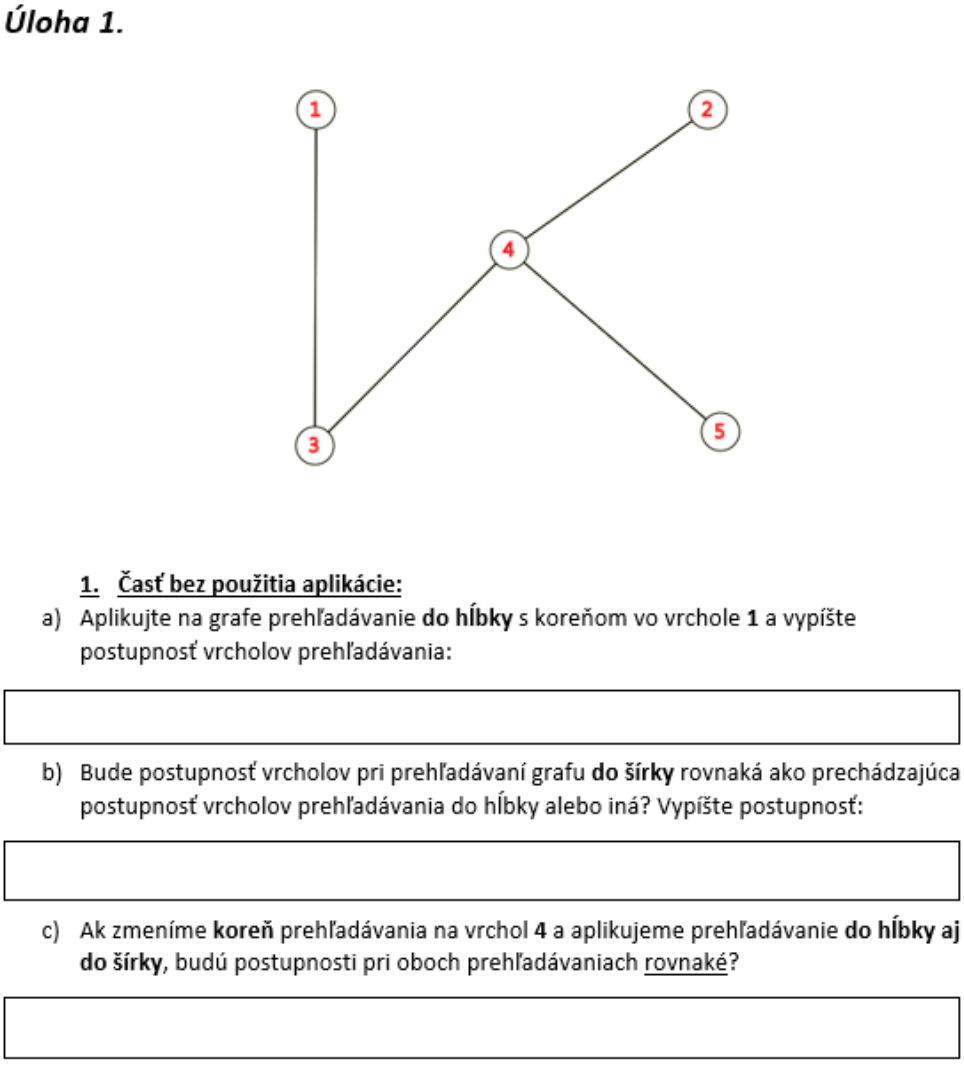
Pripravený pracovný list obsahoval 5 úloh [1], ktoré boli podľa obťažnosti rozdelené na dve časti. V prvej časti výskumu sme žiakom rozdali prvé dve úlohy, ktoré boli obťažnosťou ľahšie, aby sme overili, či rozumejú potrebným algoritmom, ktoré sme im vysvetlili na predchádzajúcej hodine. Ďalšie tri úlohy, ktoré boli náročnejšie, žiaci

[1] Celý obsah pracovného listu je k dispozícii v prílohách na konci tejto práce.

vypracovali na ďalších dvoch hodinách seminára s týždenným časovým odstupom. Každá úloha v pracovnom liste bola rozdelená na dve časti:

- časť bez použitia aplikácie
- časť s použitím aplikácie

Naším cieľom bolo otestovať, či pripravený program žiakom pomôže, dá sa využiť iným spôsobom, alebo je to program, ktorý žiakov neposunie vpred nijakým smerom. Z tohto dôvodu sme každú úlohu, ako vidíme na obrázku 3.1, rozdelili na dve časti, aby sme v prvej časti úloh overili, či žiaci správne pochopili jednotlivé algoritmy a aby si sami v druhej časti úloh overili svoje riešenie. Počas testovania sa ukázalo, že takáto štruktúra pracovného listu bola veľmi nápomocná. Mnoho študentov si bolo svojím riešením istých, no až v druhej časti úlohy s použitím aplikácie objavili chybu.



Obr. 3.1: Ukážka úlohy v pracovnom liste

3.2.1 1. časť testovania

V prvej časti práce na pracovnom liste žiaci pracovali s prvými dvomi úlohami, ktoré boli oproti ostatným jednoduchšie.

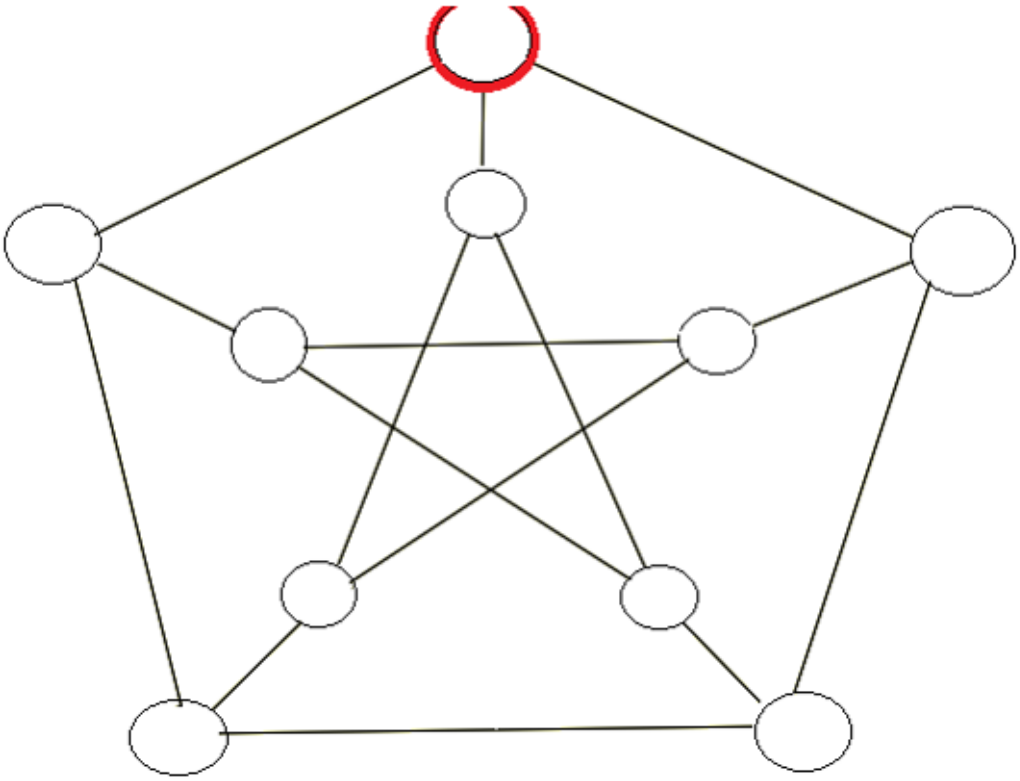
Prvá úloha obsahovala jednoduchý graf s piatimi vrcholmi. Žiaci mali na ňom aplikovať prehľadávanie do hĺbky s koreňom prehľadávania vo vrchole 1. V ďalšom kroku mali zhodnotiť, či bude postupnosť vrcholov grafu v prehľadávaní do šírky totožná s postupnosťou vrcholov prehľadávania do hĺbky alebo rôzna. Následne mali prehľadávanie do šírky na graf aplikovať a vypísať postupnosť. V poslednom kroku sme sa žiakov pýtali, že ak zmeníme koreň prehľadávania na vrchol s číslom 4, či budú postupnosti oboch prehľadávaní vychádzajúce zo zmeneného koreňa totožné alebo rôzne.
Cieľom tejto úlohy bolo zistiť, či žiaci ovládajú jednotlivé algoritmy na takej úrovni, aby dokázali s algoritmami pracovať aj imaginatívne. Ďalším z cieľov bolo overenie, či žiaci chápu potrebnosť koreňa a či dokážu aplikovať algoritmus aj pri jeho zmene.

V druhej úlohe bolo zadanie podobné s úlohou prvou. Úloha obsahovala graf s deviatimi vrcholmi, na ktorom žiaci aplikovali vyžadované algoritmy. Následne sme pôvodný graf zapísali do tvaru binárneho stromu a od žiakov sme vyžadovali odpoveď, či postupnosti vrcholov pri prehľadávaní budú rovnaké ako pri pôvodnom grafe alebo rôzne.
Keďže žiaci ovládali binárne stromy už predtým, túto ich vedomosť sme v tejto úlohe využili. Naším cieľom bolo zistiť, či si žiaci dokážu uvedomiť, že binárny strom môže byť len prekreslený graf, resp. opačne, že graf sa dá jednoducho zmeniť na binárny strom. Obe tieto úlohy si následne overili použitím pripraveného programu. Jednotlivé výsledné postupnosti zapísali do druhej časti úloh v pracovnom liste.

Počas práce na prvej časti pracovného listu sme si všimli, že jednoduché úlohy žiakom nerobili problém. Niektorí dokonca tvrdili, že program vôbec nepotrebujú, že sú si svojím riešením istí. Práca s programom sa im zdala zbytočná. Niektorí program, pre overenie svojich výsledkov, naopak veľmi ocenili. Pomohol im k overeniu ich riešenia a k lepšiemu pochopeniu prehľadávacích algoritmov. Počas vypracovania úloh sme narazili na niekoľko problémov, ktoré by sa bez použitia aplikácie neprejavili. Žiaci si boli istí, že na základe výkladu algoritmom rozumejú, no počas vypracovania úloh a použitia programu zistili, že nerozumejú zmene koreňa prehľadávania. Nevedeli rozdeliť zmenu koreňa a zmenu indexovania vrcholov. Mali zafixované, že koreň prehľadávania musí mať stále hodnotu jeden. Vďaka programu a úlohám sme tieto nejasnosti odstránili, čo sme si potvrdili aj v druhej časti testovania.
Prvú časť testovania, ktorá obsahovala výklad problematiky, ukážku programu a 1. časť pracovného listu sme stihli počas dvoch vyučovacích hodín, čo sme aj predpokladali.

3.2.2 2. časť testovania

Druhá časť testovania prebiehala s týždenným časovým odstupom od prvej časti. V tejto časti žiaci vypracovali ďalšie tri úlohy z pracovného listu, ktoré boli oproti prvým úlohám náročnejšie. Na základe výsledkov z prvej časti testovania sme odhadovali prácu na týchto úlohách na jednu vyučovaciu hodinu.
V tretej úlohe pracovného listu mali žiaci sami navrhnúť graf na pripravených šiestich vrcholoch na základe dvoch postupností v zadaní. Prvá postupnosť predstavovala postupnosť vrcholov pri prehľadávaní do hĺbky s koreňom prehľadávania vo vrchole 1, druhá postupnosť predstavovala postupnosť vrcholov pri prehľadávaní do šírky s koreňom prehľadávania vo vrchole 5. Nimi vytvorený graf mal spĺňať obe postupnosti zo zadania. Cieľom tejto úlohy bolo testovanie opačného prístupu oproti predchádzajúcim úlohám, s ktorými nemali problém. Zistili sme, že takýto typ úloh bol náročný pre všetkých žiakov a robil im veľké problémy. Pri overovaní svojho riešenia v programe každý jeden zistil, že navrhnutý graf nemal správne. Už pri tejto úlohe žiaci ocenili program, ktorý im nakoniec pomohol nájsť správne riešenie a vďaka nemu pochopili chybovosť svojho predchádzajúceho riešenia.



Obr. 3.2: Ukážka pripraveného grafu pre žiakov v úlohe 5 v pracovnom liste

Ďalšia úloha mala podobný charakter ako predchádzajúca s rozdielom, že žiaci nemali dopĺňať hrany, ale mali vypĺňať obsah vrcholov v pripravenom grafe, viď obrázok 3.2. Vrcholy mali vyplniť tak, aby postupnosť vrcholov pri aplikovaní prehľadávania do šírky, s koreňom vo vyznačenom vrchole, vytvorila názov pripraveného grafu, a to Petersenov. Ako sme spomenuli v predchádzajúcej kapitole o programe, pri premenovávaní vrcholov sa vo vrcholoch zachováva jeho index, pre jasnosť prioritného výberu vrcholov algoritmom. Z tohto dôvodu mali žiaci vyplniť v pracovnom liste vrchol tak, by obsahoval index aj prislúchajúce písmeno.
Pri tejto úlohe sme narazili na ďalší problém. Niektorí žiaci si neuvedomovali dôležitosť hrany medzi jednotlivými vrcholmi. Všimli sme si, že pri overovaní riešenie v programe vytvorili vrcholy na prislúchajúcich miestach a vôbec nevytvárali hrany. V takomto prípade im samozrejme program potvrdil ich riešenie, pretože sa algoritmus spúšťal pre každý vrchol samostatne. Po vyzvaní, aby v programe vytvorili aj hrany, zistili, že ich riešenie nie je správne. Pri tomto type úloh žiaci nemali až taký veľký problém ako v prechádzajúcom type úlohy.
Posledná úloha v pracovnom liste obsahovala graf, ktorý predstavoval cestnú sieť medzi mestami. Žiaci v zadaní dostali tri trasy: odkiaľ - kam sa chceme dostať. Vychádzajúce mesto predstavovalo koreň prehľadávania, konečné mesto predstavovalo posledný navštívený vrchol v postupnosti prehľadávania. Postupnosť miest mala obsahovať všetky mestá z pripraveného grafu. Úlohou žiakov bolo k jednotlivým trasám zvoliť vhodné prehľadávanie, aby sa vďaka nemu dostali do cieľového mesta. Taktiež mali v grafe k jednotlivým mestám priradiť správne indexy vrcholov tak, aby postupnosť jednotlivých miest bola korektná aj podľa abecedy. Všetky tri trasy mali byť aplikované na jednom a tom istom grafe s indexmi, žiaci nemali pri každej trase graf preindexovávať.
Táto úloha robila žiakom najväčšie problémy. Väčšina žiakov nedokázala vyriešiť úlohu bez použitia aplikácie, resp. s pomocou spolužiakov, ktorí jeho čiastkové riešenia kontrolovali v aplikácii. Väčšina žiakov zhodnotila, že bez pripraveného programu by úlohu nevedeli vyriešiť.
Keďže sme predpokladali, že vypracovanie úloh im zaberie jednu vyučovaciu hodinu, na druhej vyučovacej hodine mali žiaci prehľadávacie algoritmy programovať. Ako sa ukázalo, žiaci úlohy veľmi nezvládali, boli pre nich veľmi náročné. Ich vypracovanie zabralo cca 70 min, čiže jednu a pol vyučovacej hodiny, čím sme programovanie algoritmov museli presunúť.

Po vypracovaní celého pracovného listu si žiaci naozaj uvedomili veľkú pomoc, ktorú mali v pripravenom programe. Dokonca aj dva žiaci, ktorí danú problematiku ovládali, ocenili, že program im pomohol k rýchlejšiemu nájdeniu správneho riešenia a odhalenia chyby v ich riešení. Na záver testovania žiaci vyplnili dotazník, aby sme zistili ich spätnú väzbu k vzhľadu programu, jeho ovládaniu, resp. čo by na ňom oni sami vylepšili, aby sa im v ňom pracovalo lepšie, efektívnejšie.

3.3 Dotazník

Dotazník obsahoval celkovo osem otázok. Na päť otázok mali odpovedať pomocou pripravenej škály, ďalšie tri slúžili na ich postrehy a nápady, ktorými by sa mohol program vylepšiť. Dotazník vyplnilo desať študentov, z toho štyria študenti vyplnili všetkých osem otázok. Šiesti študenti sa vyjadrili k programu len pomocou otázok, na ktoré sa odpovedalo pomocou pripravenej škály.

1. otázka: *Ako hodnotíte ovládanie aplikácie?* (Bolo to veľmi jednoduché a intuitívne (**1**) – Bolo to príliš zložité (**5**))

   - hodnota 1 - 3
   - hodnota 2 - 6
   - hodnota 3 - 1
   - hodnota 4 - 0
   - hodnota 5 - 0

2. otázka: *Ako hodnotíte prostredie aplikácie?* (Veľmi sa mi páči (**1**) – Vôbec sa mi nepáči (**5**))

   - hodnota 1 - 7
   - hodnota 2 - 2
   - hodnota 3 - 1
   - hodnota 4 - 0
   - hodnota 5 - 0

3. otázka:*Pomohla vám aplikácia k lepšiemu pochopeniu algoritmov?* (Určite áno (**1**) – Určite nie (**5**))

   - hodnota 1 - 7
   - hodnota 2 - 0
   - hodnota 3 - 2
   - hodnota 4 - 0
   - hodnota 5 - 1

4. otázka: *Používali by ste aplikáciu na overenie svojich vedomostí?* (Určite áno (**1**) – Určite nie (**5**))

   - hodnota 1 - 8
   - hodnota 2 - 1

- hodnota 3 - 1
- hodnota 4 - 0
- hodnota 5 - 0

5. otázka: *Ocenili by ste používanie aplikácie už pri vysvetľovaní algoritmov?* (Áno, bolo by to lepšie (**1**) – Nie, príde mi zbytočná (**5**))

   - hodnota 1 - 8
   - hodnota 2 - 0
   - hodnota 3 - 2
   - hodnota 4 - 0
   - hodnota 5 - 0

Z ďalších našich postrehov od žiakov sme zaznamenali, že žiaci by ocenili pri vytváraní hrany farebné označenie začiatočného vrcholu, aby dostali spätnú väzbu od programu o tom, že začiatočný vrchol už majú vybraný. Keďže pri vytváraní hrany stačí kliknúť na vrcholy len pomocou pravého tlačidla myši, žiaci mali niekedy problém s tým, že si neboli istí, či už na niektorý vrchol klikli, a tak vytvárali hrany aj tam, kam pôvodne nechceli. Pri vytváraní zložitejších grafov im to vytváralo komplikácie.
Žiaci by upravili aj systém premenovania vrcholov, ktorý im príde veľmi komplikovaný. Tento problém sme predpokladali už pri samotnej implementácii, no doposiaľ sme nenašli vhodnejší spôsob, ktorý by bol pre žiakov dostatočne intuitívny. Žiaci navrhovali jednoduché kliknutie na vrchol a premenovanie pomocou klávesnice. Keďže sa pomocou kliknutia ľavého tlačidla vrchol vytvára a jeho podržaním je možné vrchol presúvať, doplnenie jeho funkčnosti o premenovanie by už jeho korektné používanie mohlo skomplikovať. Taktiež navrhovali zmenu indexovania jednotlivých vrcholov, ktorá v programe momentálne nie je možná.

3.4 Vyhodnotenie testovania

Testovanie dopadlo nad naše očakávania. Až na pár detailov boli žiaci s programom spokojní, po vizuálnej stránke sa im program páčil, ovládanie im prišlo jednoduché, a hlavne sa im v ňom dobre pracovalo. Pri vypracovávaní pracovného listu im bol veľmi nápomocný, pomohol im odhaliť prípadné chyby v ich riešeniach. Ako zhodnotili aj v dotazníku, väčšine testovaných žiakov používanie programu pomohlo k lepšiemu pochopeniu algoritmov, používali by ho na overovanie svojich vedomostí a ocenili by jeho využitie už pri samotnom vysvetľovaní algoritmov. Rovnako im vyhovovala aj rozdelená práca na jednotlivých úlohách v pracovnom liste. Ak by boli úlohy pripravené len

v programe, pracovalo by sa im síce jednoduchšie, ale nezapájali by tak svoje myslenie. Práca s úlohou najprv na papier im umožnila tvorivo myslieť a následne si svoje riešenie overiť. Takýto štýl vypracovania a testovania úloh dostal žiakov na jednu úroveň. Žiaci, ktorí sa necítili byť v skupine žiakov úspešní, si po overení svojho správneho riešenia programom pozdvihli svoje sebavedomie v preberanej téme. Stali sa aj opačné prípady, kedy veľmi šikovní žiaci si svojím riešením boli natoľko istí, že až pomocou programu zistili jeho chybovosť.
Celkovo môžeme zhodnotiť, že aj keď daná problematika nie je štandardnou témou vo vyučovaní informatiky, využitie programu by dopomohlo jej sprístupneniu pre vyučovanie učiteľom, čím je veľká pravdepodobnosť sprístupnenia problematiky aj samotným žiakom.

3.5 Budúca práca

V budúcej práci by sme chceli doplniť program o návrhy, ktoré boli podané žiakmi a taktiež aj o návrhy, ktoré sme dostali od niektorých stredoškolských učiteľov. Samotný dizajn aplikácie je v celku vyhovujúci. Je potrebné upraviť dizajn grafu o výraznejšie farby počas vizualizácie prehľadávacieho algoritmu, aby bola samotná vizualizácia prehľadnejšia.
Po zrealizovanom výskume a nadobudnutých poznatkoch by sme chceli doplniť program o algoritmus hľadania najkratšej cesty na grafe. Taktiež chceme doplniť program o počítanie jednotlivých komponentov grafu, ktoré by boli aj farebne odlišné. Z existujúcej funkčnosti programu plánujeme odstrániť vymazávanie jednotlivých vrcholov. Počas testovania bola táto funkcia úplne nevyužitá. Z praktického hľadiska žiaci využívali úplný reštart aplikácie alebo intuitívnejšie presúvanie vrcholov v priestore okna. Odstránením funkcie vymazávania vrchola by sme mohli dospieť k zjednodušeniu premenovávania vrcholov, ktoré bolo aj zo strany žiakov, ako užívateľa, žiadané.
V samotnej príprave hodiny by sme do budúcnosti chceli program využiť ako nástroj pre učenie objavovaním. Žiakom by prehľadávacie algoritmy neboli vysvetlené, bola by im ponúknutá vytvorená aplikácia, pomocou ktorej by mali dospieť k pochopeniu ich princípu. Keďže sme počas výskumu zaznamenali u žiakov veľké využívanie okna so stručným zhrnutím týchto algoritmov, vo vyučovacom modeli učenia s objavovaním by toto okno nebolo žiakom sprístupnené.

Záver

V záverečnej práci sme sa zaoberali grafovými algoritmami vo vyučovaní informatiky na strednej škole, presnejšie algoritmami na prehľadávanie grafových reprezentácií. Implementovali sme program, ktorý je možné využiť už pri samotnom vysvetľovaní alebo pri overovaní správnych žiackych riešení jednotlivých úloh.

V prvej časti práce, ktorá predstavuje východiská práce, sme zhrnuli štátny vzdelávací program, Bloomovu taxonómiu a v jednoduchosti opísané potrebné grafové znalosti pre prácu s implementovaným programom.

V ďalšej kapitole sme opísali implementovaný program na základe vlastností dizajnu, jeho prostredia a samotného ovládania programu pre správne použitie vo vyučovacom procese. Dôležitú časť kapitoly tvorí softvérové rozhranie, ktorého požiadavky je potrebné dodržať. V inom prípade môže dôjsť k úplnému znemožneniu použitia programu.

V poslednej kapitole, ktorá je venovaná testovaniu programu na strednej škole, sme testovali potrebnosť využitia daného typu programu. Na základe testovania sme zistili, že žiaci na strednej škole prácu s pripraveným programom ocenili. V prichystanom pracovnom liste vypracovali päť úloh, ktorých riešenie si následne mali overiť v pripravenom programe. Mnoho študentov využívalo program už pri samotnom riešení úloh, hlavne pri úlohách náročnejšieho typu.

Na záver študenti ocenili praktickú pomôcku vo forme programu, ktorý v budúcnosti odporúčali využiť už pri samotnom vysvetľovaní prehľadávacích algoritmov. Mnoho študentov zhodnotilo, že vytvorená aplikácia im obohatila vedomosti získané z výkladu učiteľa o praktické poznatky, ktoré im pomohli k odstráneniu nejasností a k lepšiemu pochopeniu samotných algoritmov.

V samotnom závere môžeme zhodnotiť, že záverečná práca splnila požadovaný cieľ.

Literatúra

[1] Jack Conklin, Lorin W Anderson, David Krathwohl, Peter Airasian, Kathleen A Cruikshank, Richard E Mayer, Paul Pintrich, James Raths, and Merlin C Wittrock. A taxonomy for learning, teaching, and assessing: A revision of bloom's taxonomy of educational objectives complete edition, 2005.

[2] Stéphane Crozat, Philippe Trigano, and Olivier Hû. Empi: A questionnaire based method for the evaluation of multimedia interactive pedagogical software. In *PDPTA'99*, 1999.

[3] Shimon Even. *Graph algorithms*. Cambridge University Press, 2011.

[4] David Galles. Data structure visualizations, 2011.

[5] FMFI UK Katedra základov vyučovania informatiky. Informatická súťaž ibobor.sk, 2008.

[6] Chokchai Box Leangsuksun. Lafore's non-directed non-weighted graph graph, 2012.

[7] Janka MAJHEROVÁ. Revidovaná bloomova taxonómia a kompetencie pre používanie ikt. *Interdisciplinárny dialóg odborovỳch didaktík*.

[8] Jarmila Skalková. *Obecná didaktika-2., rozšířené a aktualizované vydání*. Grada Publishing as, 2007.

[9] Robert E Slavin and Nicola Davis. *Educational psychology: Theory and practice*. Allyn and Bacon Needham Heights, MA, 1997.

[10] El Tramo. Jsearchdemo, 2013.

[11] Jakub Černý. *Základní grafové algoritmy*. Matematicko - fyzikální fakulta, Univerzita Karlova v Praze, 2010.

[12] Štátny pedagogický ústav. *Štátny vzdelávací program, Informatika - príloha isced 3*. Bratislava, 2008.

[13] Štátny pedagogický ústav. *Inovovaný štátny vzdelávací program, Informatika - gymnázium so štvorročným alebo päťročným vzdelávacím programom*. Bratislava, 2015.

Prílohy

Príloha č.1 : Pracovný list