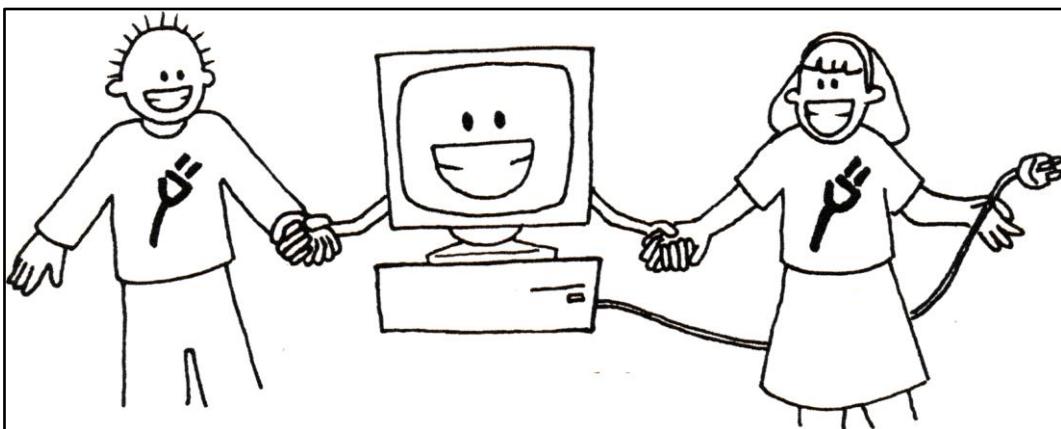
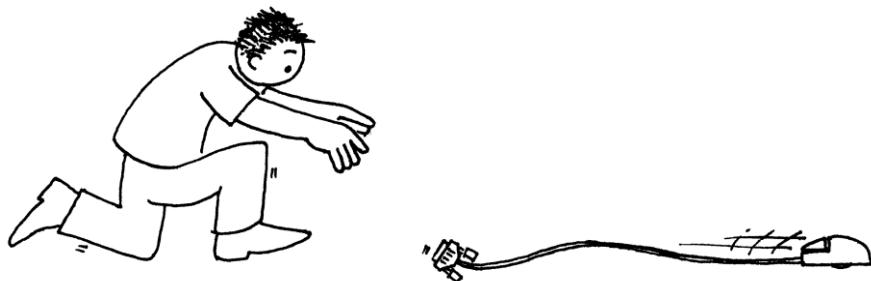


CS UNPLUGGED



**Obogaćeni i dodatni program
za učenike osnovnih škola**



Created by

Tim Bell, Ian H. Witten and Mike Fellows

Adapted for classroom use by Robyn Adams and Jane McKenzie

Illustrations by Matt Powell

2015 Revision by Sam Jarman

Uvod

Računari su svuda oko nas. Svi trebamo naučiti kako da ih koristimo, a već ih mnogi od nas koriste svaki dan. Ali kako zapravo računari rade ? Na koji način razmišljaju ? I kako ljudi mogu napisati software koji je brz i efikasan i lagan za korištenje. Računarske nauke (Computer science) je zadivljuće lijepa naučna grana koja se bavi izučavanjem ovih pitanja. Lagane i zabavne aktivnosti predstavljene u ovoj knjizi su namijenjene učenicima svih uzrasta. Osnovni cilj svih ovih aktivnosti je da predstavi osnovne pojmove i koncepte koji opisuju kako jedan računar radi. I sve to bez korištenja računara.

Knjigu je moguće koristiti kao dadatnu literaturu za obogaćivanje osnovnog programa u školama ali i kao osnovni udžbenik u redovnoj nastavi. Nastavnik ne mora biti ekspert za računare ili za programiranje da predaje i koristi (i usput uči) predstavljene osnove zajedno sa svojim učenicima. Knjiga sadrži niz aktivnosti koje su predstavljene zajedno sa teoretskim osnovama na kojima su zasnovane i sve je to objašnjeno na jednostavan način. Odgovori na sva pitanja su dati, i svaka aktivnost završava sa dijelom ‘O čemu su zapravo ovdje radi?’ koji detaljno objašnjava važnost i mjesto za računarske nauke ovih aktivnosti.

Mnogo od predstavljenih aktivnosti su zasnovane na matematici, na primjer predstavljanje binarnih brojeva, pridruživanja i grafovi, problemi sa uzorcima i sortiranje, kao i kriptografija. Drugi imaju posebnu vezu sa tehnologijama i tehničkom kulturom, dakle znanjem i razumjevanjem kako računari rade. Učenici su aktivno uključeni u zajednički rad, rješavanje problema, kreativnost, i razvoj svojih konjunktivnih sposobnosti u dobro dizajniranom okruženju. Sve aktivnosti takođe omogućavaju aktivan način upoznavanja i istraživanja kompjutacionog razmišljanja (“computational thinking”) koji sve više dobiva svoje mjesto u programu škola.

Pored same knjige, “Unplugged” projekat ima puno dodatnih online sadržaja i aktivnosti koje uključuju video, slike kao i dodatne materijale koje sve možete naći na csunplugged.org. Uporedo sa edicijom knjige za 2015 godinu napravljen je i novi website, sa više materijala, boljim i lakšim pristupom open source sadržajima, kao i jasnim i čvršćim vezama sa aktuelnim planom i programom za računarske nauke i kompjutaciono razmišljanje koje se pojavljuje u školama.

Ova knjiga je napisana od strane tri predavača računarskih nauka i dva školska učitelja i zasnovana je na našim iskustvima iz učionica kao i povratnim informacijama i sugestijama od stotina nastavnika tokom više od dvije decenije. Otkrili smo da se mnogi važni koncepti mogu predstaviti i naučiti bez korištenja računara – u stvari, računari su često samo odvrćali pažnju učenika od učenja i razumjevanja. Vrlo često, računarske nauke se uče (i predaju) prvo kroz programiranje, ali ne nisu svi studenti dobro motivisani za takav pristup, i to može biti osnovna prepreka da bi došli do zaista interesantnih ideja koje su u osnovi računarskih nauka. Prijedlog je dakle da isključimo (unplug) računar, i da se spremimo za učenje šta su to stvarno računarske nauke !

**This book is available as a free download thanks to a generous grant by Google, Inc.
It is distributed under a Creative Commons Attribution-NonCommercial-
ShareAlike licence, which means that you are free to share (copy, distribute, and
transmit) the book. It also allows you to remix the book. These are only available
under the following conditions: you include attribution to the authors, you do not
use this book for commercial purposes, and if you alter, transform or build upon**

this work, you share under the same or similar license. More details of this license can be found online by searching: CC BY-NC-SA 3.0.

We encourage the use of this material in educational settings, and you are welcome to print your own copy of the book and distribute worksheets from it to students. We welcome enquiries and suggestions, which should be directed to the authors (see csunplugged.org).

This book has been translated into many languages. Please check the web site for information about the availability of translations.

Zahvalnost

Mnogo djece i nastavnika je pomoglo da se ideje u ovoj knjizi predstave na ovakav način. Učenici i nastavnici u South Park School (Victoria, BC), Shirley Primary School, Ilam Primary School kao i Westburn Primary School (Christchurch, New Zealand) su bili prvi zamorčići za mnoge od ovih aktivnosti. Naročitu zahvalnost dugujemo Linda Picciotto, Karen Able, Bryon Porteous, Paul Cathro, Tracy Harrold, Simone Tanoa, Lorraine Woodfield, i Lynn Atkinson što su nas primili u svoje razrede i dali sugestije i prijedloge kako unaprijediti aktivnosti. Gwenda Bensemann je isprobala nekoliko aktivnosti i dala sugestiju kako ih unaprijediti. Richard Lynders i Sumant Murugesh su pomogli sa probama u učionicama. Neke aktivnosti iz kriptografije su razvijene od strane kolege Ken Noblitz. Neke aktivnosti su izvodjene u okviru Victoria "Mathmania" grupe, i pod nadzorom i uz pomoć Kathy Beveridge. Rane verzije ilustracija su uradjene od strane Malcolm Robinson i Gail Williams, a koristili smo pomoć i savjete od Hans Knutson. Matt Powell je takodje vrijedno pomagao tokom razvoja "Unplugged" projekta. Zahvalni smo Brian Mason Scientific i Technical Trust za nesebičnu i velikodušnu pomoć i sponzorstvo u ranoj fazi razvoja knjige.

Naročita zahvalnost ide prema Paul i Ruth Ellen Howard, koji su isprobali mnoge od predstavljenih aktivnosti i dali brojne korisne sugestije. Peter Henderson, Bruce McKenzie, Joan Mitchell, Nancy Walker-Mitchell, Gwen Stark, Tony Smith, Tim A. H. Bell¹, Mike Hallett, i Harold Thimbleby su takodje dali niz korisnih savjeta.

Mi svi dugujemo naročitu zahvalnost našim porodicma: Bruce, Fran, Grant, Judith, i Pam za njihovu podršku, kao i Andrew, Anna, Hannah, Max, Michael, i Nikki koji su bili inspiracija za dobar dio uradjenog posla,² i koji su često bili prva djeca koja su probavala neku od aktivnosti.

Naročito smo zahvalni Google Inc. Za sponzorstvo i podršku Unplugged projektu, i što su nam omogućili da ova edicija bude dostupna za slobodno preuzimanje putem interneta.

Svi komentari i sugestije o aktivnostima su dobrodošle. Autori mogu biti kontaktirani preko csunplugged.org.

1

Nema veze sa prvim autorom.

2

U stvari, aktivnost za kompresiju teksta je predložena od strane Michael.

Contents

Introduction	Error! Bookmark not defined.
Acknowledgements	iii
Data: the raw material—<i>Representing information</i>	1
Count the Dots— <i>Binary Numbers</i>	3
Colour by Numbers— <i>Image Representation</i>	16
You Can Say That Again! — <i>Text Compression</i>	26
Card Flip Magic— <i>Error Detection & Correction</i>	34
Twenty Guesses— <i>Information Theory</i>	42
Putting Computers to Work—<i>Algorithms</i>	49
Battleships— <i>Searching Algorithms</i>	51
Lightest and Heaviest— <i>Sorting Algorithms</i>	70
Beat the Clock— <i>Sorting Networks</i>	77
The Muddy City— <i>Minimal Spanning Trees</i>	84
The Orange Game— <i>Routing and Deadlock in Networks</i>	90
Tablets of Stone— <i>Network Communication Protocols</i>	94
Telling Computers What To Do—<i>Representing Procedures</i>	101
Treasure Hunt— <i>Finite-State Automata</i>	103
Marching Orders— <i>Programming Languages</i>	119
Really hard problems—<i>Intractability</i>	125
The poor cartographer—Graph coloring	128
Tourist town— <i>Dominating sets</i>	142
Ice roads — <i>Steiner trees</i>	151
Sharing secrets and fighting crime—<i>Cryptography</i>	162
Sharing secrets— <i>Information hiding protocols</i>	Error! Bookmark not defined.
The Peruvian coin flip— <i>Cryptographic protocols</i>	Error! Bookmark not defined.
Kid Krypto— <i>Public-key encryption</i>	Error! Bookmark not defined.

The human face of computing— <i>Interacting with computers</i>	Error! Bookmark not defined.
The chocolate factory— <i>Human interface design</i>	Error! Bookmark not defined.
Conversations with computers— <i>The Turing test</i>	Error! Bookmark not defined.

DIO I

**Podaci: sirovi materijal—
Predstavljanje informacija**

Podaci: Sirovi materijal

Kako možemo čuvati informacije u računaru?

Riječ “computer” dolazi od latinske riječi *computare*, koja znači računati ili dodavati, ali su računari danas mnogo više od ogromnih digitrona. Računari mogu biti biblioteka, mogu nam pomoći pri pisanju, mogu pronašlaziti informacije za nas, mogu nam reproducirati muziku ili čak prikazivati filmove. Pa kako onda računari čuvaju sve te informacije? Vjerovali ili ne, računari koriste samo dvije stvari za to: nulu i jedan!

Koja je razlika izmedju podataka i informacija?

Podaci su sirovi materijal, brojevi sa kojima, i pomoću kojih, računari rade. Računar poslije pretvara te (sirove) podatke u informacije (rijecu, brojeve i slike) koje onda vi i ja možemo razumjeti.

Kako se onda brojevi, slova, riječi i slike mogu pretvoriti u nule i jedinice?

U ovom dijelu ćemo učiti o binarnim brojevima, kako računari crtaju slike, kako faks mašina funkcioniše, koji je najefikasniji način da čuvamo velike količine podataka, kako možemo izbjegći pojavljivanje grešaka i kako možemo izmjeriti količinu informacija koje želimo sačuvati.



Aktivnost 1

Prebrojavanje tačaka—*Binarni brojevi*

Sažetak

Podaci u računarima se čuvaju i prenose kao nizovi nula i jedinica. Kako možemo predstaviti riječi i brojeve koristeći samo ova dva simbola?

Veza sa Curriculum-om

- ✓ Matematika: Brojevi – Izučavanje brojeva po drugim bazama. Predstavljanje brojeva u zapisu po bazi dva.
- ✓ Matematika: Algebra – Nastaviti dati sekvencijalni (nizovni) uzorak, i opisati riječima pravilo za taj uzorak. Uzorci i veze za stepena broja dva.

Vještine

- ✓ Prebrojavanje
- ✓ Uparivanje
- ✓ Sekvenciranje

Dobna/Starosna grupa

- ✓ 6 godina i stariji

Materijal

- ✓ Trebaćete napraviti skup od pet binarnih karata (pogledati na stranici 7) za demonstraciju aktivnosti.
A4 stranica sa smiley naljepnicama može dobro poslužiti.

Svaki učenik treba da ima:

- ✓ Skup od pet karata.
Copy Photocopy Master: Binarni brojevi (stranica 7) na kartonu, i izrezani.
- ✓ List za Aktivnost: Binarni brojevi (stranica 6)

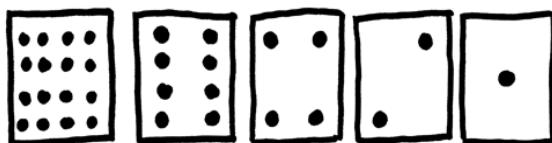
Postoje i dodatne aktivnosti za koje će svaki učenik trebati:

- ✓ List za Aktivnost: Rad sa binarnim brojevima (stranica 8)
- ✓ List za Aktivnost: Slanje tajnih poruka (stranica 9)
- ✓ List za Aktivnost: Email i modemi (stranica 10)
- ✓ List za Aktivnost: Brojanje brojeva većih od 31 (stranica 11)
- ✓ List za Aktivnost: Više o binarnim brojevima (stranica 12)

Uvod

Prije nego što date zadaćnicu sa strane 7 biće korisno da pokažete cijeloj grupi učenika osnovne principe aktivnosti.

Za ovu aktivnost će vam trebati pet karata, kao što je prikazano ovdje dolje, sa tačkicama sa jedne strane i potpuno praznom drugom stranom. Izaberite pet učenika koji će držati karte za pokazivanje i koji će stajati ispred cijelog razreda. Karte treba da budu u sljedećem redoslijedu:



Diskusija

Dok dijelite karte (počevši od onih sa desna na lijevo), provjerite da li učenici mogu pogoditi koliko tačkica ima na sljedećoj karti. Šta ste primjetili vezano za broj tačkica na kartama? (Svaka karta ima dva puta više tačkica nego karta sa njene desne strane.)

Koliko tačkica će imati sljedeća karta ukoliko nastavimo niz prema lijevo? (32) A sljedeća...? (64)

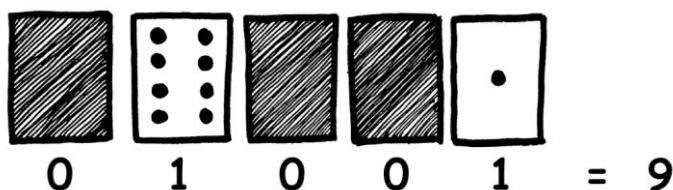
Možemo koristiti ove karte da predstavimo brojeve tako što ćemo neke okrenuti licem prema dolje (sakrivajući tačkice) i onda sabirati tačkice koje vidimo. Zatražite od učenika da pokažu 6 tačkica (karta sa 4 tačkice i karta sa 2 tačkice), pa onda 15 (8-, 4-, 2- i 1-tačkica karta), pa onda 21 (16, 4 i 1)... Jedino pravilo je da sadržaj svake karte mora biti ili potpuno vidljiv ili potpuno sakriven.

Koji je najmanji broj mogućih tačaka? (Moguće je da ćete dobiti odgovor jedna tačkica ali pravi odgovor je nula).

Sada probajte brojati počevši od nule koristeći karte.

Cijeli preostali razred treba pažljivo posmatrati kako se karte mijenjaju kako bi pokušali razumjeti način na koji se karte okreću (svaka karta se okreće upola manje puta nego njena susjedna karta na desnoj strani). Možete probati isti zadatak sa nekoliko grupa uzastopno.

Kada karta sa binarnim brojem **nije** pokazanaWhen a binary number, onda je ona predstavljena nulom. Ovo je binarni brojni sistem.



Zatražite od učenika da naprave broj 01001. Koji je to broj u decimalnom brojnom sistemu? (9) Kako bi predstavili 17 u binarnom brojnom sistemu? (10001)

Pokušajte sa nekolik drugih brojeva prije nego se uvjerite da su učenici razumjeli cijeli koncept.

Slijedi pet dodatnih aktivnosti koje služe da se utvrди stečeno znanje. Učenici treba da urade što je moguće više ovih aktivnosti (u zavisnosti koliko vremena vam preostane).

Radni List za Aktivnost: Binarni Brojevi

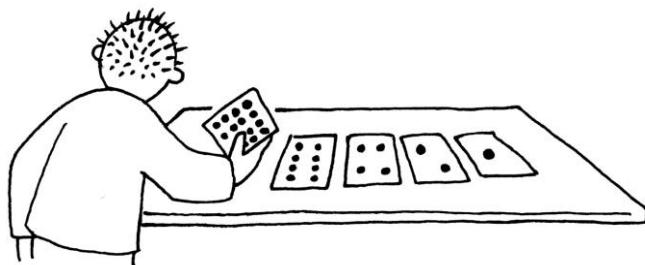
Naučiti kako računati

Znači vi mislite da znate kako računati? Dobro, ali evo ovdje jedan drugi, novi način računanja!

Da li ste znali da računari koriste samo nule i jedinice? Sve što vidite ili čujete na nekom računaru – riječi, slike, brojevi, filmovi pa čak i zvukovi su sačuvani u računaru koristeći samo ova dva znaka, nula i jedan! Ove aktivnosti će vas naučiti kako poslati tajne poruke vašim prijateljima koristeći tačno isti metod koji koriste i računari.

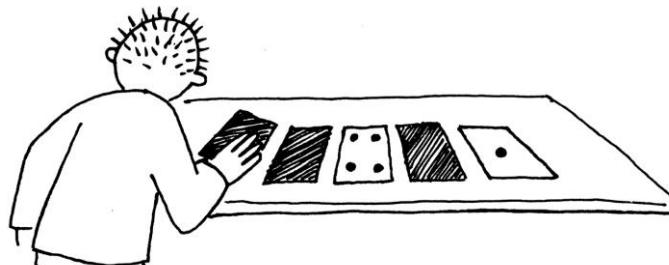
Uputstva

Napravite i isjecite karte od papira i posložite ih na sto tako da je karta sa 16 tačaka sa lijeve strane (kao što je prikazano ovdje na slici):



Uvjerite se da su karte poredane upravo kao na slici, u istom redoslijedu.

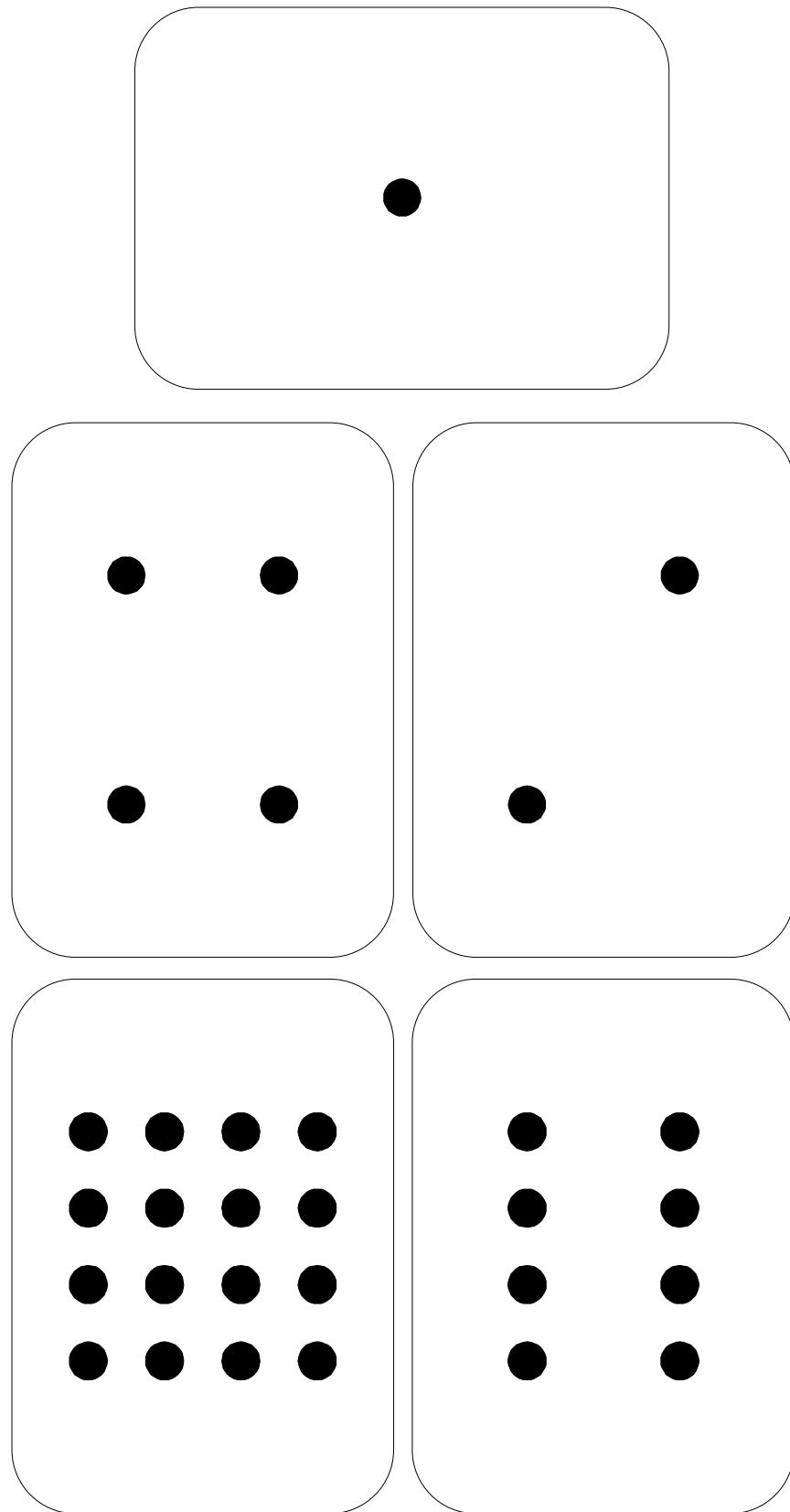
Sada okrenite karte tako da vidimo ukupno 5 tačaka kao što pokazuje slika—i pri tome pažljivo čuvajte isti redoslijed karata!



Sada razmislite kako bi dobili 3, 12, 19. Da li postoji više od jednog načina da se dobije neki od brojeva? Koji je najveći broj koji možete dobiti pomoću ovih karata? Koji je najmanji broj? Da li postoji neki broj izmedju najmanjeg i najvećeg broja koji ne možete prikazati pomoću karata sa tačkicama?

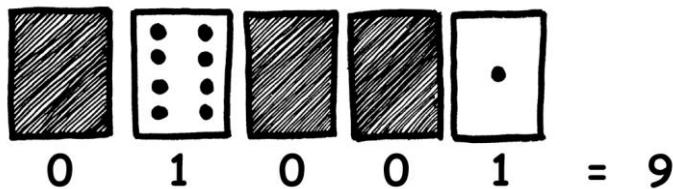
Ekstra za Eksperte: Pokušajte dobiti brojeve 1, 2, 3, 4 u tom redoslijedu. Možete li naći neku logičnu i jasnu metodu za okretanje karata tako da broj veći za jedan?

Uzorak za fotokopiranje: Binarni Brojevi



Radni List za Aktivnost: Rad sa Binarnim brojevima

Binarni sistem koristi **nule** i **jedinice** kakobi predstavio da li je neka karta okrenuto licem prema gore ili ne. **0** označava da je karta sakrivena, a **1** označava da možete vidjeti njene tačke. Na primjer:



Možete li reći koliko je **10101**? A šta je sa **11111**?

Koji dan u mjesecu ste rodjeni? Zapišite taj dan u binarnom sistemu. Napišite dan u mjesecu kada je rodjen vaš prijatelj.

Pokušajte odgonetnuti koji brojevi su kodirani:

$$\boxed{\text{X}} \ \boxed{\checkmark} \ \boxed{\text{X}} \ \boxed{\text{X}} \ \boxed{\checkmark} = \\ (\checkmark=1, \text{X}=0)$$

$$\boxed{\leftarrow} \ \boxed{\leftarrow} \ \boxed{\leftarrow} \ \boxed{\leftarrow} = \\ (\leftarrow=1, \rightarrow=0)$$

$$\boxed{\uparrow} \ \boxed{\downarrow} \ \boxed{\uparrow} = \\ (\uparrow=1, \downarrow=0)$$

$$\boxed{+} \ \boxed{+} \ \boxed{\times} \ \boxed{+} = \\ (+=1, \times=0)$$

$$\circlearrowleft \circlearrowleft \circlearrowleft \circlearrowleft \circlearrowleft = \\ (\odot=1, \circlearrowleft=0)$$

$$\circlearrowright \circlearrowright \circlearrowright \circlearrowright \circlearrowright = \\ (\circlearrowright=1, \circlearrowleft=0)$$

$$\boxed{-} \ \boxed{-} = \\ (-=1, -=0)$$

$$\blacktriangle \ \blacktriangledown \ \blacktriangle \ \blacktriangledown \ \blacktriangle = \\ (\blacktriangle=1, \blacktriangledown=0)$$

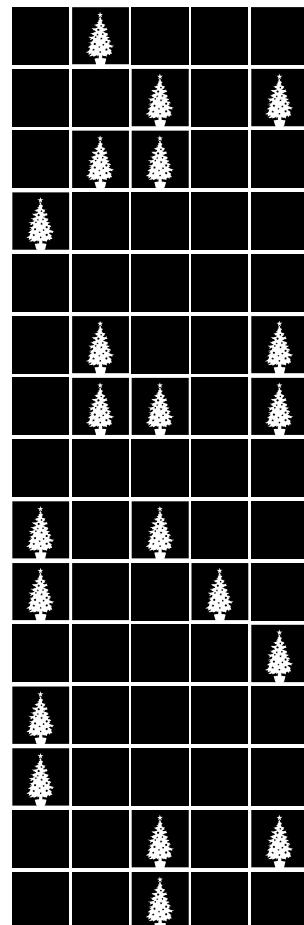
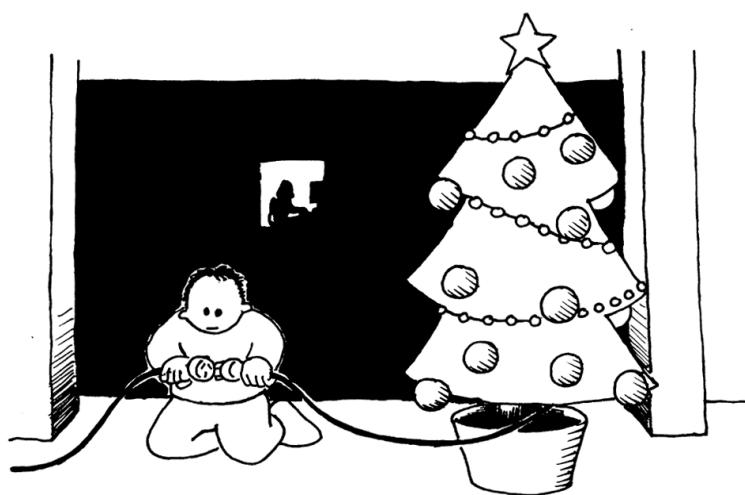
$$\circledcirc \ \circledcirc = \\ (\circledcirc=1, \circledcirc=0)$$

$$\spadesuit \ \clubsuit \ \heartsuit \ \diamondsuit \ \spadesuit = \\ (\spadesuit=1, \clubsuit=0)$$

Ekstra za Eksperte: Koristeći komplet šipki dužina 1, 2, 4, 8 i 16 mjernih jedinica pokažite kako možete izmjeriti svaku dužinu od 0 do 31 jedinica. Ili iznenadite odrasle objašnjavajući kako možete uz pomoć vase sa dva tasa i samo nekoliko težina izmjeriti težinu mnogo različitih, teških stvari kao što su koferi ili kutije!

Radni List za Aktivnost: Slanje tajnih poruka

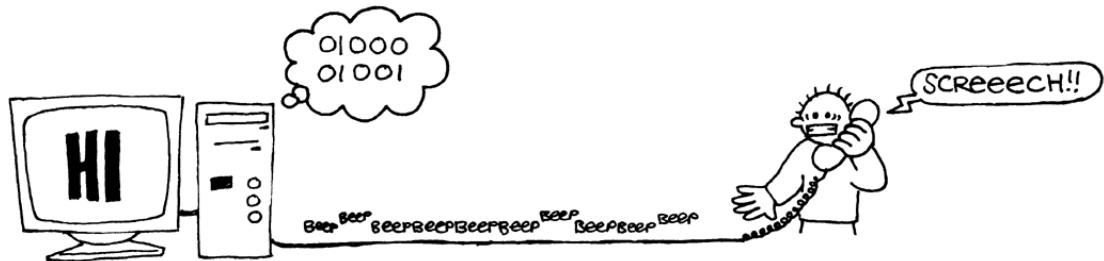
Tom je zarobljen na posljednjem spratu robne kuće. Upravo počinju praznici i on želi ići kući sa svojim poklonima. Šta može uraditi? Već je pokušao zvati u pomoć, čak se i deroao, ali nema nikoga ko ga može čuti. Preko puta ulice je primjetio nekoga ko se bavi računarima i koji je ostao da radi do kasno u noć. Tom pokušava smisliti način kako da privuče njegovu pažnju? Tom je istražio sprat da vidi šta bi moglo biti od koristi. Došao je na super ideju—može iskoristiti svjetla ukrasnih jelki kako bi poslao poruku preko puta ulice! Našao je i skupio sva svjetla i uključio ih tako da može na lagan način da ih uključuje i isključuje. Koristiće jednostavan binarni kod, za koji zna sa sigurnošću da gospodja preko puta može razumjeti. Možete li mu pomoći da pošalje poruku?



1	2	3	4	5	6	7	8	9	10	11	12	13
A	b	c	D	e	f	g	h	i	j	k	l	m
14	15	16	17	18	19	20	21	22	23	24	25	26

Radni List za Aktivnost: E-mail i Modemi

Računari spojeni na internet pomoću modema takodje koriste binarni sistem za slanje poruka (i informacija u opštem slučaju). Jedina razlika je da oni koriste biiip. Biip sa visokim tonom može predstavljati jedinicu a biip sa niskim tonom se može koristiti da predstavimo nulu. Ovi tonovi idu velikom brzinom, tako brzo u stvari da sve što mi možemo čuti je neugodan neujednačen zvuk. Modemi su stara tehnologija i moguće je da ih nikada niste vidjeli ni čuli, ali možete probati pozvati fax i čut ćete isti zvuk—faks koristi takodje modeme da bi slao informacije.



Koristeći isti kod koji je koristio Tom na posljednjem spratu robne kuće, pokušaj poslati e-mail svom prijatelju. Učini to na način da bude lagan i za tebe i za tvog prijatelja—nije moguće, a ni potrebno da budeš brz kao modem!



Radni List za Aktivnost: Brojanje brojeva većih od 31

Posmatrajmo još malo binarne karte. Ukoliko bi željeli da napravite još jednu, novu kartu u nizu : koliko bi tačaka bilo na toj tački? Kako bi izgledala nova sljedeće karta poslije ove? Koje je pravilo koje ste primjenili za pravljenje novih karata? Kao što vidite i sami samo nekoliko karata nam je potrebno da bi brojali i radili i sa stvarno velikim brojevima.

Ukoliko pažljivo posmatrate dobiveni niz, možete primjetiti jako interesantnu vezu:

1, 2, 4, 8, 16...

Pokušajte sabrati: $1 + 2 + 4 = ?$ Koji je rezultat koji ste dobili?

Sada pokušajte $1 + 2 + 4 + 8 = ?$

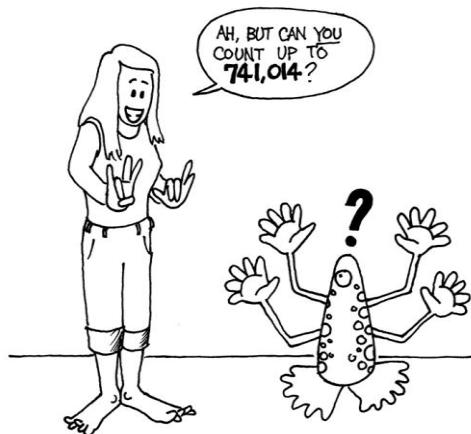
Šta se dogadja ako saberete sve brojeve od početka do kraja?

Da li ste ikada čuli pjesmu “Neka tvoji prsti prošeću”? Upravo sada možete računati pomoću svojih prstiju, i to tako da radite sa brojevima mnogo većim od 10, a da ne izgledate kao neko iz svemira! Ukoliko koristite binarni sistem tako da svaki prst jedne ruke predstavlja jednu kartu sa tačkicama onda možete brojati od 0 do 31. To su ukupno 32 broja. (Ne zaboravimo da je nula takodje broj!)

Pokušajte sada brojati koristeći sve prste na obje ruke. Ukoliko je prst ispružen onda je to jedan, a ako je savijen onda on predstavlja nulu.

Na takav način možete brojati sve od 0 do 1023 ako koristite obje ruke! To je ukupno 1024 brojeva!

Sada ako imate stvarno gipke nožne prste (onda možda i izgledate kao iz svemira) onda možete ići još i dalje i više. Ako jedna ruka može biti od koristi za brojanje 32 broja, a dvije ruke nam omogućavaju brojanje $32 \times 32 = 1024$ brojeva, koji je najveći broj koji gospodjica Gipki-Nožni-Prsti može dostići?



Radni List za Aktivnost: Još i više o Binarnim Brojevima

1. Još jedna interesantna osobina binarnih brojeva je ono što se dogadja kada dopištete nulu sa desne strane nekog binarnog broja. Ukoliko radimo sa bazom 10 (decimalni brojevi), kada dopišete nulu na desnu stranu nekog broja onda dobijete 10 puta veći broj (kao da ste početni broj pomnožili sa 10). Na primjer, 9 postaje 90, a 30 postaje 300.

Ali šta se dogadja ako dopišete 0 sa desne strane nekog binarnog broja? Probajte ovo:

$$\begin{array}{r} \mathbf{1001} \\ (9) \end{array} \rightarrow \begin{array}{r} \mathbf{10010} \\ (?) \end{array}$$

Razmislite o još nekoliko primjera kako bi bolje testirali vašu tvrdnju. Dakle koje je pravilo? Šta mislite da se dogadja?

2. Svaka karta koju smo koristili do sada predstavlja jedan ‘bit’ u računaru (‘bit’ je engleska skraćenica za ‘binary digit’). Pa prema tome naš kod za alfabet koji smo koristili može biti predstavljen sa samo pet karata, ili ‘bitova’. Ipak, računar treba da razlikuje da li su slova mala ili velika, i treba da prepoznaće i predstavlja i cifre, znake interpunkcije kao i neke specijalne simbole kao što su \$ ili ~.

Pogledajmo sada jednu uobičajenu tastaturu za računar i odgovorimo na pitanje koliko znakova (karaktera) jedan računar mora moći predstaviti. Koliko dakle bitova neki računar mora koristiti da predstavi (i sačuva) sve te različite znakove?

Većina računara danas koristi standardizovano predstavljanje koje zovemo ASCII (American Standard Code for Information Interchange), koji je zasnovan na korištenju 8 bitova po jednom znaku, ali neke zemlje sa ne-Engleskog govornog područja moraju koristiti kodove koji su i duži od osam bitova.



Zašto je ovo sve važno?

Računari danas koriste binarni brojni sistem da bi predstavili informaciju. Nazivamo ga binarni zato što koristimo samo dvije različite cifre, nulu i jedan. Nekada taj sistem nazivamo i *po bazi dva* (ljudi obično koriste sistem po bazi 10). Svaka nula ili jedinica se naziva i *bit* (**binary digit**). Jedan bit je obično predstavljen u glavnoj memoriji računara pomoću jednog tranzistora koji je uključen ili isključen, ili pomoću kapaciteta koji je napunjeno ili prazan.



Kada neke podatke trebamo prenijeti preko telefonske ili radio veze, onda se visoki i niski tonovi koriste kao nule i jedinice. Na magnetnim diskovima (tvrdom disku) i trakama, bitovi su predstavljeni orijentacijom magnetnog polja na obloženoj površini, orijentacija je dakle ili Sjever-Jug ili Jug-Sjever.



Audio CD, CD-ROM kao i DVD zapisuju bitove optički—dio površine koji odgovara jednom bitu reflektuje ili ne svjetlost.

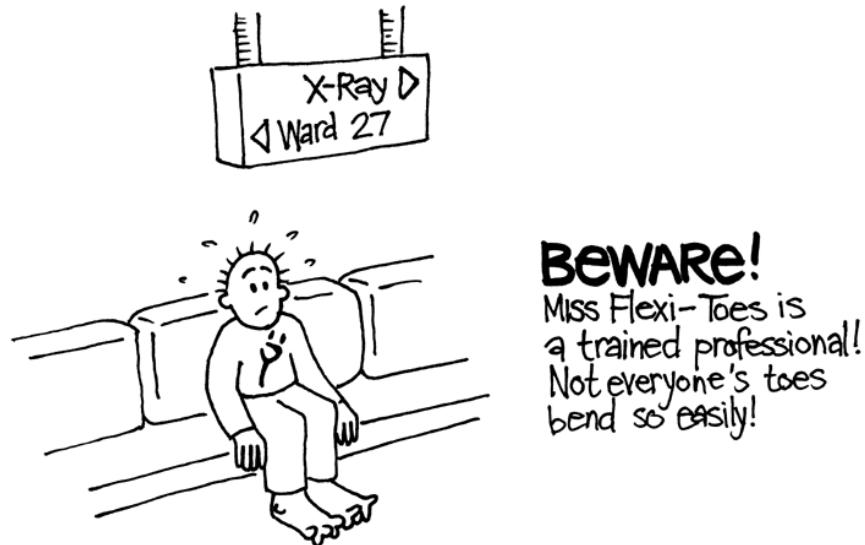


Razlog zašto računari koriste samo dvije različite vrijednosti je taj što je mnogo lakše tehnički napraviti funkcionalne uređaje koji koriste upravo samo dvije vrijednosti. Moguće bi bilo imati i CD-ove koji imaju 10 različitih nivoa refleksije svjetlosti tako da možemo predstaviti sve cifre od 0 do 9, ali bi onda morali napraviti (jako) skup i precizan uređaj koji to sve može koristiti (zapisivati i čitati 10 različitih refleksija). Jedna druga stvar koju ste mogli primjetiti je da iako mi govorimo da računari zapisuju i rade sa nulama i jedinicama, u stvari nema nikakvih nula i jedinica u računaru—postoje samo visoki i niski napon, ili sjever/jug orijentacija magnetnog polja, i tako dalje. To je zato što je brže i razumljivije napisati i reći “0” i “1” od stvari kao što su “svjetao” i “taman”. Sve što imamo na računarima je predstavljeno koristeći ove bitove – dokumenti, slike, pjesme, video, brojevi, pa i cijeli programi i aplikacije koje koristimo su u stvari samo skup (mnogo) nula i jedinica, binarnih cifara.

Samo jedan bit sam za sebe ne može predstaviti mnogo stvari tako da se bitovi obično grupišu u grupe od osam bitova i onda možemo predstaviti brojeve od 0 do 255. Jedna grupa od osam bitova se naziva bajt (eng. byte).

Brzina jednog računara zavisi koliko bitova računar može obraditi (procesirati) u jednom trenutku. Na primjer, jedan 32-bitni računar može procesirati 32-bitni broj u samo jednoj operaciji dok neki 16-bitni računar mora 32-bitni broj razbiti na manje dijelove pa je onda i sporiji (ali i jeftiniji!).

U nekim od narednih aktivnosti vidjećemo kako drugi oblici informacija mogu biti predstavljeni u računaru pomoću binarnih cifara, dakle samo pomoću nula i jedinica.



Rješenja i pomoć

Binarni Brojevi (strana 6)

za **3** trebamo karte 2 i 1

za **12** trebamo karte 8 i 4

za **19** trebamo karte 16, 2 i 1

Postoji samo jedan način da predstavimo (dobijemo) neki broj.

Najveći broj koji možete dobiti je 31. Najmanji broj je 0. Moguće je napraviti bilo koji broj izmedju ova dva broja, i svaki od njih ima samo jednu reprezentaciju (način dobivanja je jedinstven).

Eksperts: Da bi povećali neki broj za je, treba okretati karte redom sa desna na lijevo sve dok ne okrenete neku kartu tako da joj lice dodje gore.

Rad sa binarnim brojevima (strana 8)

$10101 = 21, 11111 = 31$

Slanje tajnih poruka (strana 9)

Codirana poruka je : HELP IM TRAPPED

Brojanje brojeva većih od 31 (strana 11)

Ako saberete sve brojeve od početka zbir će uvijek biti za jedan manji od sljedećeg broja u nizu.

Gospodjica Gipki-Nožni-Prsti može prebrojati $1024 \times 1024 = 1,048,576$ brojeva—od 0 do 1,048,575!

Više o Binarnim Brojevima (strana 12)

Kada dopišete nulu sa desne strane nekog binarnog broja onda se taj broj udupla.

Sva mjesta koja su imala jedinicu sada vrijede duplo više od prethodne vrijednosti tako da će se cijeli broj uduplati. (U bazi 10 dopisivanje nule sa desne strane množi broj sa 10, uvećava mu vrijednost 10 puta.)

Računaru je potrebno 7 bitova da predstavi sve znakove (karaktere). To vam omogućava predstavljanje do 128 znakova. Uobičajeno je da 7 bitova zipišemo u grupu od 8 bitova s tim da je taj jedan bit viška potrošen uzalud, to jest nekoristan i izgubljen.

Aktivnost 2

Boje pomoću brojeva—Predstavljanje slika

Sažetak

Računari čuvaju slike, fotografije i druge tipove vizualnih objekata koristeći samo brojeve. Sljedeća aktivnost pokazuje na koji način se to radi, i kako je uopšte to moguće.

Veza sa Curriculum-om

- ✓ Matematika: Geometrija – oblici i prostori
- ✓ Tehnologija: korištenje cijelih brojeva za predstavljanje drugih tipova podataka
- ✓ Tehnologija: reduciranje prostora koji se koristi za smještaj podataka koji se ponavljaju

Vještine

- ✓ Brojanje i prebrojavanje
- ✓ Grafičko izražavanje

Dobna/Starosna grupa

- ✓ 7 godina i stariji

Potreban materijal

- ✓ Slide za prezentaciju: Boje pomoću brojeva(strana 19)
- ✓ Svaki učenik će trebati:
- ✓ List za Aktivnost: Kid Fax (strana 20)
- ✓ List za Aktivnost: Napravite svoju sopstvenu sliku (strana 21)

Boje pomoću Brojeva

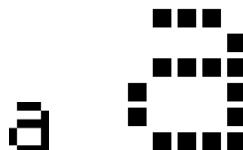
Uvod

Pitanja za diskusiju

1. Šta radi jedna fax mašina?
 2. U kojim situacijama bi računari trebali sačuvati neku sliku? (Program za crtanje, igrica sa grafikom, ili korištenje nekog multimedijalnog sistema.)
 3. Na koji način računari čuvaju slike ako znamo da su u stanju raditi samo sa brojevima ?

(Možete probati organizovati sa učenicima da pošalju/prime fax kao pripremu za ovu aktivnost)

Demonstracija uz korištenje projektor-a



Monitor jednog računara je podjeljen u mrežu malih tačkica koje nazivamo piksel (eng. *pixels (picture elements)*).

U jednoj crno bijeloj slici, svaki piksel je ili crne ili bijele boje.

Ovdje gore je prikazano uvećano slovo "a" kako bi jasno mogli rasponzati piksele. Kada jedan računar želi sačuvati neku sliku onda sve što treba uraditi je da sačuva koje tačke su crne i koje tačke su bijele.

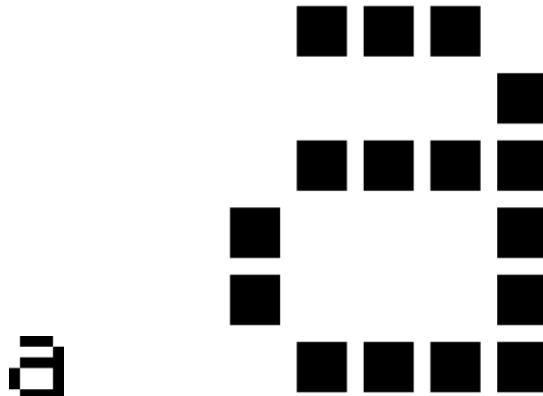
							1, 3, 1
							4, 1
							1, 4
							0, 1, 3, 1
							0, 1, 3, 1
							1, 4

Gornja slika nam pokazuje kako jedna slika može biti predstavljene brojevima. Prva linija se sastoji od jednog bijelog piksela pa onda slijede tri crna piksela i na kraju dolazi jedan bijeli piksel. Prema tome, prvu liniju možemo predstaviti brojevima 1, 3, 1.

Prvi broj označava broj bijelih piksela. Ukoliko je prvi piksel u liniji crne boje onda će reprezentacija započeti sa nulom.

List za aktivnost na strani 20 predstavlja nekoliko prigodno izabranih slika koje učenici mogu dekodirati koristeći upravo predstavljenu metodu.

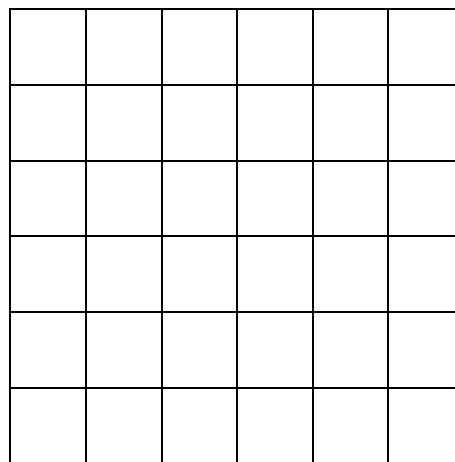
Boje pomoću brojeva



- ▲ Slovo "a" uzeto sa ekrana jednog računara i odgovarajuće uvećanje tako da se jasno vidi koji pikseli čine slova i na koji način se pravi slika slova "a".

	1, 3, 1
	4, 1
	1, 4
	0, 1, 3, 1
	0, 1, 3, 1
	1, 4

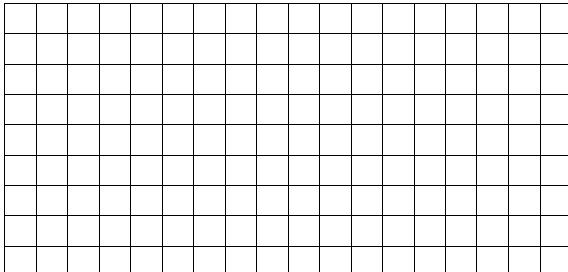
- ▲ Ista slika koja odgovara slovu "a" kodirana samo pomoću brojeva



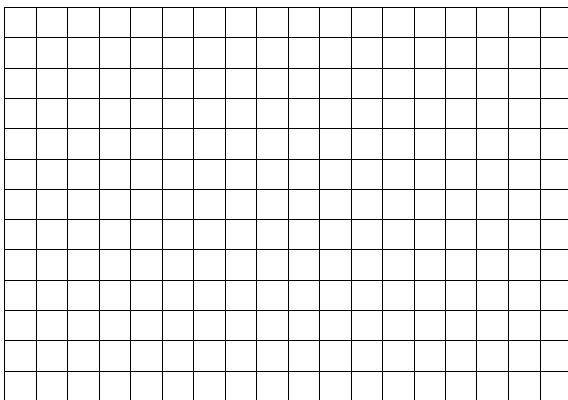
- ▲ Prazna mreža (za učenje i vježbanje)

Radni List za Aktivnost: Dječiji Fax

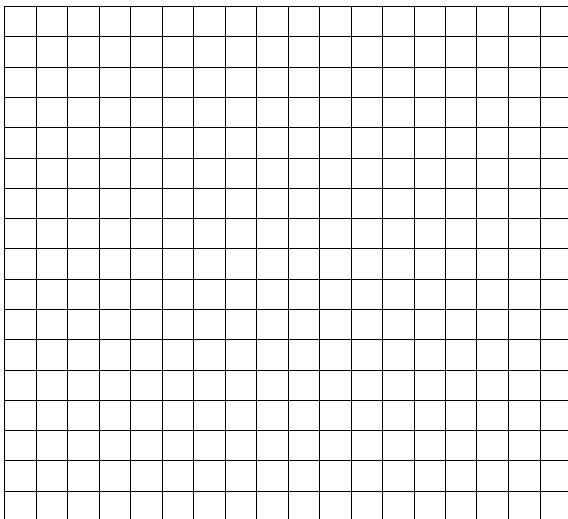
Prva slika je najlakša dok je ona posljednja najteža. Jako je lagano napraviti greške pa je dobra ideja da koristimo običnu grafitnu olovku za bojenja i da imamo guminu za brisanje pri ruci!



4, 11
4, 9, 2, 1
4, 9, 2, 1
4, 11
4, 9
4, 9
5, 7
0, 17
1, 15



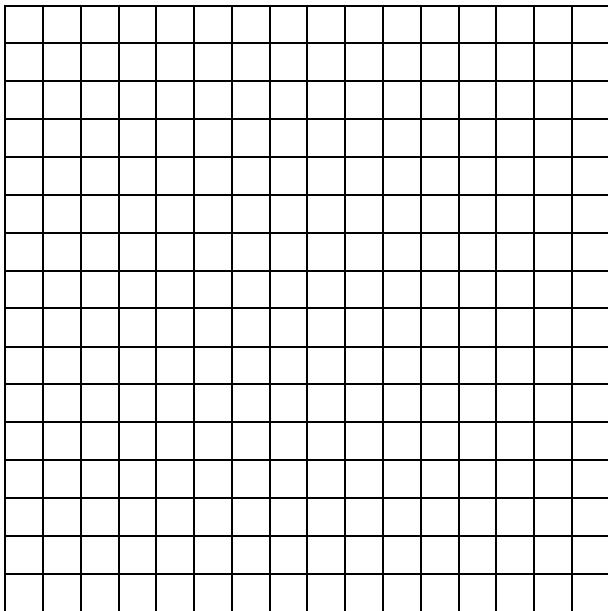
6, 5, 2, 3
4, 2, 5, 2, 3, 1
3, 1, 9, 1, 2, 1
3, 1, 9, 1, 1, 1
2, 1, 11, 1
2, 1, 10, 2
2, 1, 9, 1, 1, 1
2, 1, 8, 1, 2, 1
2, 1, 7, 1, 3, 1
1, 1, 1, 1, 4, 2, 3, 1
0, 1, 2, 1, 2, 2, 5, 1
0, 1, 3, 2, 5, 2
1, 3, 2, 5

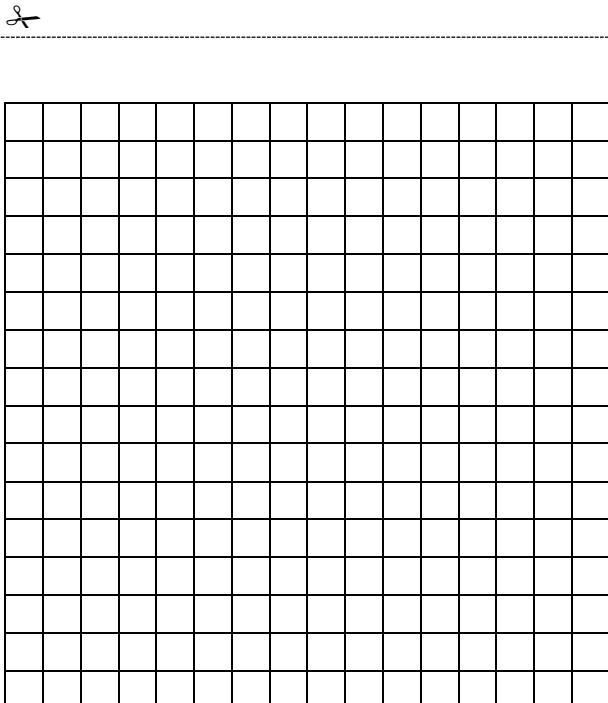


6, 2, 2, 2
5, 1, 2, 2, 2, 1
6, 6
4, 2, 6, 2
3, 1, 10, 1
2, 1, 12, 1
2, 1, 3, 1, 4, 1, 3, 1
1, 2, 12, 2
0, 1, 16, 1
0, 1, 6, 1, 2, 1, 6, 1
0, 1, 7, 2, 7, 1
1, 1, 14, 1
2, 1, 12, 1
2, 1, 5, 2, 5, 1
3, 1, 10, 1
4, 2, 6, 2
6, 6

Radni List za Aktivnost: Napravite svoju sopstvenu sliku

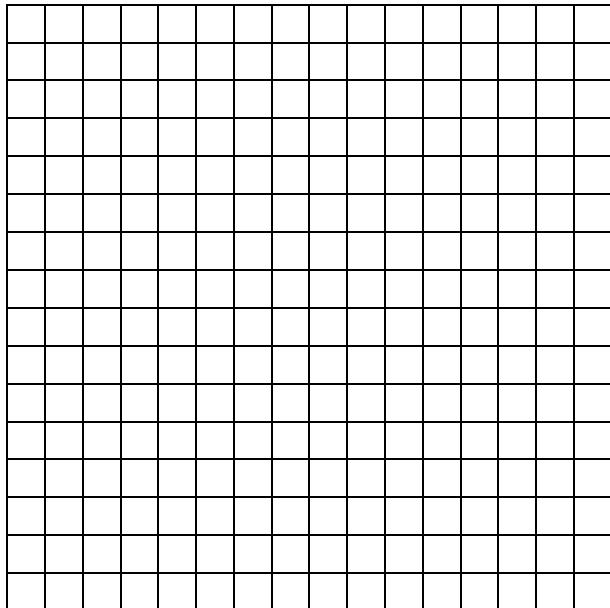
Sada kada znate kako se brojevi mogu iskoristiti za predstavljanje slika zašto ne bi probali kodirati svoju sopstvenu sliku za svog prijatelja? Nacrtajte svoju sopstvenu sliku na gornjoj praznoj mreži i kada završite zapišite brojeve za kodiranje te slike pored donje mreže. Presjecite papir duž isprekidane linije i predajte mrežu na donjem dijelu papira svom prijatelju da je oboji. (Pažnja: niste obavezni da koristite cijelu ponuđenu mrežu ukoliko to ne želite—jednostavno ostavite prazne neiskorištene linija na dnu mreže.)

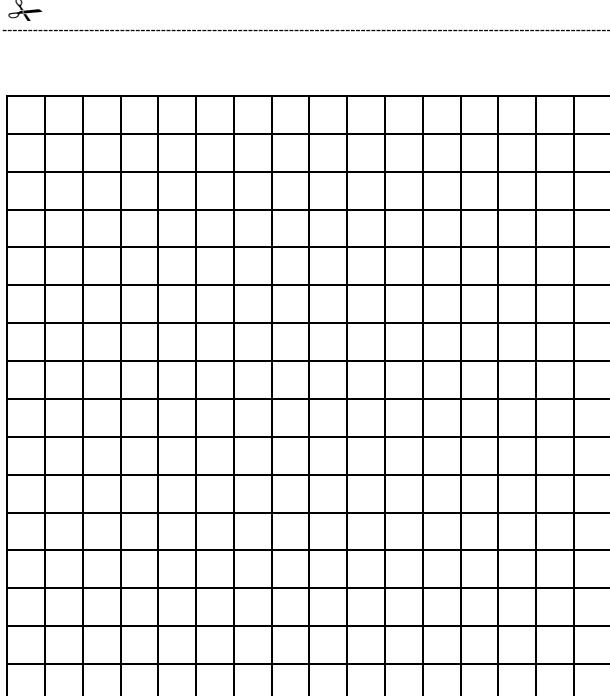




Radni List za Aktivnost: Napravite svoju sopstvenu sliku

Ekstra za Eksperte: Iukoliko želite da proizvedete slike u boji onda možete koristiti brojeve koje će predstavljati različite boje (na primjer 0 je crna, 1 je crvena, 2 je zelena itd.). Sada koristimo dva broja kako bi predstavili jedan niz piksela: prvi broj nam daje dužinu niza kao i ranije dok drugi broj označava boju piksela. Pokušajte nacrtati sliku u boji za svog prijatelja. Ne zaboravite reći svom prijatelju koji broj označava koju boju!





Varijacije i dodatne teme

1. Pokušajte nacrtati sliku na providnom papiru postavljenom iznad mreže tako da krajnju sliku možemo vidjeti i bez mreže. Slika koju dobijemo će biti mnogo jasnija.
2. Umjesto da boje mrežu na papiru učenici mogu koristiti ljepljive papiriće, ili postavljati prigodne objekte, na nekoj većoj mreži.

Mjesto za diskusiju

Obično postoji ograničenje za dužinu niza piksela iste boje jer je dužina predstavljena jednim binarnim brojem. Kako bi predstavili niz od dvanaest crnih piksela ukoliko možete koristiti samo brojeve od 0 do 7. (Dobra ideja bi bila da to kodirate prvo kao niz od sedam crnih piksela, pa onda nula bijelih piksela pa na kraju niz od pet crnih piksela.)

Zašto je ovo sve važno?

Jedna fax mašina je zaista jedan jednostavan računar koji skenira crno-bijele stranice u mrežu od oko 1000×2000 piksela. Ta se mreža piksela onda šalje pomoću modema prema drugoj fax mašini, koja onda štampa piksele na list papira. Uobičajeno je da slike koje koristi fax imaju veće komade bijelog prostora (na primjer, marge) ili dijelove od crnih piksela (na primjer, horizontalne linije). Slike u bojama takođe često imaju mnogo dijelova koji se ponavljaju. Kako bi uštedio na prostoru za čuvanje takvih slika programeri mogu koristiti različite tehnike za kompresiju podataka. Metod koji je korišten u ovoj aktivnosti se naziva ‘po-dužini kodiranje’ (eng. ‘run-length coding’), i to je jedan dosta efikasan način za kompresiju slika. Da nismo koristili kompresiju slika onda bi bilo potrebno puno više vremena za prenos slika i bilo bi potrebno puno više prostora za njihovo čuvanje. U krajnjoj liniji, to bi učinilo praktično nemogućim (preskupim) slanje faxa, ili mogućnost da postavimo svoje slike na neku web stranicu. Na primjer, slika sa fax mašina su kompresovane na oko sedminu njihove orginalne veličine. Bez korištenja kompresije bilo bi potrebno sedam puta više vremena za njihovo slanje!

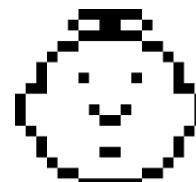
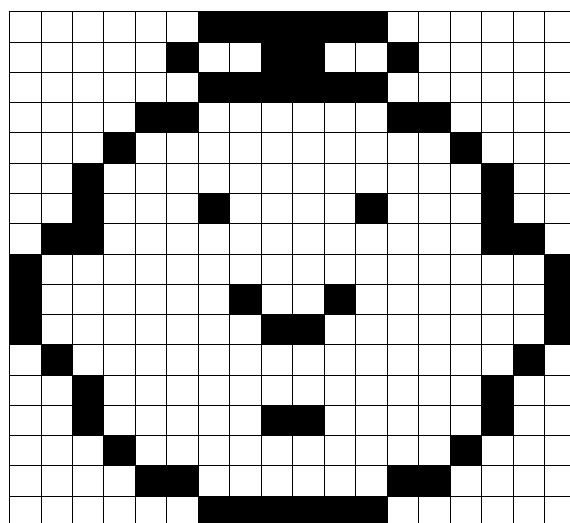
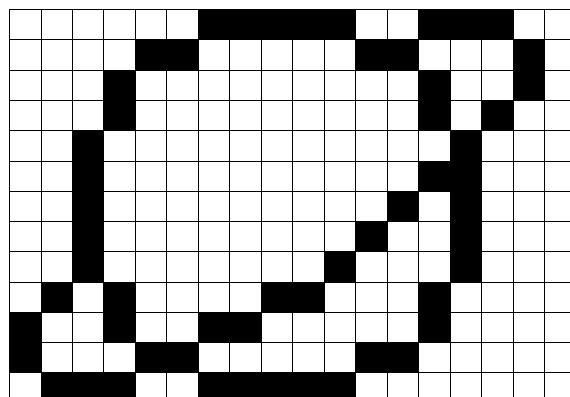
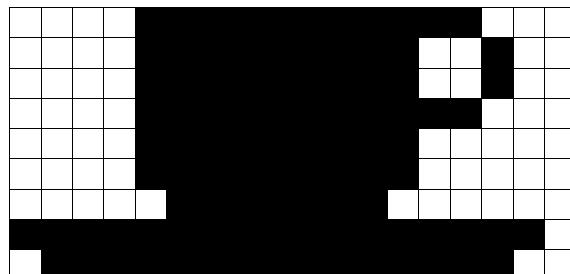
Fotografije i slike na računarima su obično kompresovane na deseti ili čak stoti dio njihove ukupne veličine (korištenjem odgovarajućih tehnika kao što su JPEG, GIF i PNG). Ovo omogućava čuvanje puno više ovakvih slika na disku, i znači takođe da se njihovo pregledanje na web-u može uraditi vrlo brzo.

Sam programer može izabrati koja tehnika kompresije će najbolje poslužiti za slike koje ona ili on želi poslati.



Rješenja i pomoć

Odgovori za radni list aktivnosti : Dječiji Fax



Aktivnost 3

Možete to reći ponovo! —Kompresija (sažimanje) Teksta

Sažetak

Kako računari imaju samo ograničen prostor za čuvanje informacija, potrebno je predstaviti sve te informacije što je moguće efikasnije. Ovu tehniku nazivamo kompresija ili sažimanje. Kodiranjem podataka prije nego budu sačuvani, kao i njihovim dekodiranjem kada ih želimo koristiti, računar može sačuvati više podataka, ili može poslati podatke internetom na brži način.

Veza sa Curriculum-om

- ✓ BHS: Prepoznavanje uzoraka u riječima i u tekstu.
- ✓ Tehnologija: reduciranje prostora koji koristimo za podatke koji se ponavljaju

Vještine

- ✓ Kopiranje napisanog teksta

Dobna/Starosna grupa

- ✓ 7 godina i stariji

Potreban materijal

- ✓ Slide za prezentaciju: Možete to reći ponovo! (strana 28)
Each student will need:
 - ✓ List za Aktivnost: You can say that again! (strana 29)
 - ✓ List za Aktivnost: Ekstra za eksperte (strana 30)
 - ✓ List za Aktivnost: Kratko i slatko (strana 31)
 - ✓ List za Aktivnost: Ekstra za prave eksperte (strana 32)

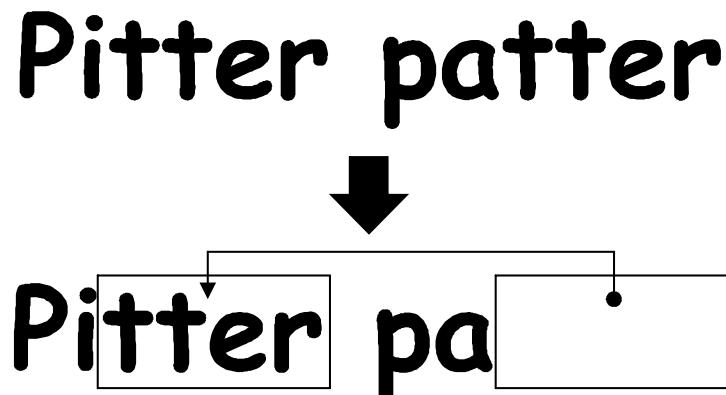
Možete to reći ponovo!

Uvod

Računari moraju sačuvati ili poslati velike količine podataka. Kako ne bi morali koristiti zaista veliki prostor za čuvanje podataka, ili da ne bi puno vremena trošili na slanje podataka preko mreže (interneta) računari sažimaju (kompresuju) tekst otprilike na sljedeći način.

Demonstracija i diskusija

Pokažite “The Rain” slide (strana 28). Potražite uzorke od nekoliko slova koji se ponavljaju u ovoj pjesmi. Možete li naći grupe od 2 ili više slova koji se ponavljaju, ili da li možete naći cijele riječi ili možda i cijele rečenice koje se ponavljaju? (Zamjenite ih sa kvadratićima kao što je prikazano na ovom dijagramu ovdje dolje.)



Možete to reči ponovo!

The Rain

Pitter patter

Pitter patter

Listen to the rain

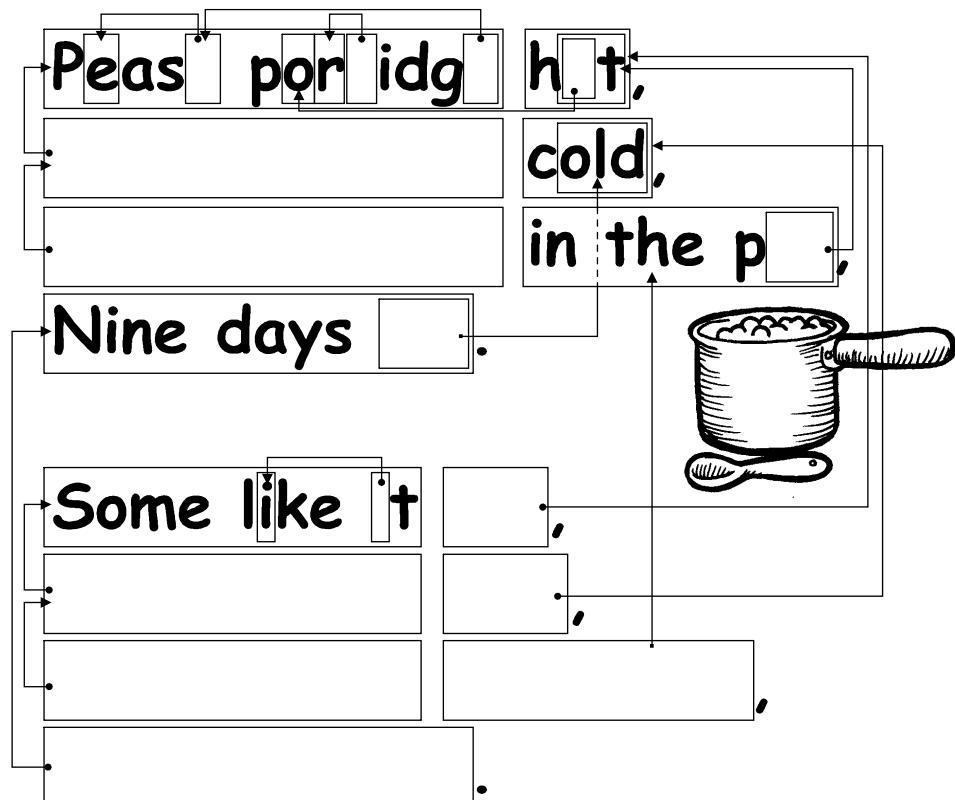
Pitter patter

Pitter patter

On the window pane

Radni List za Aktivnost: Možete to reći ponovo!

Many of the words and letters are missing in this poem. Can you fill in the missing letters and words to complete it correctly? You will find these in the box that the arrow is pointing to.



Sada izaberite neku jednostavnu pjesmicu ili možda uspavanku i kreirajte svoju sopstvenu zagonetku. Osigurajte da sve vaše strelice pokazuju na neka ranija mesta u tekstu. Vaša pjesmica treba da se može dekodirati idući sa lijeva na desno i od vrha prema dolje na isti načina na koji uobičajeno čitamo neki tekst.

Izazov: Istražite koji je najmanji broj orginalnih riječi moramo sačuvati!

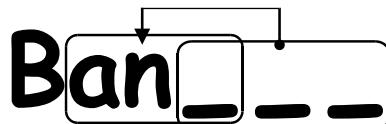
Evo nekoliko prijedloga pjesmica: Three Blind Mice, Mary Mary Quite Contrary, Hickory Dickory Dock—or try some Dr Seuss books!

Pomoć: Pokušajte da izbjegnete preveliku gužvu crtajući svoje strelice. Ostavite dovoljno prostora oko slova i oko riječi dok ih zapisujete tako da imate dovoljno prostora za kvadratiće unutar kvadratića kao i za strelice koje pokazuju na njih.

Biće dosta lakše ukoliko kreirate zagonetku tako što napišete pjesmicu prvu pa tek onda odlučite koje kvadratiće želite koristiti.

Radni List za Aktivnost: Ekstra for Eksperete

Kako bi riješili datu zagonetku?



Nekada dio nedostajućeg teksta pokazuje na dio tog istog teksta. U tom slučaju je još uvijek moguće dekodirati tekst na pravilan način ukoliko slova kopiramo sa lijeva na desno. U tom slučaju, svako slovo je poznato i dostupno za kopiranje prije nego što je stvarno potrebno. Ovo je naročito korisno ako u računaru imamo neki dugi niz istog slova ili ponavljujućeg uzorka.

Pokušajte naći neki vaš primjer za ovu tehniku.

U računarnim su kvadratići i stelice predstavljeni takodje brojevima. Na primjer,

Banana

se može napisati kao **Ban(2,3)**. “2” označava da treba brojati unazad dva karaktera kako bi došli do početne pozicije koju ćemo kopirati,

Ban...

a onda “3” označava da treba kopirati sljedeća tri slova:

Ban--

Banan-

Banana

Kako se koriste dva broja za kodiranje ovih riječi onda je jedino interesantno sažimati grupe od dva ili više slova, jer inače ne bi bilo nikakvog sažimanja memorijskog prostora. U stvari, veličina datoteke se može i povećati ukoliko bi koristili dva broja za kodiranje jednog slova.



Predložite neke svoje riječi koje želite kodirati i zapишite ih na način na koji bi ih računar zapisao u kompresovanom obliku. Da li ih vaši prijatelji mogu dekodirati?

Radni List za Aktivnost: Kratko i slatko

Koliko stvarno riječi vam je potrebno ovdje?

Pretvarajte se da ste računar koji pokušava da zapiše što više podataka na svoj disk. Prekrižite sve grupe od dva ili više slova koji su se već bili pojavili. Oni više neće biti potrebni jer se mogu zamijeniti sa strelicom (eng. Pointer-om). Vaš cilj je da prekrižite što je više moguće slova.

I know an old lady who swallowed a bird
How absurd! She swallowed a bird!

She swallowed the bird to catch the spider
That wriggled and jiggled
and tickled inside her

She swallowed the spider to catch the fly
I don't know why she swallowed a fly
Perhaps she'll die...

Radni List za Aktivnost: Ekstra za Prave Eksperte

Jeste li spremni za jednu zaista zahtjevnu kompresiju?

Priča koja slijedi je obrađena pomoću jednog računarskog programa koji je našao da je moguće prekrižiti najmanje 1,633 slova. Koliko slova koja se mogu prekrižiti možete naći sami? Zapamtite, samo grupe od dva ili više slova koje se ponavljaju mogu biti eliminisane. Neka vam je sa srećom!

Once upon a time, long, long ago, three little pigs set out to make their fortunes. The first little pig wasn't very clever, and decided to build his house out of straw, because it was cheap. The second little pig wasn't very clever either, and decided to build his house out of sticks, for the "natural" look that was so very much in fashion, even in those days. The third little pig was much smarter than his two brothers, and bought a load of bricks in a nearby town, with which to construct a sturdy but comfortable country home.

Not long after his housewarming party, the first little pig was curled up in a chair reading a book, when there came a knock at the door. It was the big bad wolf, naturally.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the first little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he *did* huff, and he *did* puff, and the house soon collapsed. The first little pig ran as fast as he could to the house of sticks, and was soon safe inside. But it wasn't long before the wolf came calling again.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the second little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he *did* huff, and he *did* puff, and the house was soon so much firewood. The two terrified little pigs ran all the way to their brother's brick house, but the wolf was hot on their heels, and soon he was on the doorstep.

"Little pig, little pig, let me come in!" cried the wolf.

"Not by the hair on my chinny-chin-chin!" squealed the third little pig.

"Then I'll huff, and I'll puff, and I'll blow your house down!" roared the wolf, and he huffed, and he puffed, and he huffed some more, but of course, the house was built of brick, and the wolf was soon out of breath. Then he had an idea. The chimney! He clambered up a handy oak tree onto the roof, only to find that there was no chimney, because the third little pig, being conscious of the environment, had installed electric heating. In his frustration, the wolf slipped and fell off the roof, breaking his left leg, and severely injuring his pride. As he limped away, the pigs laughed, and remarked how much more sensible it was to live in the city, where the only wolves were in the zoo. And so that is what they did, and of course they all lived happily ever after.

Zašto je ovo sve važno?

Prostor za čuvanje podataka na računarima raste nevjerojatnom brzinom posljednjih 25 godina, pa se tako količina podataka koju možemo sačuvati na jednom računaru milion puta uvećala—ali nam je uvijek potrebno više prostora na našim računarima. Računari sada mogu čuvati cijele knjige ili čak cijele biblioteke, a onda i mnog muzike i puno filmova, jedino je važno da imaju dovoljno prostora za smještaj podataka. Velike datoteke predstavljaju takodje problem za Internet, jer mogu trebatи previše vremena za preuzimanje (eng. download). U isto vrijeme želimo napraviti računare što je moguće manjim—sada čak i od smartphone-a ili elektronskog sata očekujemo da može čuvati mnogo informacija!

Ipak postoji jedno rješenje za ovaj problem. Umjesto da kupujemo još više prostora za čuvanje podataka, ili da obezbjedimo bržu Internet konekciju, možemo sažeti svoje podatke tako da zahtjevaju manje prostora za čuvanje. Ovaj proces kompresovanja i dekompresovanja podataka se obično radi na automatski način od strane samog računara. Mi ćemo jednostavno primjetiti da možemo sačuvati više podataka na našem disku ili da se pak web stranice brže pokazuju. U stvari, naš računar samo izvršava više operacija manipulacije podacima (procesira više i detaljnije naše podatke).

Mnogo različitih metoda kompresije je otkriveno i izumljenno. Metod koji smo koristili tokom ove aktivnosti, koji u principu pokazuje na dijelove teksta koji se pojavljivao ranije, se često naziva ‘Ziv-Lempel koding,’ ili ‘LZ koding’, i otkriven je od strane dva Izraelska profesora tokom sedamdesetih godina prošlog vijeka. Metod možemo koristiti za bilo koji jezik i vrlo često je u stanju prepovoljiti količinu podataka koji se kompresuju (prostor potreban za zapisivanje podataka se prepolovi). Ponekad ga nazivamo i ‘zip’ metod, a koristi se takodje i za kompresovanje ‘GIF’ i ‘PNG’ tipova slika, a bio je korišten i za modeme velikih brzina. Kada se koristi u modemima, onda smanjuje količinu podataka koje treba poslati preko telefonske linije pa je slanje mnogo brže.

Neki drugi metodi se zasnivaju na ideji da slova koja se koriste češće budu kodirana sa kraćim kodovima u odnosu na slova koja se pojavljuju rjeđe. I Morse-ova kod koristi ovu ideju (još u 19 vijeku kada nije bilo računara).

Rješenja i pomoć

Možete to reći ponovo! (strana 29)

**Pease porridge hot,
Pease porridge cold,
Pease porridge in the pot,
Nine days old.**

**Some like it hot,
Some like it cold,
Some like it in the pot,
Nine days old.**

Aktivnost 4

Magija Okretanja Karata—Otkrivanje i Ispravljanje Grešaka

Sažetak

When data is stored on a disk or transmitted from one computer to another, we usually assume that it doesn't get changed in the process. But sometimes things go wrong and the data is changed accidentally. This activity uses a magic trick to show how to detect when data has been corrupted, and to correct it.

Veze sa Curriculum-om

- ✓ Matematika: Broj – Istraživanje izračunavnaja i približne procjene.
- ✓ Matematika: Algebra – Istraživanje uzoraka i veza, rješavanje i traženje nepoznate vrijednosti.
- ✓ Matematika: Redovi i kolone, koordinate
- ✓ Tehnologija: Validiranje podataka

Vještine

- ✓ Brojanje
- ✓ Prepoznavanje neparnih i parnih brojeva

Dobna/Starosna grupa

- ✓ 7 godina i stariji

Potreban materijal

- ✓ Komplet od 36 "frižider magnet" karata, obojenih samo sa jedne strane
 - ✓ Metalna tabla (takođe bijela magnetna tabla može poslužiti) za demonstraciju.
- Svaki par učenika će trebati:
- ✓ 36 jednakih karata, obojenih samo sa jedne strane

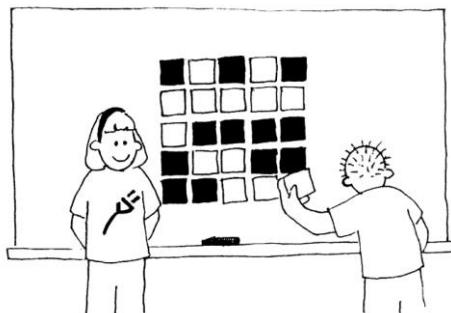
“Magični Trik”

Demonstracija

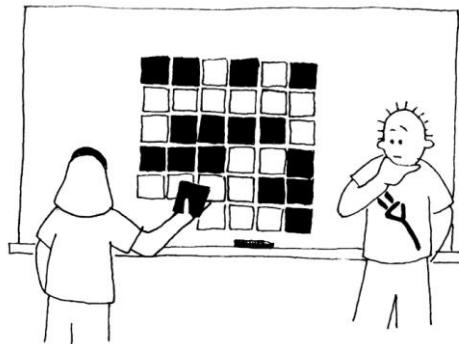
Evo ovdje vaše prilike da budete mađioničar!

Treba vam komplet jednakih karata sa dvije strane. (Kako bi imali svoj vlastiti komplet izrežite ih od velikog parčeta papira obojenog samo sa jedne strane). Za demonstraciju je najpogodnije koristiti ravne magnetne karte koje imaju dvije različite boje na svoje dvije strane—frižider magneti se čine idealnim izborom, jedino se treba uvjeriti da su magnetni na obje strane (veliki broj tih karata ima magnet samo sa jedne strane pa možete zalijepiti dvije karte i označiti jednu stranu sa jednim bijelim krugom).

1. Izaberite učenika koji će postaviti karte u kvadrat dimenzija 5×5 tako što će na slučajan način izabrati koje boje (stranu) će svaka karta pokazivati.



Na slučajan način dodajte jedan red i jednu kolonu, “samo da bi zadatak napravili malkice težim”.



Ove dodatne karte su ključne za cijeli trik. Morate izabrati karte koje ste dodali na takav način da imamo paran broj obojenih karata u svakom redu i u svakoj koloni.

2. Izaberite učenika koji će okrenuti samo jednu kartu dok vi držite svoje oči zatvorenim. Sada će i red i kolona koja sadrži promjenjenu kartu imati neparan broj obojenih karata, pa vam to omogućava da odredite koja je karta promjenjena. Da li učenici mogu pogoditi na čemu je ovaj trik zasnovan?

Naučite svoje učenike istom triku:

1. Radeći u parovim, učenici postavljaju karte u tabelu dimenzija 5×5 .
2. Koliko obojenih karata ima u svakom redu i u svakoj koloni? Da li je to paran ili neparan broj? Zapamtite da je 0 paran broj.
 1. Sada dodajte šestu kartu u svaki red osiguravajući da je ukupan broj obojenih karata paran u svakom redu. Ova dodatna karta se naziva “parna” (eng. “parity”) carta.
 2. Dodajte šesti red karata za svaku kolonu tako da je ukupan broj obojenih karata u svakoj koloni paran.
 3. Sada okrenite neku kartu. Šta možete reći o redu i koloni gdje se ta karta nalazi? (I red i kolona imaju neparan broj obojenih karata.) Parne karte vam služe da pokažete kada je (tj. gdje je) greška napravljena.
 4. Sada zamijenite uloge da bi ponovo izveli ‘trik’.

Dodatne aktivnosti:

1. Pokušajte koristiti i neke druge stvari. Bilo šta što ima dva “stanja” je pogodno za trik. Na primjer, možete koristiti karte za igranje, novčiće (pismo glava) ili karte sa 0 i 1 odštampanim na njima (to bi najviše podsjećalo na binarni sistem).
2. Šta se događa ako se dvije, ili više, karata okrene? (Nije uvijek moguće odrediti tačno koje dvije karte su se okrenule iako je moguće reći da se nešto ipak promjenilo. Obično je moguće suziti sve mogućnosti na jedan ili dva para karata. Ipak sa 4 promjene moguće je da su se svi parni bitovi ispravili u procesu pa se onda greška ne može primjetiti.)
3. Pokušajte uraditi sve isto sa nekim većim kvadratom na primjer 9×9 karata, sa dodatnim redom i kolonom koji proširuju kvadrat na dimenzije 10×10 . (Sve u principu funkcioniše za bilo koju dimenziju koju izaberete, i dobiveni pravougaonik ne mora biti ni kvadratnog oblika).
4. Još jedan interesantan zadatak je da posmatramo kartu u donjem desnom uglu. Ukoliko je izaberete da bude dobro izabrana za kolonu iznad nje da li će biti korektno postavljena i za red na svojoj lijevoj strani? (odgovor je da, uvijek, ukoliko koristimo parnost za kolone i redove.)
5. U ovom zadatku sa kartama smo koristili parnost po redovima i kolonama—koristili smo paran broj obojenih karata. Da li je moguće isto uraditi sa neparnosti. (Moguće je, ali karta u donjem desnom uglu je korisna za odgovarajuću kolonu i odgovarajući red samo ako je broj i redova i kolona ili paran ili da su oba neparna. Na primjer, sve će u dimenzijama 5×9 funkcionišati sasvim dobro, ili sa dimenzijama 4×6 , ali neće na primjer pravougaonik dimenzija 3×4 .)

Jedan primjer iz stvarnog života za Eksperte!

Ista tehnika provjere koda se koristi za kodovo knjiga ili bar kodove. Javno objavljene knjige imaju ili 10- ili 13-cifreni kod koji se obično može naći na stražnjoj strani omota. Posljednja cifra je cifra provjere, upravo kao što je to parni bit u našem zadatku.

To znači da ukoliko naručite knjigu koristeći njen ISBN (eng. International Standard Book Number) broj, website može provjeriti zbir za provjeru (eng. checksum). Na taj način se neće dogoditi da čekate na isporuku pogrešne knjige!

Ovdje ćemo objasniti detalje provjere zbira za 10-cifreni ISBN kod neke knjige:

Pomnožimo prvu cifru sa deset, drugu sa devet, treću sa osam, i tako dalje sve dok ne pomnožimo devetu cifru sa dva. Onda se svaka od ovih vrijednosti sabere i dobijemo ukupan zbir.

Na primjer, ISBN 0-13-911991-4 nam daje vrijednost

$$\begin{aligned} & (0 \times 10) + (1 \times 9) + (3 \times 8) + (9 \times 7) + (1 \times 6) \\ & + (1 \times 5) + (9 \times 4) + (9 \times 3) + (1 \times 2) \\ & = 172 \end{aligned}$$

Nakon toga dobiveni broj podijelimo sa jedanaest. Šta je ostatak pri tom dijeljenju?

$$172 \div 11 = 15 \text{ ostatak } 7$$

Ukoliko je ostatak nula onda je cifra provjere (checksum) nula a inače oduzmimo ostatak od 11 da bi dobili cifru provjere.

$$11 - 7 = 4$$

Pogledajmo ponovo naš ISBN. Da li je to posljednja cifra našeg ISBN-a? Jeste!

Ukoliko posljednja cifra ISBN-a nije bila četiri onda bi znali sigurno da negdje postoji greška.

Moguće je prilikom računanja dobiti cifru provjere jednaku 10 što bi onda zahtjevalo još jednu dodatnu cifru (jedanaestu). Ukoliko se to dogodi onda ćemo koristiti slovo X (na primjer 0-9752298-0-X).



▲ Primjer bar-koda (UPC) sa jedne kutije Weet-Bix™

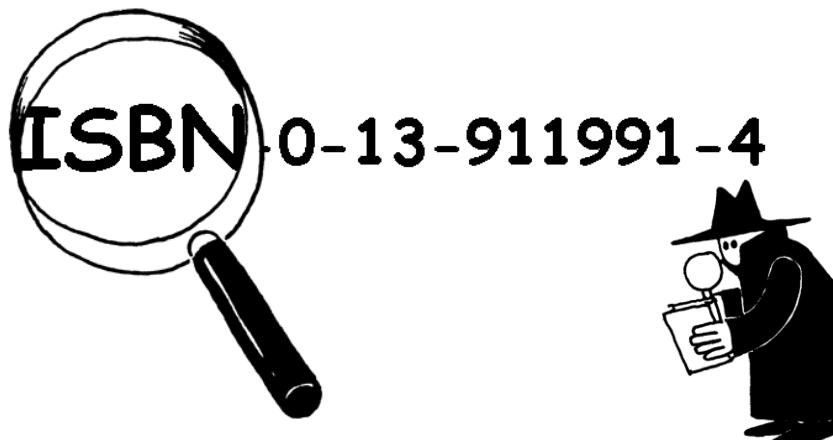
Drugi primjer kada koristimo cifru provjere je bar-kod na proizvodimo u nekom supermarketu. Ovaj sistem koristi jednu drugačiju formulu (ista formula se koristi za 13-cifreni ISBN kod za knjige). Ukoliko je neki bar-kod pogrešno pročitan (sa nekom greškom) posljednja cifra bi trebala biti različita od svoje izračunate vrijednosti. Kada se

to dogodi skener će dati signal (beep) i operator treba da ponovi operaciju skeniranja. Cifre provjere se koriste i za brojeve bankovnih računa, porezne brojeve, brojeve vozova kao i lokomotiva, kao i u mnogim drugim primjenama gdje ljudi moraju kopirati broj i gdje ima potreba za dodatnom provjerom tačnosti podataka.

Provjerite tu knjigu!

Detektiv Blockbuster

Servis za Praćenje Knjiga, d.o.o.



Pronalizimo i provjeravamo vaš ISBN I računamo zbir za provjeru.

Cijana sitnica.

Pridružite se našoj agenciji—provjerite u vašem razredu ili u školskoj biblioteci prave ISBN kodove.

Da li su ove cifre provjera dobre?

Ponekad se dogadjaju greške.

Neke od uobičajenih grešaka su:

- ✗ neka cifra je promjenila svoju vrijednost;
- ✗ dvije susjedne cifre su izmjenile svoja mesta;
- ✗ jedna cifra je ubaćena negdje u sredinu broja; i
- ✗ jedna cifra je izbaćena (greškom) iz broja

Da li možete pronaći jednu knjigu koja ima slovo X na mjestu cifre provjera (cifra provjera je jednaka 10)? To ne bi trebalo biti preteško jer jedna od 11 knjiga bi trebala imati slovo X u svom ISBN-u.

Koji tip grešaka se može dogoditi i koje bi prošle neopaženo? Možete li promjeniti samo jednu cifru ali još uvijek imati tačnu cifru provjere? Šta se događa ukoliko dvije cifre zamjene mjesta (uobičajena greška prilikom kucanja)?

Zašto je ovo sve važno?

Zamislite da posjedujete 10 KM gotovine na vašem bankovnom računu. Službenik kuca sumu na vašem računu i šalje je u centralni računar. Sada pretpostavimo da se dogodila neka nezgoda na komunikacionoj liniji i kod za 10 KM se promjenio u kod za 1000 KM. Nema problema za vas ako ste mušterija banke ali je sigurno problem za banku!

Važno je prepoznati greške koje se mogu dogoditi prilikom prenosa podataka. Prema tome računar koji preuzima podatke mora provjeriti da podaci koji dolaze nisu korumpirani na neki način zbog recimo elektronske interferencije na mreži. Ponekad je moguće ponovo poslati orginalne podatke kada se detektuje neka greška prilikom prenosa ali postoje situacije kada to nije moguće, na primjer kada je disk korumpiran zbog izloženosti magnetnom ili elektro zračenju, kada je pretrpio štetu zbog pregrijavanja ili jednostavno fizičkog oštećenja. Ukoliko su podaci dobiveni sa neke sonde iz dubokog okeana ili iz svemira bilo bi prilično neugodno čekati ponovo slanja kada se primjeti greška u podacima! (Potrebno je upravo pola sata da dobijemo radio signal sa Jupitera kada se on nalazi u svojoj najbližoj tački planeti Zemlja!)

Potrebno je da smo u stanju prepoznati kada su podaci korumpirani (otkrivanje greške, eng. error detection) kao i da smo u stanju ispraviti grešku i dobiti orginalne podatke (otkrivanje greške, eng. error correction).

Potpuno ista tehnika koju smo koristili u igri "okretanje karata" se koristi i u računarama. Stavljući bitove u zamišljene redove i kolone, i dodajući bit parnosti u svaki red i u svaku kolonu, moguće je ne samo otkriti da li je došlo do neke greške već i odrediti gdje se ta greška dogodila. Promjenjeni bit, onaj koji pravi problem, se može vratiti na svoju originalnu vrijednost tako da možemo reći da smo uradili i ispravljanje greške.

Naravno da računari često koriste složenije sisteme za provjeru grešaka koji su u mogućnosti otkriti i ispraviti više grešaka u isto vrijeme. Of course computers often use more complex error control systems that are able to detect and correct multiple errors. Na primjer, tvrdi disk u računaru veliki dio svog memorijskog prostora rezerviše za ispravljanje grešaka tako da će raditi prilično sigurno i tačno čak i ako neki dijelovi diska budu potpuno uništeni. Sistem koji se koristi da bi se ovo postiglo je u uskoj vezi sa schemom parnosti koju smo upravo vidjeli.

I da završimo sa jednom šalom kako bi više cijenili aktivnost koju smo uradili (šalu ćemo ostaviti na engleskom):

Q: What do you call this: "Pieces of nine, pieces of nine"?

A: A parrot error.



Rješenja i pomoć

Greške koje se ne bi otkrile sa ISBN-10 cifrom provjere su one gdje se jedna cifra smanji a druga se poveća i ove se promjene pokušavaju poništiti. Tada dobivena suma može ostati ista. Ipak, zbog načina na koji se cijeli izračun radi, malo je vjerovatno da će se to dogoditi. U nekim drugim sistemima (na primjer ISBN-13) postoje neki drugi tipovi grešaka koji se ne moraju otkriti, na primjer ona kada se tri uzastopne cifre okrenu, ali se najveći broj uobičajenih grešaka (pogrešno kucanje jedne cifre, ili izmjena dvije susjedne cifre) lagano otkrije.

Aktivnost 5

Dvadeset Pogadanja—Teorija *Informacija*

Sažetak

Koliko informacija ima u jednoj knjizi od 1000 stranica? Da li ima više informacija u telefonskom imeniku od 1000 stranica, ili u jednom paketu od 1000 praznih bijelih listova, ili možda u Tolkien-ovom romanu Gospodar prstenova? Ako bi mogli izmjeriti i uporediti ove informacije onda bi mogli procijeniti koliko nam je prostora potrebno da bi ih čuvali. Na primjer, da li ste još uvijek u stanju da pročitate sljedeću rečenicu?

Jdn rcnc gdj ndstj smglsnc.

Vjerovatno možete dokučiti značenje rečenici jer nema mnogo ‘informacija’ u samim samoglasnicima. Ova aktivnost nam daje kratak uvod u način mjerena sadržaja informacija.

Veze sa Curriculum-om

- ✓ Matematika: Broj – Istraživanje brojeva: veći od, manji od, domen.
- ✓ Matematika: Algebra – Uzorci i nizovi
- ✓ BHS jezik: pisanje, prepoznavanje elemenata jednog teksta

Vještine

- ✓ Poređenje brojeva i rad sa domenima (intervalima) brojeva
- ✓ Zaključivanje dedukcijom
- ✓ Postavljanje pitanje

Dobna/starosna grupa

- ✓ 10 godina i više

Materijal

- ✓ Nije potreban nikakav materijal za prvu aktivnost

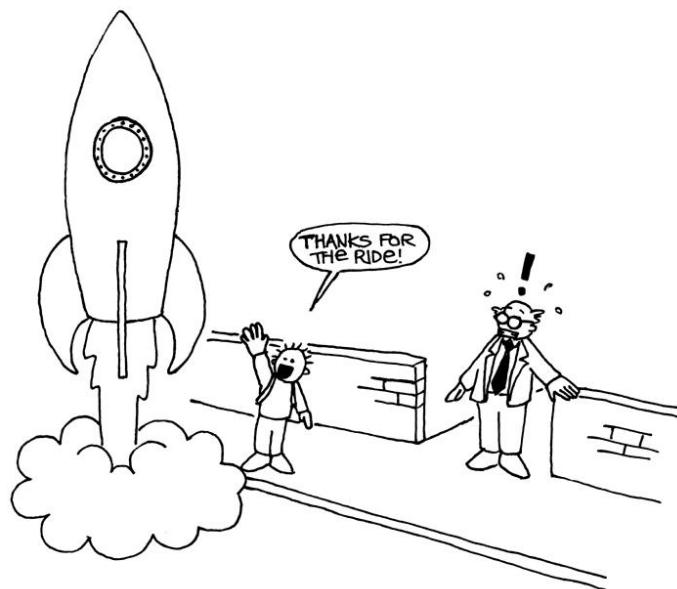
Ima jedna dodatna aktivnost za koju će učenici trebati :

List za aktivnost: stabla odlučivanja (strana 44)

Dvadeset pogadanja

Diskusija

1. Diskutujte sa učenicima o tome šta oni misle da je informacija.
2. Kako bi mogli izmjeriti količinu informacija koje postoje u jednoj knjizi? Da li je važniji broj riječi ili broj stranica teksta? Da li jedna knjiga može imati više informacije nego neka druga knjiga? Šta ako je to neka jako dosadna knjiga, ili, s druge stane, jedna vrlo interesantna knjiga? Da li će knjiga od 400 stranica koja sadrži samo rečenicu “bla, bla, bla” imati više ili manje informacija nego, na primjer, jedan telefonski imenik?
3. Objasnite da naučnik informatičar mjeri količinu informacija prema tome kako je poruka (ili knjiga!) iznenađujuća. Reći vam nešto što već znate—na primjer, kada vam prijatelj koji svaki dan ide pješke u školu kaže “Danas sam došao pješke u školu”—onda vam to ne daje nikakvu informaciju jer vas ne iznenađuje. Ako vam sada taj prijatelj kaže, “Danas sam se dovezao helikopterom u školu,” onda je to vrlo iznenađujuće, i prema tome vam daje mnogo informacija.
4. Na koji način se može izmjeriti iznenađenje u jednoj poruci?
5. Jedan od načina bi bio da provjerimo koliko je teško pogoditi tu informaciju. Ako vaš prijatelj kaže, “Pogodi kako sam danas došao u školu,” a pri tome je taj dan došao pješke onda ćete to vjerovatno pogoditi iz prvog puta. Moguće je da će vam trebati nekoliko pokušaja više nego što bi pogodili da je to bio helikopter, a onda i nekoliko više ako bi se radilo o svemirskom brodu.
6. Količina informacija koje sadrži jedna poruka se mjeri koliko je lagano ili teško pogoditi tu informaciju. Sljedeća igra nam objašnjava ove pojmove malo bolje.



Aktivnost za Dvadeset Pitanja

Ovo je jedna prilagođena verzija igre od 20 pitanja. Učenici mogu postavljati pitanja izabranom učeniku koji može odgovarati samo sa da ili ne sve dok se odgovor na pitanje ne pogodi. Moguće je postaviti bilo koje pitanje pod uslovom da je odgovor strogo ‘da’ ili ‘ne’.

Prijedlozi:

Razmišljaj o sljedećim stvarima:

- ✓ broj između 1 i 100
- ✓ broj između 1 i 1000
- ✓ broj između 1 i 1,000,000.
- ✓ bilo koji cijeli broj
- ✓ niz od 6 brojeva uzetih po nekom pravilu ili uzorku (pravilo treba prilagoditi grupi/razredu sa kojim se radi). Pogadati po redu od prvog prema zadnjem. (na primjer 2, 4, 6, 8, 10)

Prebrojte ukupan broj pitanja koja su postavljena. Ovaj broj mjeri vrijednost “informacije”.

Diskusija nakon vježbe

Koje sve strategije ste koristili? Koja je bila najbolja strategija?

Pokažite da je potrebno samo 7 pogadanja da bi našli neki broj između 1 i 100 ako bi polovili interval (domen) svaki sljedeći put. Na primjer:

Da li je manji od 50?	Da.
Da li je manji od 25?	Ne.
Da li je manji od 37?	Ne.
Da li je manji od 43?	Da.
Da li je manji od 40?	Ne.
Da li je manji od 41?	Ne.
Onda mora biti 42!	Da!

Interesantno je da ako bi povećali interval na 1000 onda se broj pitanja ne povećava 10 puta—samo tri dodatna pitanja su potrebna za pogadanje broja. Svaki put kada se interval udupla vama treba samo jedno pitanje više da nađete odgovor.

Dobar prijedlog učenicima za nastavak ove aktivnosti je igra Mastermind.

Dodatna aktivnost: Koliko informacija ima u jednoj poruci?

Naučnici-informatičari ne koriste pogadanje samo za brojeve—oni mogu predvidjeti ili dobro pogadati koje slovo ima više šanse da bude na sljedećem mjestu u nekoj riječi ili rečenici.

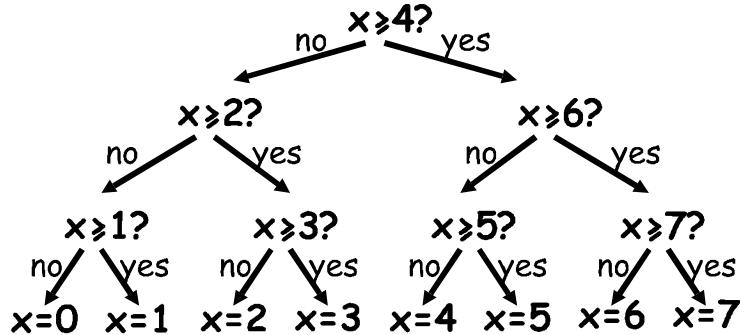
Pokušajte ovu igru pogadanja sa nekim kratkim rečenicama od 4–6 riječi. Slova moraju biti pogadana u svom rasporedu u rečenici, od prvog do posljednjeg. Izaberite nekoga ko će zapisivati slova onda kada se ona pronađu i pogode i bilježite koliko pogadanja je bilo potrebno za svako slovo. Bilo koje pitanje sa da/ne odgovorom je dozvoljeno. Primjeri pitanju su “Da li je to *t*?” “Da li je to samogradnik?” “Da li dolazi prije slova *m* u abecedi?” Razmak između dvije riječi u rečenici se takođe broji kao “slovo” i mora biti

pogađano. Na kraju analizirajte šta se događalo tokom igre i otkrijte koji dijelovi poruke su bili najlakši za pronalaženje.

Radni List za Aktivnost: Stabla odlučivanja

Ako već poznajete strategiju za postavljanje pitanja onda možete prenijeti neku poruku bez potrebe da postavljate bilo kakva pitanja.

Ovdje je dijagram koji nazivamo ‘stablu odlučivanja’ koji koristimo za pograđanje broja između 0 i 7:



Koje da/ne odluke su potrebne da bi ‘pogodili’ broj 5?

Koliko da/ne odluka vam je potrebno da bi mogli “pogoditi” bilo koji broj?

Sada obratimo pažnju na nešto zaista fascinantno. Ispod brojeva 0, 1, 2, 3... u posljednjem redu stabla napišimo te brojeve u binarnom zapisu (pogledati Aktivnost 1).

Pogledajmo sada pažljivo stablo. Ako stavimo da je ne=0 a da je da=1, šta možemo zaključiti?

U igri pograđanja broja pokušavali smo izabrati pitanja tako da niz odgovora koje dobijemo tokom pograđanja broja nam daje upravo i reprezentaciju tog broja u binarnom sistemu.

Kreirajte vaše sopstveno stablo odlučivanja za brojeve između 0 i 15.

Ekstra za eksperte: Kakav tip stabla odlučivanja bi koristili da pogodite nečiju starost u godinama?

Šta bi bilo stablu odlučivanja za pograđanje sljedećeg slova u rečenici?

Zašto je ovo sve važno?

Proslavljeni američki matematičar (i žongler kao i vozač uniklka) po imenu Claude Shannon je uradio mnoge eksperimente koristeći ovu igru. On je mjerio količinu informacija u broju bitova—svaki da/ne odgovor odgovara jednom 1/0 bit. Claude je otkrio da količina “informacija” sadržanih u jednoj poruci zavisi od toga šta neko već zna. Nekada možemo postaviti jedno pitanje čiji odgovor će eliminisati potrebu za postavljanjem mnogih drugih pitanja. U tom slučaju je informacija sadržana u poruci mala. Na primjer, informacija u jednom bacanju novčića je jedan bit: pismo ili glava. Ali ako je novčić trikovan tako da daje glavu u devet od deset pokušaja onda informacija nije više jedan bit—vjerovali vi ili ne, onda je manje od bita. Kako je moguće znati šta je rezultat bacanja novčića sa manje od jednog da/ne pitanja? Jednostavno—dovoljno je koristiti pitanja kao što su “da li su oba naredna bacanja glava?” Za niz bacanja trikovanog novčića, odgovor na ovo pitanje će biti “da” u oko 80% slučajeva. U ostalih 20% slučajeva kada je odgovor “ne,” morate postaviti još neka pitanja. Ali ćete prosječno pitati manje od jednog pitanja za svako bacanje novčića!



Shannon je nazvao sadržaj informacije u jednoj poruci “entropija” (eng. entropy). Entropija zavisi ne samo od broja mogućih rezultata—u slučaju bacanja novčića to je dva,—već i od vjerovatnoća da se neki rezultat dogodi. Malo vjerovatni događaji, ili izmenadjuće informacije, trebaju mnogo više pitanja da bi se pogodio sadržaj poruke jer nam govore, daju više informacija nego što smo već znali—upravo kao sa situacijom kada dolazite u školu helikopterom.

Entropija jedne poruke je jako važna za naučnike-informatičare. Nije moguće kompresovati neku poruku tako da koristite manje prostora nego što je njena entropija, pa je onda najbolji način kompresije obično ekvivalentan igri pogađanja. I dok računarski program “pogada”, niz pitanja se može reproducirati kasnije pa ako su odgovori (bitovi) sačuvani onda je moguće i rekonstruisati cijelu informaciju! Najbolji sistemi kompresije mogu reducirati neki tekst na četvrtinu njegove prvobitne veličine—što je velika ušteda na memoriskom prostoru!

Metod pogađanja se može koristiti kao osnova za pravljenje računarskog sučelja koje će pretpostaviti šta će korisnik ukucati sljedeće! Ovakav sistem može biti jako koristan za fizički hendikepirane osobe koje imaju poteškoće prilikom kucanja teksta. Računar predlaže šta misli da bi korisnik mogao htjeti ukucati sljedeće i onda korisnici izaberu šta stvarno žele. Jedan dobar sistem treba u prosjeku samo dva da/ne odgovora po slovu, i može biti od velike pomoći za nekoga ko ima poteškoće da uradi fine pokrete neophodne za efikasnu kontrolu miša i tastature. Ista vrsta programa se koristi i za olakšavanje “kucanja” teksta ne velikom broju smartphone-a.

Rješenja i pomoć

Odgovor na jedno da/ne pitanje odgovara tačno jednom bitu informacija—bez obzira da li je to jednostavno pitanje kao što je “Da li je veći od 50?” ili neko složenije pitanje tipa “Da li je broj između 20 i 60?”

U igri pogađanja brojeva, pitanja su izabrana na jedan određen način i niz odgovora je upravo binarna reprezentacija broja. Tri je 011 u binarnom zapisu i predstavljen je odgovorima “ne, da, da” u stablu odlučivanja, što je isto ako bi zapisali ne za 0 i da za 1.

Stablo koje bi koristili da pogodite nečiju starost moglo bi biti naklonjenije, to jest razvijenije, za odgovore sa malim brojevima.

Odluka o slovima u rečenici može zavisiti u dobroj mjeri od toga koje je bilo prethodno slovo.

DIO II

Kako računari rade svoj posao—
Algoritmi

Kako računari rade svoj posao

Računari obavljaju neki zadatak prateći i izvršavajući korak po korak unaprijed dati niz naredbi, instrukcija. Ove naredbe omogućavaju računarima da sortiraju, pronalaze i šalju informacije. Da bi svi ovi zadaci bili obavljeni na najbrži mogući način potrebno je raspolagati sa dobrom metodama za pronalaženje određenih stvari u velikim skupinama podataka, kao i za brzo slanje informacija preko telekomunikacionih mreža.

Jedan *algoritam* je skup naredbi za izvršenje jednog zadatka. Ideja i pojam algoritma ima centralno mjesto u računarskim naukama. Algoritmi omogućavaju računarim da riješavaju probleme koje im zadamo. Neki algoritmi su brži, efikasniji od drugih. Mnogi otkriveni algoritmi nam omogućavaju da sada riješavamo probleme za čije rješavanje je prije trebalo mnogo, mnogo vremena – na primjer, nalaženje miliona cifara broja pi, ili nalaženje svih stranica na World-Wide Web koje sadrže vaše ime, ili nalaženje najboljeg načina pakiranja paketa u transportni kontejner, ili odgovoriti na pitanje da li je neki broj sa 100 cifara prost ili nije.

Same riječ “algoritam” dolazi od imena istraživača Mohammed ibn Musa Al-Khowarizmi—Mohammed, sin Musin, iz Khowarizm-a—koji se priključio akademskom naučnom centru poznatom pod imenom Kuća mudrosti u Bagdadu oko 800 n.e. Njegovi radovi su prenijeli indijsko umijeće računanja prvo Arapima pa dakle nakon toga i u Evropu. Kada su bili prevedeni na latinski godine 1120 n.e., prve dvije riječi su bile “Dixit Algorismi”—“tako je govorio Algorismi”.

Activity 6

Aktivnost 6

Ratni brodovi—Algoritmi pretraživanja

Sažetak

Računari često moraju naći informacije u velikim skupinama podataka. Prema tome potrebno je razviti brze i efikasne načine da se to uradi. Ova aktivnost pokazuje tri različita metoda pretraživanja: linearno pretraživanje, binarno pretraživanje i pretraživanje raspršivanjem (eng. hashing).

Veze sa Curriculum-om

- ✓ Matematika: Brojevi – Istraživanje brojeva: Veći od, manji od i jednak broju
- ✓ Matematika: Geometrija – Istraživanje oblika i prostora: Koordinate
- ✓ Računarske nauke: Algoritmi

Vještine

- ✓ Logičko razmišljanje

Dobna/starosna grupa

- ✓ 9 godina i više

Materijal

Svaki učenik će trebati:

- ✓ Kopiju igara Ratnih brodova
 - 1A, 1B za igru 1
 - 2A, 2B za igru 2
 - 3A, 3B za igru 3
- ✓ Vi sami kao nastavnik ćete trebati nekoliko dodatnih kopija listova za igru 1A', 1B', 2A', 2B', 3A', 3B'.

Ratni brodovi

Uvodna aktivnost

1. Izaberite 15-ak učenika i poredajte ih ispred razreda. Dajte svakom učeniku kartu sa brojem (i to u slučajnom redoslijedu). Tražite od učenika da ne pokazuju i kriju brojeve od ostatka razreda.
2. Sada dajte jednom učeniku posudu sa četiri ili pet slatkiša u njoj. Njegov zadatak je da pronađe jedan dati broj. Ovaj učenik ima mogućnost da “plati” kako bi pogledao određenu kartu. Ukoliko pronađu tačan broj prije nego potroše sve slatkiše koje posjeduju onda mogu zadržati ostatak.
3. Igru možete ponavljati nekoliko puta.
4. Sada pomješajte karte i podijelite ih još jednom svima. Ovaj puta, tražite od učenika da se poredaju, sortiraju u rastućem redoslijedu. Proces pretraživanje se ponavlja.

Ukoliko su brojevi sortirani, pametan način bi bio da se koristi samo jedno “plaćanje” da bi se eliminiralo pola učenika samo tako što će učenik u sredini pokazati svoju kartu. Ponavljajući isti postupak i sa preostalom polovinom svako bi trebao biti u stanju pronaći traženi broj koristeći samo tri slatkiša. Povećana efikasnost će biti jasna svima.

Aktivnost

Učenici mogu osjetiti i razumjeti na koji način računari rade pretrage igrajući igru ratnih brodova. Dok igraju igru tražite od njih da razmišljaju o strategiji koju koriste za lociranje brodova.

Ratni brodovi—Igra linearog pretraživanja

Pročitajte data uputstva učenicima

1. Rasporedite se u parove. Jedna osoba u paru ima list 1A, a druga list 1B. Ne pokazujte svoj list svom partneru!
2. Svako će zaokružiti jedan ratni broj na gornjoj liniji vašeg lista za igranje i saopštiti svom partneru pripadajući broj.
3. Sada naizmjenično pogadajte gdje se nalazi brod vašeg para. (Vi ćete reći slovno ime broda a vaš par će vam reći koji je broj na brodu koji odgovara datom slovu.)
4. Koji je ukupan broj pucanja potreban da bi locirali ratni brod vašeg partnera ? Taj broj je onda vaš rezultat za ovu igru.

(Listovi 1A' i 1B' su dodani za učenike koji bi željeli da igraju više igara ili za one koji “slučajno” vide list svog para. Listovi 2A', 2B' i 3A', 3B' su tu za igre koje slijede.)

Diskusija nakon aktivnosti

1. Koji su rezultati postignuti?
2. Koji bi bio najmanji a koji najveći mogući rezultat? (Odgovori su 1 i 26, pod pretpostavkom da učenici ne pucaju dva puta na isti brod. Ovaj metod se naziva ‘linearno pretraživanje’, jer podrazumjeva prolazak kroz sve moguće pozicije, jednu po jednu.

Ratni brodovi—Igra binarnog pretraživanja

Uputstva

Uputstva za ovu verziju igre su ista kao i za prethodnu igru ali su sada brojevi na brodovima dati u rastućem redoslijedu. Objasnite to učenicima prije nego što igra počne.

1. Podijelite se ponovo u parove. Jedna osoba u paru posjeduje list 2A, a druga posjeduje list 2B. **Ne pokazujte svoj list svom paru!**
2. Svako od vas će zaokružiti jedan ratni brod na gornjoj liniji svog lista za igru i saopštiti ga partneru.
3. Sada naizmjenično pokušajte pronaći gdje se nalazi brod vašeg partnera. (Vi ćete reći slovno ime broda a vaš par će vam reći koji je broj na brodu koji odgovara datom slovu.)
4. Koji je ukupan broj pucanja potreban da bi locirali ratni brod vašeg partnera ? Taj broj je onda vaš rezultat za ovu igru.

3.

Diskusija nakon igre

1. Koji su rezultati postignuti?
2. Koja strategija pretraživanja omogućava najniži rezultat?
3. Koji brod bi vi izabrali za prvu metu? (Brod u sredini vam govori u kojoj polovini reda se nalazi traženi brod.) Koju poziciju bi izabrali u sljedećem pokušaju?(Ponovo, najbolja strategija je uvijek izabrati brod u sredini dijela za koji znate da sadrži traženi brod.)
4. Ukoliko se primjeni ovakva strategija koliko pokušaja je potrebno da bi pronašli brod (najviše pet).

Ovaj metod se naziva ‘binarno pretraživanje’, jer dijeli problem na dva njegova dijela.

Ratni brodovi—Igra raspršenog (hashing) pretraživanja

Uputstva

1. Svako uzme po jedan list kao u prethodnim igrama i saopšti svom partneru broj izabranog broda.
2. U ovoj igri možete pronaći kolonu (od 0 do 9) u kojoj se nalazi brod. Jednostavno saberete cifre broja izabranog broda. Posljednja cifra dobivene sume je kolona gdje se nalazi brod. Na primjer, da bi odredili kolonu gdje se nalazi brod 2345 saberećete sve cifre $2+3+4+5$, što daje 14. Posljednja cifra sume je 4 pa se tako traženi brod mora nalaziti u koloni 4. Nakon što ste odredili u kojoj koloni se nalazi traženi brod treba odrediti koji od brodova u toj koloni je traženi brod. Ova tehnika se naziva tehnika raspršivanja (eng. Hashng) jer su cifre isjeckane(raspršene, razbijene) pa pomješanje sve zajedno.
3. Sada igrajte istu igru koristeći novu strategiju pretraživanja. Moguće je da želite da igrate više igara koristeći isti list—jednostavno izaberite svoje brodove iz različitih kolona.

(Primjetimo da, za razliku od ostalih igara, dodatni listovi 3A' i 3B' moraju biti korišteni kao par, jer raspored brodova u kolonama mora odgovarati.)

Diskusija nakon aktivnosti

1. Sakupite i razmotrite postignute rezultate kao i ranije.
2. Koji brodovi se brzo i lagano pronalaze? (Oni koji su jedini u svojoj koloni.) Koji brodovi bi mogli biti teži za pronalaženje? (Oni u čijim pripadajućim kolonama ima mnogo brodova.)
3. Koja od tri tehnike pretraživanja je najbrža? Zašto?

Koje su prednosti svake od tri razmatrane tehnike pretraživanja? (Druga strategija je brža nego prva, ali prva ne zahtjeva da su brodovi sortirani u nekom redoslijedu (rastućem ili opadajućem). Treća strategija je obično brža nego prve dvije, ali je moguće, da u nekom nesretnom slučaju, bude jako spora. U najgorem slučaju, ako svi brodovi završe u istoj koloni onda će biti spora upravo kao i prva strategija.)

Dodatne Aktivnosti

1. Zatražite od učenika da naprave svoje primjerke listova za sve tri igre. Za drugu igru moraju složiti pripadajuće brojove brodova u rastućem redoslijedu. Pitajte ih kako bi mogli napraviti Igru raspršivanja jako teškom za igranje. (Najteža igra je ona kada su svi brodovi u istoj koloni.) Kako možete napraviti istu igru najlakšom za igranje? (Trebate pokušati imati isti broj brodova u svakoj koloni.)
2. Šta bi se dogodilo ako brod koji tražite nije uopšte na listu? (U slučaju Linearnog pretraživanja potrebno je ukupno 26 pokušaja da bi ovo dokazali. U Binarnom pretraživanju potrebno je pet pokušaja da bi ovo dokazali. Što se tiče Raspršenog pretraživanja ovaj broj zavisi od toga koliko brodova ima u odgovarajućoj koloni.)
3. U slučaju kada koristimo Binarno pretraživanje koliko je pokušaja potrebno ako bi bilo ukupno 100-njak pozicija (oko šest pokušaja), a oko 1000 lokacija (oko devet pokušaja), a onda ako bi imali oko milion lokacija (oko devetnaest pokušaja)? (Primjetimo da broj pokušaja raste jako sporo u odnosu na broj brodov. Samo jedan dodatni pokušaj je potreban kada se broj brodova udupla pa je prema tome broj pokušaja proporcionalan logaritmu ukupnog broja brodova.)

My Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
9058	7169	3214	5891	4917	2767	4715	674	8088	1790	8949	13	3014
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Your Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
8311	7621	3542	9264	450	8562	4191	4932	9462	8423	5063	6221	2244
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

1A

My Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
1630	9263	4127	405	4429	7113	3176	4015	7976	88	3465	1571	8625
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Your Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
2587	7187	5258	8020	1919	141	4414	3056	9118	7117	7021	3076	33336
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

1B

My Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
163	445	622	1410	1704	2169	2680	2713	2734	3972	4208	4871	5031
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Your Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
5283	5704	6025	6801	7440	7542	7956	8094	8672	9137	9224	9508	9663
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

2A

My Ships

Number of Shots Used:

-33	-183	-730	-911	-1927	-1943	-2200	-2215	-3451	-3519	-4055	-5548	-5655
A	B	C	D	E	F	G	H	I	J	K	L	M
5785	5897	5905	6118	6296	6625	6771	6831	7151	7806	8077	9024	9328
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Your Ships

Number of Shots Used:

-33	-183	-730	-911	-1927	-1943	-2200	-2215	-3451	-3519	-4055	-5548	-5655
A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

2B

3A

My Ships

Number of Shots Used:

	0	1	2	3	4	5	6	7	8	9	
A	9047	C	3080	E	5125	H	8051	I	1481	L	7116
B	1829	D	9994	F	1480	J	4712	K	6422	M	8944
				G	8212	N	4128	Q	4110	P	7432
						R	9891	S	1989	V	4392
						O	6000	T	2050	U	8199
								W	1062	X	2106
								Y	5842	Z	7057

Your Ships

Number of Shots Used:

	0	1	2	3	4	5	6	7	8	9
A										
B										
C										
D										
E										
F										
G										
H										
I										
J										
K										
L										
M										
N										
O										
P										
Q										
R										
S										
T										
U										
V										
W										
X										
Y										
Z										

3B

My Ships

Number of Shots Used:

	0	1	2	3	4	5	6	7	8	9
A	9308	E	6519	H	1524	L	9050	R	3121	V
B	1478	F	2469	I	8112	M	1265	S	9503	W
C	8417	G	5105	J	2000	N	5711	T	1114	X
D	9434							U	7019	Z

Your Ships

Number of Shots Used:

	0	1	2	3	4	5	6	7	8	9
A					H		R			V
B					I		S			W
C					J		T			X
D					K		U			Z
E						L				
F						M				
G						N				
							O			
							P			
							Q			

My Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
6123	1519	9024	5164	2038	2142	7156	9974	9375	7104	1004	1023	5108
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1884	3541	5251	4840	3289	3654	2480	5602	8965	4053	2405	2304	1959

Your Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
6123	1519	9024	5164	2038	2142	7156	9974	9375	7104	1004	1023	5108
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

1A'

My Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
2387	9003	3951	5695	1284	4761	7118	1196	1741	3791	3405	3132	6682
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
9493	9864	7359	1250	7036	2916	7562	9299	8910	6713	5173	8617	4222

Your Ships

Number of Shots Used:

A	B	C	D	E	F	G	H	I	J	K	L	M
.....
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

1B'

My Ships

	Number of Shots Used:												
	A	B	C	D	E	F	G	H	I	J	K	L	M
N	28	326	943	1321	1896	2346	2430	2929	3106	3417	4128	4717	4915
O	5123	5615	6100	7015	7120	7695	7812	8103	8719	9020	9608	9713	9911
P													
Q													
R													
S													
T													
U													
V													
W													
X													
Y													
Z													

Your Ships

	Number of Shots Used:												
	A	B	C	D	E	F	G	H	I	J	K	L	M
N	28	326	943	1321	1896	2346	2430	2929	3106	3417	4128	4717	4915
O	5123	5615	6100	7015	7120	7695	7812	8103	8719	9020	9608	9713	9911
P													
Q													
R													
S													
T													
U													
V													
W													
X													
Y													
Z													

2A'

My Ships

Number of Shots Used:

56	194	306	1024	1510	1807	2500	2812	3011	3902	4178	5902	5915
A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

Your Ships

Number of Shots Used:

56	194	306	1024	1510	1807	2500	2812	3011	3902	4178	5902	5915
A	B	C	D	E	F	G	H	I	J	K	L	M
N	O	P	Q	R	S	T	U	V	W	X	Y	Z

2B'

My Ships

Number of Shots Used:

Your Ships

Number of Shots Used:

3A'

3B'

My Ships

Number of Shots Used:

0	1	2	3	4	5	6	7	8	9
A \ 8615	E \ 1361	H \ 7726	I \ 9003	K \ 3000	L \ 1814	O \ 9656	R \ 6993	V \ 8208	Y \ 2917
B \ 7003	F \ 7644	G \ 5600	J \ 5557		M \ 2002	P \ 4002	S \ 3121	W \ 9423	Z \ 4122
C \ 1991					N \ 8844	Q \ 1221	T \ 4300	X \ 4176	
D \ 6211							U \ 1907		

Your Ships

Number of Shots Used:

0	1	2	3	4	5	6	7	8	9
A	C	E	H	I	L	O	R	S	W
B	D	F	G	J	K	M	N	P	X
							V	T	Y
							U		Z

Zašto je ovo sve važno?

Računari čuvaju i rade sa velikim količinama podataka i moraju biti u stanju prolaziti i pretraživati ove podatke na brz i efikasan način. Jedan od najvećih problema pretraživanja u svijetu danas je pretraživanje podataka na cijelom Internetu. Potrebno je pretražiti milijarde web stranica u samo jednom djeliću sekunde. Pretraživači danas dosta dobro rješavaju ovaj problem. Kada tražimo od računara da pronađe jedan podatak kao što je riječ, bar kod broj ili ime autora onda to nazivamo *ključem pretraživanja* (eng. *search key*).

Računari mogu obrađivati informacije na vrlo brz način, i u prvi trenutak bi mogli pomisliti da računari da bi našli neki podatak samo počnu pretragu na početku memorije gdje čuvaju podatke i nastave pretraživati redom sve dok ne nađu traženu informaciju. To je ono što smo i sami radili igrajući Igru linearog pretraživanja. Na žalost, ovaj metod je vrlo spor—čak i za računare. Na primjer, pretpostavimo da neka velika samoposluga ima 10,000 različitih proizvoda na svojim policima. Kada na izlazu prilikom plaćanja skeniramo bar kod računar mora pregledati do 10,000 brojeva da bi našao ime proizvoda i pripadajuću cijenu. Čak i ako bi nam trebalo samo hiljaditi dio sekunde da provjerimo svaki kod potrebno bi bilo ukupno 10 sekundi da bi pregledali cijelu listu. Sada zamislimo koliko bi vremena trebalo ako bi htjeli pronaći imena svih proizvoda tokom jedne velike porodične kupovine!

Bolja strategija je binarno pretraživanje. U ovom metodu, brojevi su sortirani u nekom redoslijedu. Provjera vrijednosti elementa u sredini će odrediti u kojoj polovini podataka se ključ pretraživanja nalazi. Proces se ponavlja sve dok se ne pronađe traženi elemenat. Ako ponovo pogledamo primjer sa samoposlugom i 10,000 proizvoda vidjećemo da je potrebno samo četrnaest provjera, što bi uzelo samo dvije stotinke sekunde—vremenski interval koji bi jako teško primjetili.

Treća strategija za pronađenje podataka se naziva *raspršivanje (hashing)*. U ovom slučaju, ključ pretraživanja je obrađen na određeni način kako bi imali dodatnu informaciju gdje se traženi elemenat nalazi. Na primjer, ako je ključ pretraživanja telefonski broj, možete sabrati sve cifre datog broja i onda uzeti ostatak pri djeljenju sa 11 dobivenog zbira. U ovom svjetlu, hash ključ je nekako najbliži cifri provjere koju smo već vidjeli i diskutovali u Aktivnosti 4—jedan mali podatak čija vrijednost zavisi od podataka koje obrađujemo. Uobičajeno je da računar koristeći ovu tehniku pronađe traženu informaciju odmah nakon izračunavanja hash ključa. Ipak, postoji malo vjerovatnoća da će više različitih ključeva završiti na istoj lokaciji i u tom slučaju računar mora pretražiti svaki elemenat sve dok ne nađe onaj koji traži.

Programeri obično koriste neki oblik strategije raspršivanja za pretraživanje, osim u slučaju kada je važno imati sve podatke u nekom redoslijedu ili ukoliko, slučajan ali moguć, spori odgovor nije prihvatljiv (na primjer, u slučaju računara koji upravlja avionom).

Aktivnost 7

Najlakši i Najteži—Algoritmi sortiranja

Sažetak

Računari se često koriste kako bi složili neku listu u neki traženi redoslijed. Na primjer, imena treba složiti u alfabetском redoslijedu, sastanke ili e-mail-ove prema njihovom datumu, ili neke elemente u numeričkom redoslijedu. Sortirane listen nam omogućavaju da nađemo stvari na brži način kao što omogućavaju da lagano pronađemo ekstremne, krajnje vrijednosti. Ako na primjer sortirate sve ocjene u jednom razredu onda se najveća i najmanja ocjena same ističu.

Ukoliko izaberete pogrešan metod za sortiranje vrlo je moguće da će samo sortiranje uzeti previše vremena čak i na nekom super-brzom računaru. Tokom ove aktivnosti učenici će se upoznati i otkriti različite metode za sortiranje, i uvjeriti se sami kako mudro izabrani metod može obaviti zadatku mnogo brže nego neki jednostavan metod.

Veze sa curriculum-om

- ✓ Matematika: Mjerenja – Vježba praktičnog mjerenja težina.
- ✓ Izračunavanja: Algoritmi

Vještine

- ✓ Korištenje vase
- ✓ Uređivanje i sortiranje
- ✓ Poređenje
- ✓

Dobna/starosna grupa

- ✓ 8 godina i više

Materijal

Svaka grupa učenika će trebati:

- ✓ Komplet od 8 posuda iste veličine ali različitih težina (na primjer, čaše za jogurt ili tetrapak za mlijeko ispunjeni pijeskom)
- ✓ Balasna (pijačna) vaga
- ✓ Radni list za aktivnost: Sortiranje težina (strana 72)
- ✓ Radni list za aktivnost: Podijeli i vladaj (strana 72)

Najlakši i Najteži

Diskusija

Računari često moraju sortirati datu listu stvari u neki redoslijed. Razgovarajte na neobavezan način (brainstorm) sa učenicima gdje je sve važno imati sortirane objekte. Šta bi se dogodilo ako ovi objekti ne bi bili sortirani?

Računari obično mogu upoređivati samo dvije stvari u jednom trenutku. Aktivnost koja slijedi slikovito opisuje koja su značenja i posljedice ovog ograničenja.

Aktivnost

1. Podijelite učenike u grupe.
2. Svaka grupa će trebato jedan radni list sa strane 72, i treba imati na raspolaganju jednu balansnu vagu sa težinama.
3. Neka učenici urade traženu aktivnost pa onda diskutujte rezultate.

Radni list za Aktivnost: Sortiranje Težina

Cilj: To find the best method of sorting a group of unknown weights into order.

Trebate imati: Pjesak ili vodu, 8 identičnih posuda, komplet and or water, 8 identical containers, a set of balance scales

Šta treba uraditi:

1. Ispunite svaku posudu sa različitom količinom pjeska ili vode. Čvrsto zatvorite svaku posudu.
2. Izpremještajte posude tako da više ne znate koji je dobar, pravilan redoslijed težina.
3. Nađite najlakšu težinu. Koji je najlakši način da se to uradi?

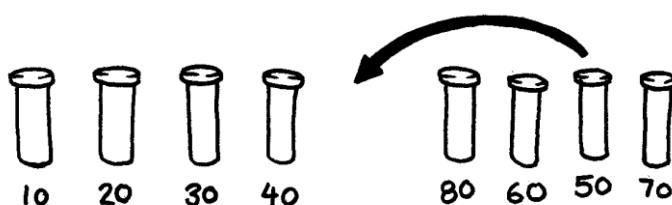
Primjedba: Imate parvo koristiti samo vagu da utvrdite koliko je težak svaka od posuda. Samo dvije težine mogu biti poređene u jednom datom trenutku.

4. Izaberite na slučajan način tri posude (težine) i sortirajte ih u redoslijedu od najlakše do najteže koristeći samo datu vagu. Na koji način ste to uradili? Koji je najmanji broj poređenja koja su vam potrebno za ovaj zadatak? Zašto?
5. Sada sortirajte sve objekte (posude) u redoslijed od najlakše do najteže.

Kada mislite da ste završili provjeriti vaš redoslijed upoređujući vaganjem svaka dva susjedna objekta.

Sortiranje selekcijom

Jedan metod koji računar može koristiti se naziva *sortiranje selekcijom* (eng. *selection sort*). Ovako sortiranje selekcijom radi. Prvo pronađite najlakšu težinu u cijelom skupu I stavite je sa strane. Potom, nađite najlakšu težinu među preostalim težinama i stavite je sa strane. Ponavljajte ovo sve dok sve težine ne budu uklonjene iz početnog skupa.



Prebrojte koliko poređenja ste napravili.

Eksta za Eksperte: Pokažite kako možete matematički izračunati koliko poređenja treba napraviti da bi sortirali 8 objekata. Koliko poređenja za sortiranje 9 objekata? A šta je sa 20 objekata?

Radni list za Aktivnost: Podijeli i Vladaj

Quicksort

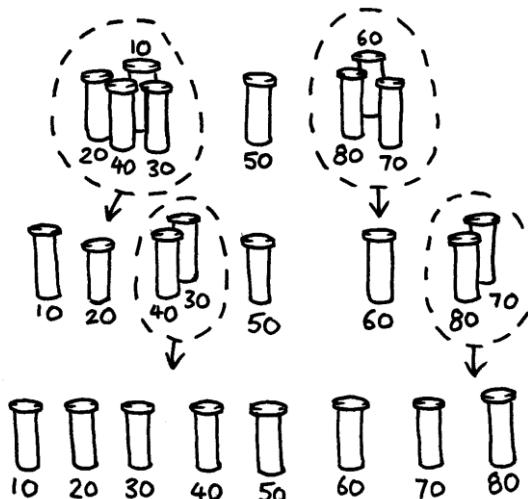
Quicksort (brzo sortiranje ali se obično ne prevodi) je mnogo brži metod od sortiranja selekcijom naročito ako treba sortirati velike liste objekata. U stvari, to je jedan od najboljih poznatih metoda za sortiranje. Ovo je način na koji quicksort radi.

Izaberite jedan od objekata na slučajan način i postavite ga na jednu stranu svoje vase.

Sada upoređujte svaki od preostalih objekata sa izabranim objektom. Oni koji se pokažu lakšim stavite na lijevu stranu, prvoizabrani objekat u sredinu pa nakon toga sa desne strane sve teže objekte. (U zavisnosti od sreće i slučajnosti moguće je da imate sa jedne strane mnogo više objekata nego na drugoj strani.)

Izaberite sada jednu od grupa (onu sa lijeve ili sa desne strane) i ponovite cijelu proceduru. Sada uraditi sve isto i za preostalu grupu. Ne zaboravite sačuvati onaj poznati prvoizabrani objekata stalno u sredini.

Sada ponavljajte istu procedure sa svim preostalim grupama i sve dok ne postoji grupa sa više od jednog elementa u njoj. Nakon što sve grupe budu podjeljene u grupe od jednog objekta cijela lista objekata će biti sortirana (tj. biće u dobrom redoslijedu).



Koliko je potrebno poređenja za cijeli ovaj proces?

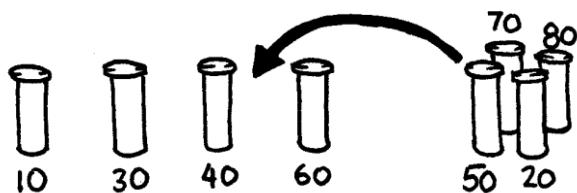
Trebali bi se uvjeriti da je quicksort efikasniji metod od metoda sortiranja selekcijom osim ako se ne dogodi da stalno izabirete najlakši ili najteži objekat za početak dijeljenja grupe. Ukoliko ste bili tako sretni da svaki put izabirete srednju težinu u datoj grupi onda ste morali uraditi samo 14 poređenja u usporedbi sa ukupno 28 poređenja potrebnih za sortiranje selekcijom. U svakom slučaju metod sortiranja quicksort neće nikada biti ništa sporiji (gori) od sortiranja selekcijom ali može biti puno bolji!

Ekstra za Eksperte: Ako quicksort na slučajan način uvijek izabire najlakši objekat koliko će onda ukupno paređenja napraviti?

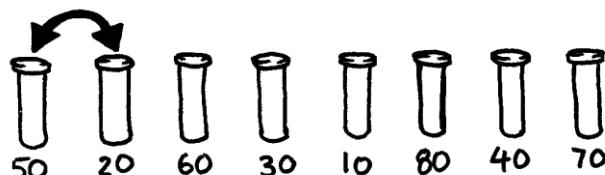
Varijacije i dodatne aktivnosti

Mnogo različitih metoda za sortiranje je otkriveno. Sada možete probati sortirati vaše težine koristeći neke od njih:

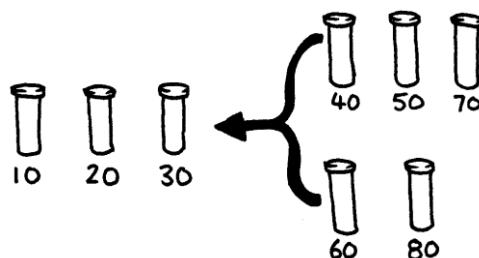
Sortiranje umetanjem (eng. *Insertion sort*) radi tako što uklanja redom po jedan objekat iz nesortirane grupe objekata i onda ga umeće na njegovu pravu poziciju u sortiranoj listi koja malo po malo raste (pogledati sliku ovdje dolje). Sa svakim novim umetanjem grupa nesortiranih objekata se smanjuje a sortirana lista raste i to sve dok cijela lista nije sortirana. Oni koji igraju karte (na primjer remi) često koriste ovaj metod za sortiranje karata koje imaju u ruci.



Bubble sort se sastoји od prolaska kroz listu iznova i iznova izmenjujući dva susjedna objekta koja su u pogrešnom redoslijedu. Lista je konačno sortirana kada se u jednom cijelom prolazu ne dogodi nijedna izmjena. Ovaj metod nije naročito efikasan ali neki ljudi ga puno lakše razumiju nego druge metode.



Sortiranje sastavljanjem (eng. *Mergesort*) je još jedan metod koji koristi ‘podijeli I vladaj’ (eng. ‘divide and conquer’) paradigm da bi sortirali listu objekata. Na početku listu podijelimo na slučajan način na dvije liste iste veličine (ili približno iste ukoliko je broj elemenata liste neparan broj). Svaka dobivena polovina je onda sortirana pa se onda te dvije liste sastave u jednu sortiranu listu. Sastavljanje dvije sortirane liste je lagano—dovoljno je izabrati manji od dva objekta koji se nalaze na čelu svake od dvije liste. Na slici ovdje dolje težine od 40 i 60 gramma su na početku dvije liste pa je sljedeći objekat koji ćemo dodati onaj od 40 gramma. Pitanje je sada kako sortirati dvije manje liste koje smo dobili dijeljenjem. Jednostavno—koristite sortiranje sastavljanjem! Na kraju, sve liste će biti usitnjene i podjeljene u jednoelementne liste tako da ne trebate brinuti kada da se zaustavite sa dijeljenjem.



Zašto je ovo sve važno?

Puno je lakše naći neku informaciju u sortiranoj listi. Telefonski imenik, rječnik stranih riječi, ili indeks na kraju knjige su uvijek dati u alfabetском redoslijedu, i naš život bi bio mnogo teži ako ne bi bilo tako. Ako je neka lista brojeva (na primjer lista cijena) sortirana onda se lagano prepoznaju ekstremne vrijednosti jer se one onda nalaze na početku i na kraju liste. Takođe je jednostavno i lagano pronaći one koji se ponavljaju jer se nalaze jedne pored drugih.

Računari troše puno svog vremena sortirajući stvari u neki redoslijed pa ona programeri moraju pronaći brze i efikasne načine kako da to i urade. Neki od sporijih metoda kao što su sortiranje umetanjem, sortiranje selekcijom ili bubble sortiranje mogu biti korisni u nekim specijalnim slučajevima ali se oni brzi metodi kao što su quicksort ili sortiranje sastavljanjem obično koriste u praksi kada treba sortirati velike liste – na primjer, za listu sa 100,000 elemenata, quicksort je obično oko 2,000 puta brži do sortiranja selekcijom a za listu sa 1,000,000 elemenata brži je oko 20,000 puta. Računari obično moraju raditi i procesirati sa milionima elemenata (nije malo website-ova sa milionima korisnika, Facebook ima skoro dvije milijarde korisnika a samo jedna slika može imati više od milion piksela); razlika između neka dva algoritma može lagano biti kao razlika kada je potrebna 1 sekunda da se obradi skup podataka i 5 sati da se obradi isti skup podataka na isti način. Ne samo da bi čekanje prouzročeno takvim pristupom bilo neprihvatljivo za korisnika nego bi računar koristio i 20,000 puta više energije (što ne samo da može potrošiti brže bateriju na vašem uređaju nego ima i veliki uticaj na naš okoliš) pa tako izbor odgovarajućeg algoritma ima zaista ozbiljne posljedice.

Quicksort koristi paradigmu, pristup poznat pod imenom Podijeli i Vladaj (eng. *Divide and Conquer*) Tokom izvršavanja quicksort-a vi neprestano dijelite listu na manje dijelove pa onda izvršavata isti quicksort na svakom novodobivenom dijelu. Lista se neprestano dijeli iznova sve dok ne postane dovoljno mala za vladanje. U slučaju quicksort-a liste se dijele sve dok ne sadrže samo po jedan elemenat. Tada je stvarno trivijalno sortirati takvu listu! Iako ovo sve može izgledati dosta zahtjevno (i možda komplikovano na početku) u praksi je ovaj metod značajno brži od ostalih metoda. Ovo je jedan primjer korištenja u praksi ideja *rekurzije* (eng. *recursion*) gdje jedan algoritam koristi (poziva) sam sebe da bi riješio problem – ponovo ovo može izgledati jako čudno ali budite uvjereni da to radi jako dobro i efikasno.

Rješenja i pomoć

1. Najbolji način da pronađemo najlakši objekt je da pregledamo svaki od njih i zapamtimo koji od njih je do sada bio najlakši. To znači, poredimo dva objekta i zapamtimo ona laci. Nakon toga poredimo taj objekat sa novim objektom i zapamtimo ponovo onaj laci koji će onda služiti za dalja poređenja. Ponavljamo postupak sve dok ne pregledamo sve date objekte.
2. Uporedimo težine na balansnoj vagi. Ovaj zadatak se lagano može uraditi sa samo tri poređenja a ponekada su dovoljna i samo dva poređenja—u slučaju da učenici razumiju da je operacija poređenja tranzitivna (to znači, ako je A laci od B i B laci od C onda I A mora biti laci od C).

Eksperти:

Ovdje je ukratko uputstvo kako sabrati sva poređenja potrebna za sortiranje selekcijom.

Da bi pronašli najmanji elemenat od dva data objekta potrebno vam je jedno poređenje, za tri objekta potrebna su dva poređenja, za četiri tri poređenja, i tako dalje. Da bi sortirali osam objekata koristeći sortiranje selekcijom potrebno je 7 poređenja da bi našli prvi elemenat, šest poređenja da bi našli naredni, pet za naredni i tako dalje. To nam daje sljedeću formulu:

$$7 + 6 + 5 + 4 + 3 + 2 + 1 = 28 \text{ poređenja.}$$

Za n objekata potrebno je $1 + 2 + 3 + 4 + \dots + n - 1$ poređenja za sortiranje.

Sabiranje ovih brojeva postaje lagano oko ih regrupišemo na dobar način.

Na primjer, da bi sabrali brojeve $1 + 2 + 3 + \dots + 20$, regrupišimo ih ovako

$$(1 + 20) + (2 + 19) + (3 + 18) + (4 + 17) + (5 + 16) +$$

$$(6 + 15) + (7 + 14) + (8 + 13) + (9 + 12) + (10 + 11)$$

$$= 21 \times 10$$

$$= 210$$

U opštem slučaju imamo da je suma $1 + 2 + 3 + 4 + \dots + n - 1 = n(n - 1)/2$.

Aktivnost 8

Brže od Sata—*Mreže Sortiranja*

Sažetak

Iako su računari poznati kao brzi (i sve brži i brži) ipak postoje stvarna ograničenja koliko brzo mogu riješavati problem. Jedan od načina da se stvari ubrzaju je da koristimo nekoliko računara za rješavanje različitih dijelova jednog problema. U ovoj aktivnosti ćemo koristiti mreže sortiranja koje mogu uraditi više poređenja u isto vrijeme.

Veze sa curriculum-om

- ✓ Matematika: Broj – Istraživanje broja: Veći od, manji od

Vještine

- ✓ Poređenje
- ✓ Slaganje stvari
- ✓ Razvoj algoritama
- ✓ Zajedničko rješavanje problema

Dobna/starosna grupa

- ✓ 7 godina i više

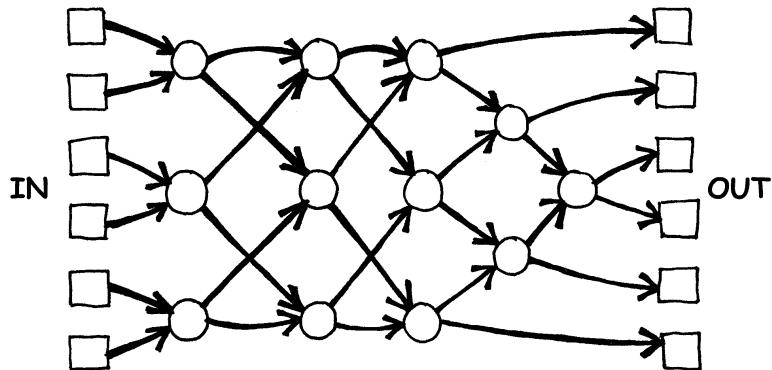
Materijal

Ovo je grupna aktivnost koja se izvodi vani.

- ✓ Kreda
- ✓ Dva kompleta od po šest karata.
Kopirati Uzorak za Kopiranje: Mreže sortiranja(strana 80) na kartu i izrezati je
- ✓ Štopericica za mjerjenje vremena

Mreže Sortiranja

Prije nego što aktivnost počne iskoristiti kredu kako bi označili jednu ovaku mrežu u dvorištu škole.

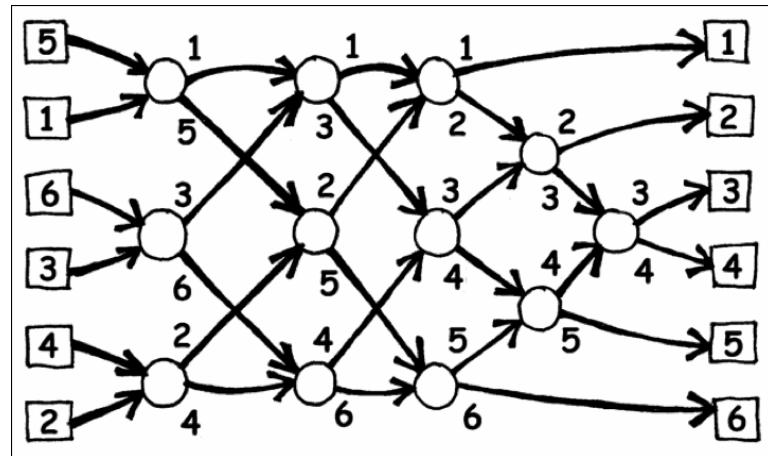


Uputstvo za Učenike

Ova aktivnost će vam pokazati kako računari sortiraju slučajne brojeve u jedan redoslijed koristeći nešto što nazivamo mreža sortiranja (eng. sorting network).

1. Organizujte se u grupe od po šest učenika. Samo jedna grupa će koristiti mrežu u određenom trenutku.
2. Svaki od članova tima uzima jednu numerisanu kartu.
3. Svaki član tima stoji u jednom od kvadrata na lijevoj (IN) strani mreže u dvorištu. Vaši brojevi treba da budu potpuno pomješani i u slučajnom rasporedu.
4. Šetate se duž označenih linija, i kada dostignete mjesto označeno krugom **morate sačekati da još jedna osoba stigne na isto mjesto**.
5. Kada I drugi član tima stigne na mjesto označeno krugom uporedite karte koje imate. Osoba sa manjom kartom odlazi putem koji void lijevo. Ukoliko imate veći broj onda vi odlazite putem koji vodi desno.
6. Da li ste u dobrom redoslijedu kada se nađete na drugom kraju mreže (OUT) u dvorištu?

Ukoliko neko od članova tima napravi grešku svi učenici moraju početi ispočetka. Provjerite još jednom da ste razumjeli operacije na svakom čvoru (krugu) mreže gdje manja vrijednost odlazi putem lijevo a veća odlazi putem desno. Na primjer:



Uzorak za fotokopiranje: Mreže sortiranja

1

2

3

4

5

6

156

221

289

314

422

499

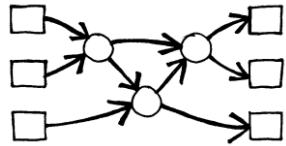
Varijacije

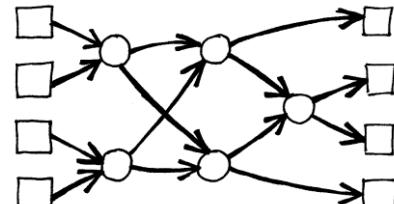
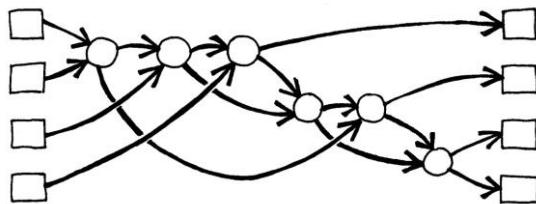
1. Onda kada učenici dobro upoznaju funkcionisanje aktivnosti iskoristite štopericu da bi izmjerili koliko vremena treba svakom timu da prođe kroz mrežu.
2. Koristite karte sa većim brojevima (na primjer trocifrene brojeve u uzorku za kopiranje).
3. Napravite karte koje će imati još i veće brojeve i za čije poređenje je potrebno neko vrijeme, ili koristite riječi koje se moraju porediti alfabetski.
4. Sve ovo može biti i jednostavna vježba za neke druge predmete: na primjer na muzičkom vaspitanju možete porediti note odštampane na kartama i redati ih od najnižeg tona prema višim ili od najkraćeg prema najdužim.

Dodatne aktivnosti

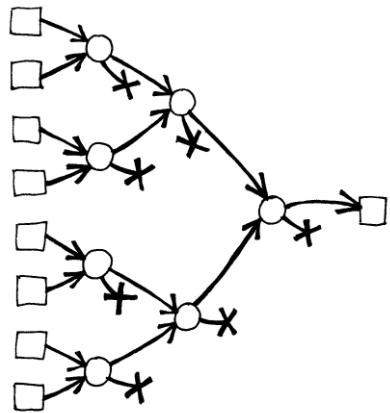
1. Šta bi se dogodilo ako bi manja karta išla desno umjesto u lijevo a veća lijevo umjesto u desno? (Brojevi bi bili sortirani u obrnutom redoslijedu.)

Da li je isti slučaj ako bi koristili mrežu unatrag? (Ne mora značiti da će sortiranje funkcionisati, i učenici bi sami trebali biti u stanju pronaći jedan primjer ulaza koji bi bio u pogrešnom redoslijedu na izlazu.)

2. Pokušajte napraviti manju ili veću mrežu. Na primjer ovdje je primjer mreže koja sortira samo tri broja. Učenici bi trebali biti u stanju konstruisati istu mrežu samostalno.
3. Ovdje dolje su predstavljene dvije različite mreže koje će pravilno sortirati ulaz sa četiri vrijednosti. Koja od njih je brža? (Druga je svakako brža. Dok prva mreža traži da se sva poređenja izvrše serijski (jedan za drugim) u drugoj mreži neka poređenja se mogu raditi i istovremeno. Prva mreža je jedan primjer serijske, sekvenčijalne obrade dok je druga mreža primjer gdje koristimo i mogućnosti paralelnog računanja kako bi ubrzali cijeli process.)



4. Pokušajte napraviti veće mreže sortiranja.
5. Mreže možemo koristiti kako bi pronašli najmanje ili najveće vrijednosti nekog ulaza. Na primjer, ovdje je jedna mreža sa osam ulaza a jedino najmanja vrijednost će se pojaviti na izlazu (sve ostale vrijednosti će ostati na slijepom putu (eng. dead end) u mreži).



6. Koji od procesa u svakodnevnom životu, koristeći paralelne aktivnosti, mogu biti ubrzani a koji ne mogu? Na primjer, kuhanje nekog jela ako bi koristili samo jednu posudu jer bi onda sva jela morali kuhati jedno za drugim. Koji od poslova se mogu završiti brže ako zaposlimo više ljudi? Koji poslovi se pak ne mogu ubrzati?

Zašto je ovo sve važno?

Kako računare koristimo sve više i više u svakodnevnom životu želimo da oni mogu obrađivati informacije što je brže moguće.

Jedan od načina da ubrzamo rad računara je da pišemo programme koje će koristiti ukupno manji broj kompjutacionih koraka (upravo kao što je to pokazano u Aktivnostima 6 i 7).

Drugi način da riješimo neki problem brže je da koristimo više računara koji bi radili na različitim dijelovima istog zadatka u isto vrijeme. Na primjer, u mreži sortiranja za šest brojeva iako imamo ukupno 12 poređenja da bi sortirali svih šest brojeva čak i do tri poređenja se rade u isto vrijeme. To znači da će potrebno vrijeme za izvršavanje svih poređenja biti jednakom kao da radimo samo 5 koraka poređenja serijski. Ova paralelna mreža sortira listu više od dva puta brže nego što bi to mogao uraditi neki kompjutacioni sistem koji može izvršavati samo jedno poređenje u jednom trenutku.

Nije moguće ubrzati izvršenje svakog zadatka koristeći paralelno računanje. Uzmimo za analogiju I primjer, osobu koja kopa kanal koji je dugačak deset metara. Ukoliko bi deset ljudi kopalo po jedan metar cijeli zadatak bi se uradio mnogo brže, i završio ranije. S druge strane, ista strategija se ne može koristiti prilikom kopanja rupe koja je duboka deset metara—drugi metar rupe nije dostupan sve dok nije iskopan u potpunosti prvi metar. Programeri i računarski naučnici stalno aktivno traže najbolje načina da podijele problem na manje dijelove kako bi ih onda skupina računara mogla rješavati paralelno.

Aktivnost 9

Blatnjavi Grad—Minimalno pokrivajuće stablo

Sažetak

Naše društvo je povezano velikim brojem različitih mreža: telefonske mreže, mreže za snabdjevanje, računarskim mrežama, kao i razgranatim putnim mrežama. Za svaku pojedinu mrežu obično postoji neki izbor o tome gdje izgraditi put, postaviti kablove ili kuda uspostaviti radio vezu. Potrebno nam je dakle da nađemo, na efikasan način, kako najbolje povezati objekte u jednu mrežu.

Veze sa curriculum-om

- ✓ Matematika: Geometrija – Istraživanje objekata i prostora: Pronalaženje najkraćih puteva na mapi

Dobna/starosna grupa

- ✓ 9 godina i više

Vještine

- ✓ Rješavanje problema
- ✓

Materijal

Svaki učenik će trebati sljedeće:

- ✓ Radni List za Aktivnost: Problem blatnjavog grada (strana 86)
- ✓ Brojači ili kvadratići od kartona (otprilike 40 po jednom učeniku)

Blatnjavi Grad

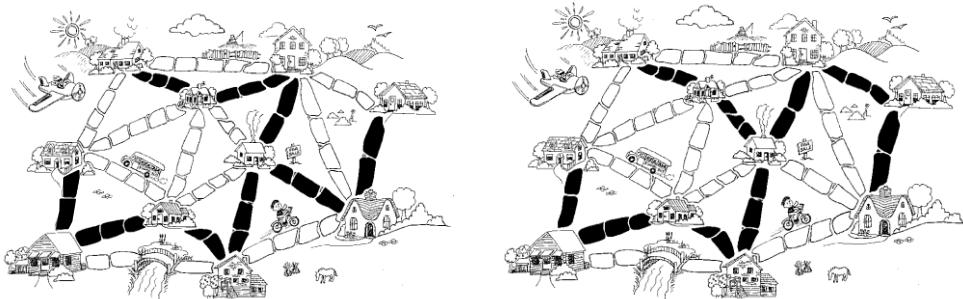
Uvod

Ova aktivnost će vam pokazati na koji način se koriste računari da bi pronašli najbolja rješenja za neke problem iz stvarnog života (eng. teal-life problems) kao što je na primjer problem kako povezati energetskim kablovima sve kuće u naselju. Pozovite učenike da koriste radni list sa strane 86 gdje je detaljno objašnjen problem ‘Blatnjavi Grad’.

Diskusija koja slijedi

Podijelite sa svima rješenja koja su pojedini učenici našli. Koju su strategiju koristili za nalaženje rješenja?

Jedna dobra strategija za nalaženje najboljeg rješenja je da počnete sa praznom mapom I onda postepeno dodavati brojače sve dok sve kuće ne budu povezane tako što ćete dodavati pojedine puteve u njihovom rastućem redoslijedu po dužini ali pazеći pri tome da ne povežete kuće koje su već povezane. Moguće je pronaći različita rješenja ako promijeniti redoslijed po kojem se dodaju putevi iste dužine. Dva moguće rješenja su prikazana ovdje dolje.



Jedna druga strategija je da počnete sa svim putevima koji su popločani pa da onda uklanjate jedan po jedan put koji nije potreban. Ova strategija će ipak zahtjevati malo više napora.

Gdje možete pronaći primjere mreža u stvarnom životu?

Programeri nazivaju modele reprezentacije ovih mreža “grafovima”. Stvarne mreže mogu biti predstavljene grafom kako bi mogli riješiti probleme kao što su dizajniranje najbolje mreže puteva između nekoliko susjednih gradova, ili određivanje skupa avionskih linija u jednoj zemlji ili na jednom kontinentu.

Ima jako puno i drugih algoritama koje možemo koristiti I primjeniti na grafovima kao što su, na primjer, nalaženje najkraće udaljenosti između dvije date tačke, ili pronalaženje najkraćeg puta koji posjećuje sve date tačke.

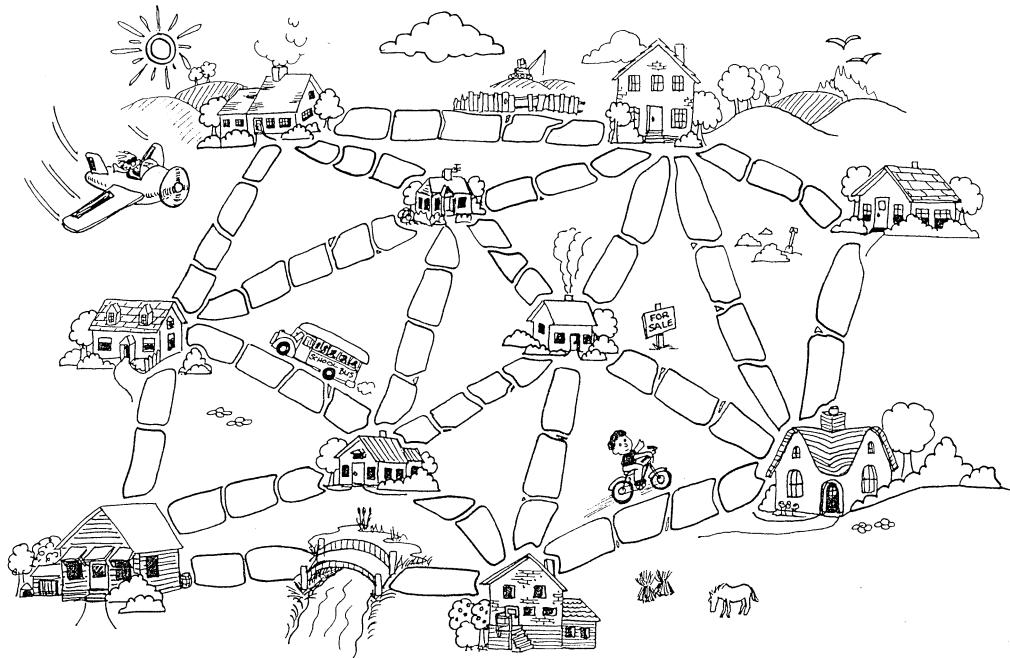
Radni list za Aktivnost: Problem Blatnjavog Grada

Jednom nekad davno bio je jedan grad koji nije imao puteva. Kretanje u samom gradu je bilo naročito otežano nakon velikih kišnih oluja jer je zemlja u gradu postajala jako blatnjava—automobili bi se zaglavljivali u blatu a ljudi bi prljali svoje čizme. Gradonačelnik grada je odlučio da neke ulice ipak moraju biti popločane ali nije želio da potroši više novca nego što je to zaista neophodno jer grad je u isto vrijeme imao projekat pravljenja bazena za plivanje. Gradonačelnik je zbog toga detaljno opisao dva uslova za popločavanje ulica:

1. Mora biti popločano dovoljno ulica tako da bude moguće svima da idu od svojih kuća do bilo koje druge kuće u gradu koristeći samo popločane ulice, i
2. Popločavanje mora biti što je moguće jeftinije.

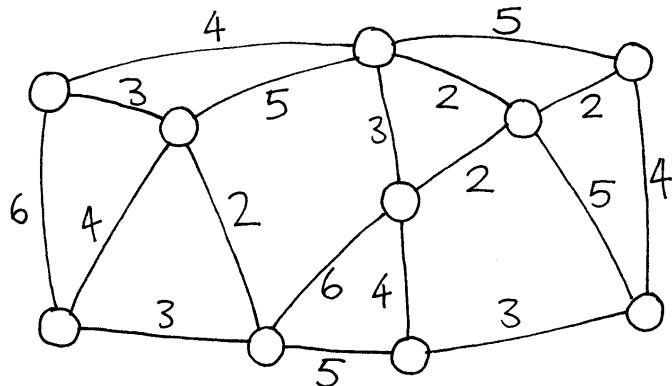
Ovdje dolje je data mapa samog grada. Broj kamenova za popločavanje između svake kuće određuje i cijenu popločavanja tog puta. Nađite najbolju mrežu koja povezuje sve kuće ali koristi što je moguće manje kamenja za popločavanje.

Koju ste strategiju za popločavanje koristili da bi riješili problem?



Varijacije i dodaci

Evo još jedan način da predstavimo grad i puteve između kuća:



Kuće su predstavljene kružićima, blatnjavi putevi su predstavljeni linijama a dužine svog od puteva su date brojem koji se nalazi uz liniju.

Programeri i matematičari često koriste ovaku vrstu dijagrama kako bi predstavili probleme na kojima rade. Oni taj dijagram nazivaju graf. Ova terminologija može biti zbumnjujuća jer se “graf” ponekad koristi i u statistici i označava tabelu sa nekim numeričkim podacima, kao što je na primjer stupčasti graf (ili grafikon). Ipak grafovi koje koriste programeri nisu nikako direktno povezani sa ovim tipom grafova. Takođe, dužine puteva na crtežu ne moraju biti proporcionalne dužini puteva u stvarnosti.

Napraviti nekoliko svojih problema Blatnjavog Grada i pozovite svoje prijatelje da ih riješe.

Možete li naći pravilo koje govori o tome koliko puteve ili veza je potrebno da bi imali jedno optimalno rješenje. Da li je taj broj zavisan od broja kuća u gradu?

Zašto je ovo sve važno?

Pretpostavimo da planirate na koji način isporučiti električnu energiju, gas ili vodu stanovnicima u jednom naselju. Mreža kablova ili cijevi treba da poveže sve kuće sa kompanijom snabdjevачem. Svaka kuća mora biti povezana na mrežu na neki način ali putevi kojima tačno dostavljamo potrebna dobra do kuća nisu zaista toliko važni, sve dok takvi putevi stvarno postoje.

Zadatak planiranje mreže sa minimalnom ukupnom dužinom se naziva problem *minimalnog pokrivačeg stabla* (eng. *minimal spanning tree* problem).

Minimalno pokrivaće stablo nije samo važno za gasnu ili električnu mrežu; rješenje za ovaj problem nam može pomoći prilikom planiranja računarskih mreža, telefonskih mreža, naftnih cjevovoda kao i mreža vazdušnog saobraćaja. Ipak, sa druge strane, ukoliko odlučujete kojim putem je najbolje putovati morate uzeti u obzir koliko je zgodno i komforno to putovanje za samog putnika kao i koja je ukupna cijena putovanja. Niko ne želi provesti duge sate u nekom avionu putujući velikim zaobilaznim putem da bi došao na svoju krajnju destinaciju samo zato što je to jeftinije. Algoritam koji rješava problem Blatnjavog Grada vjerovatno nije najbolji izbor za ovakve mreže i problema na njima jer ovaj algoritam samo minimizira ukupnu dužinu puteve ili ukupnu dužinu letova. .

Minimalno pokrivaće stablo je takođe korisno kao jedan od koraka prilikom rješavanja drugih problema na grafovima, kao što je, na primjer, problem "problem putujućeg trgovca" (eng. "travelling salesperson problem") u kojem želimo naći najkraći put koji posjećuje svaku tačku mreže jednom i samo jednom.

Postoje nekoliko efikasnih, brzih algoritama (metoda) za rješavanje problema minimalnog pokrivačeg stabla. Jednostavan metod koji uvijek daje optimalno rješenje je da započnemo bez ikakvih veza (puteva) i da onda postepeno dodajemo, jednu po jednu, vezu u njihovom rastućem redoslijedu po dužini i to samo onda kada povezuju dijelove mreže koji do tada nisu bili povezani. Ovaj algoritam se naziva Kruskal-ov algoritam prema istraživaču J.B. Kruskal, koji je objavio algoritam 1956 godine.

Za mnoge problema na grafovima, između ostalih i za problem putujućeg trgovca, programeri i računarski naučnici još uvijek traže dovoljno brze i efikasne načine za pronađenje najboljeg rješenja.

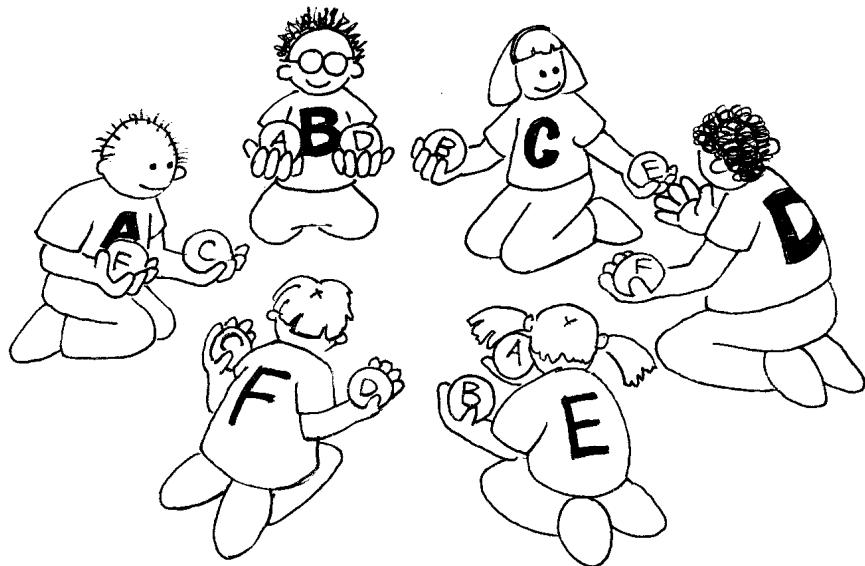
Rješenja i pomoć

Varijacije i dodaci (page 86)

Koliko mnogo puteva ili veza je potrebno da bi povezali grad sa n kuća? Pokaže se da u jednom optimalnom rješenju uvijek ima tačno $n-1$ veza, jer je taj broj veza uvijek dovoljan da se poveže svih n kuća, i da bi dodavanje još jedne veze kreiralo nepotrebnu alternativnu vezu između nekih kuća.

Aktivnost 10

Igra Narandži—Rutiranje i Potpuni zastoj na Mrežama



Sažetak

U situaciji kada imate mnogo osoba koje koriste isti resurs (kao što je slučaj sa automobilima koji koriste iste puteve, ili porukama koje putuju Internetom) postoji I mogućnost “potpunog zastoja” (eng. “deadlock”). Neki način saradnje, zajedničkog djelovanja je neophodan da bi izbjegli takve situacije.

Veze sa curriculum-om

- ✓ Matematika: Razvoj logike i razmišljanja

Vještine

- ✓ Zajedničko rješavanja problema
- ✓ Logičko razmišljanje

Dobna/starosna grupa

- ✓ 9 godina i više

Materijal

Svaki učenik će trebati sljedeće:

- ✓ Dvije narandže ili dvije teniske loptice označene istim slovom, ili neke dvije voćke (neka umjetna voćka je možda najbolji izbor)
- ✓ Naljepnica koja pokazuje odgovarajuće slovo, ili šešir u boji, bedž ili majica koja se slaže sa izabranim voćem (iste boje i slično)

Igra Narandži

Uvod

Ova igra podstiče i cilj joj je zajedničko rješavanje problema. Krajnji cilj je da svaka osoba završi tako što će u rukama imati narandže sa svojim slovom.

1. Grupe od 5 ili više učenika sjede u krugu.
2. Učenici su označeni različitim slovima alfabeta (koristeći na primjer naljepnice), ili je svakom od njih dodjeljena jedna boja (na primjer šešir određene boje, ili boja njihove odjeće i slično). Ukoliko za označavanje koristimo alfabet onda nam trebaju i dvije narandže sa istim slovom na svakoj od njih, osim za jednog učenika koji koji ima samo jednu odgovarajuću narandžu (sa odgovarajućim slovom) kako bi bili sigurni da uvijek postoji jedna prazna ruka. Ukoliko odlučimo da koristimo drugo voće, onda je potrebno imati po dva komada voća za svako dijete, na primjer učenik sa žutim šeširom će imati dvije banana dok učenik sa zelenim šeširom može imati dvije zelene jabuke, uvijek sa izuzetkom jednog učenika kojem odgovara samo jedan komad voća.
3. Podijelite sve narandže (ili svo voće) na slučajan način učenicima u krugu. Sada svaki učenik ima dva komada osim jednog koji ima samo jedan komad u ruci. (Nijedan učenik ne bi trebao imati sebi odgovarajuću narandžu ili voćku)
4. Učenici sada daju narandže/voće u krug svom susjedu sve dok svaki učenik ne dobije jednu narandžu sa svojim slovom (ili voće odgovarajuće boje). Obavezno je pratiti sljedeća dva pravila:
 - Samo jedan komad voća se može držati u jednoj ruci.
 - Komad voća se može dati samo u praznu ruku svog susjeda u krugu. (Učenik može dati susjedu bilo koju od dvije narandže koje trenutno ima.)

Učenici će brzo razumjeti da ako oni sami postanu “pohlepni” (eng. “greedy”) (i zadržavaju svoje voće čim dođu do njega) onda cijela grupa može imati problem da dođe do traženog rezultata. Potrebno je možda naglasiti činjenicu da pojedinci ne “pobjeđuju” u igri već da je igra završena onda kada svako ima odgovarajuće voće u svojoj ruci.

Diskusija nakon Aktivnosti

Koju strategiju su učenici koristili da bi riješili problem?

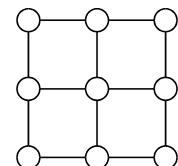
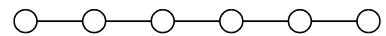
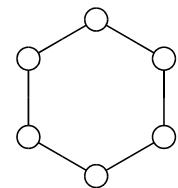
Gdje u stvarnom životu možete doživjeti ili vidjeti potpuni zastoj? (Neki od primjera mogu biti gužve u saobraćaju, ili pokušaj da mnogo ljudi odjednom prođe kroz ista vrata.)

Dodatne Aktivnosti

Pokušajte uraditi ovu aktivnost sa većim i manjim krugovima.

- Tražite od učenika da utvrde nova pravila za igru.

- Uradite sada cijelu aktivnost bez ikakvog razgovora i dogovaranja, u tišini.
- Pokušajte imate različite konfiguracije učenika kao što je sjedenje na jednoj liniji, ili tako da učenik može imate više od jednog susjeda. Neki od prijedloga su ovdje pokazani.



Zašto je ovo sve važno?

Rutiranje i potpuni zastoj su neki od problema u mnogim mrežama kao što su putne mreže, telefonski sistemi ili računarske mreže. Inžinjeri provode značajan dio svog vremena pokušavajući pronaći način da riješe ovakve probleme—i kako na kraju osmisli i planirati mreže tako da su ovi problemi lakši za rješavanje.

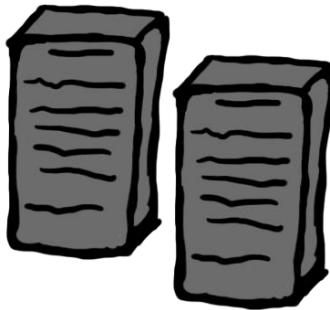
Rutiranje, gužve i potpuni zastoj mogu često biti frustrirajući problemi u različitim mrežama. Razmislite samo o automobilskom saobraćaju kada se nađete u velikoj gužvi. Dogodilo se već nekoliko puta da je saobraćaj na ulicama New York City-ja postao tako zakrčen da je došlo do potpunog zastoja: niko nije mogao pomjeriti svoje auto ni za malo! Ponekad kada računari “ispadnu” (eng. “down”) iz poslovnih sistema (kao što su banke) problem je u stvari uzrokovan potpunim zastojem u komunikacijama. Osmisljavanje i planiranje mreža tako da je rutiranje lagano i efikasno i da su gužve minimalne je jedan težak problem koji mora rješavati veliki broj inžinjera različitih struka.

Ponekad više od jedne osobe želi da ima isti podatak u isto vrijeme. Ukoliko se jedan podatak (kao što je bankovni račun jednog klijenta) ažurira onda je važno da je “zaključan” (eng. “lock”) tokom cijelog ažuriranja. Ukoliko nije zaključan neko drugi može uraditi svoje ažuriranje u isto vrijeme pa će na kraju stanje na bankovnom računu biti pogrešno. S druge strane ako se ovo zaključavanje miješa sa zaključavanjem nekog drugog podatka može opet doći do potpunog zastoja u operacijama.

Jedan od najinteresantnijih pravaca razvoja računarskih sistema je mogućnost paralelnog računanja gdje stotine ili hiljade (ili čak milioni) PC-procesora zajednički (u jednoj mreži) rade kao jedan veliki računar. Mnogo problema sličnih problemu Igre Narandži se odigravaju i rješavaju neprestano (ali mnogo, mnogo brže!) što na kraju omogućava takvim paralelnim računarima da zaista i rade.

Aktivnost 11

Kamene Pločice—Protokoli Mrežne Komunikacije



Sažetak

Računari razgovaraju između sebe preko Interneta šaljući poruke. S druge strane, Internet nije uvijek pouzdan i ponekada se poruke jednostavno izgube. Postoje zbog toga dijelovi informacije koje dodajemo našim porukama kako bi bili sigurni da su poslani. Ovi dijelovi informacije čine dio jednog protokola.

Veze sa curriculum-om

- ✓ Matematika: Razvoj logike i razmišljanja
- ✓ Maternji jezik: Komunikacija, uzajamno slušanje

Vještine

- ✓ Zajedničko rješavanja problema
- ✓ Logičko razmišljanje

Dobna/starosna grupa

- ✓ 9 godina i više

Materijal

Svaki učenik će trebati sljedeće:

- ✓ Mnogo praznih "Pločica"

Svaki kurir će trebati

- ✓ Jedan komplet karata za akciju na porukama

- ✓ Nastavnik će trebati:

- ✓ Štopericu

Kamene Pločice

Uvod

U ovoj aktivnosti učenici će razmatrati kako se uspješno realizuju različite metode komunikacije. Posmatrajući i primjenjujući zadana pravila i procedure učenici se uvode u pojam komunikacijskih protokola. Radeći kroz scenarije igra-ulogu učenici testiraju kako njima zadan protokol funkcioniše u jednom nepouzdanom okruženju, sličnom onom koje nalazimo u paketnom prospajanju na samom Internetu, ili preciznije rečeno na TCP/ IP.

Priprema (30 minuta)

1. Prije svega pripremite sve karte. Trebate odštampati Karte Akcije (koje su ovdje dolje) I trebate ih isjeći. Ove karte su osnova cijele igre.
2. Nakon toga odlučite o nekoliko poruka koje će učenici slati tokom igre. Ovdje je važno da te poruke *nisu smislene* govorne rečenice ili bilo šta čiju strukturu ili smisao sami učenici mogu prepoznati. Nešto kao "1LHC255HD(RLLS" bi bila jedna pogodna poruka za igru ili možda neki telefonski broj.
3. Odštampajte sada kopije "pločica". Na svakoj pločici ima mjesta za šest slova ili cifara tako da nije moguće staviti cijelu poruku na samo jednu pločicu. Potrebno vam je otprilike 30 pločica po jednom učeniku u zavisnosti od toga koliko dugo želite da igrate igru.

Primjedba: Postoje tri različita tipa Karata Akcije; zastoj, nije isporučeno, isporučeno. Određivanje odnosa među tipovima Karata Akcije će predstavljati kvalitet vašeg kurira. Više karata "isporučeno" znači da je kurir pouzdaniji. Više karata "zastoj" i "nije isporučeno" označava manje pouzdanu mrežu. Ove karte imaju svoje analogne podatke u jednoj računarskoj mreži/komunikacijskom kanalu.

Igranje igre

1. Podijelite razred u parove. Od krucijalne važnosti je da parovi sjede daleko jedni od drugih tako da se mogu vidjeti ili na drugi način komunicirati između sebe. Dvije učionice bi bile idealne ali sjedenje na različitim krajevima jedne učionice će biti dovoljno dobro.
2. Dajte jednoj osobi svakog para poruku koju treba prenijeti svom partneru u paru.
3. Pomiješajte Karte Akcije i onda izaberite jednog kurira. Možete vi sami biti kurir ili ako imate neparan broj učenika onda preostali učenik može biti kurir. Moguće je takođe da vam treba više od jednog kurira ukoliko imate veće razrede.
4. Sada učenik treba da zapise podatke na svojoj pločici i da je preda kuriru. Na pločici mora biti najmanje ime druge osobe iz para.
5. Kurir sada uzme prvu Kartu Akcije i okreće je, pročita je i onda odlučuje šta treba uraditi sa pločicom.
6. Ponoviti korake 4 i 5 sa svakom pločicom.

Nakon 5 ili malo više minuta potpunog haosa i frustracija učenici su već sami razumjeli da samo ime nije dovoljno dobro za neki protokol. Zaustavite sve aktivnosti i diskutujte to sa učenicima ... koji je osnovni problem koje imaju? Da li je to redoslijed? Možda bi bilo najbolje ako bi koristili jedan od ovih šest mesta na pločici za označavanje rednog broja pločice? To onda znači da ima manje prostora za prave podatke – šta to sada znači za broj pločica koje moramo koristiti da bi prenijeli jednu cijelu poruku?

Nakon još nekog vremena učenici će primjetiti koji su ostali problemi koji se dešavaju i svi trebate zajedno diskutovati te problem. Mogući problem mogu biti da jedna pločica fali tako da

ne znamo da li je pločica isporučena i da li trebamo poslati istu još jednom. Jedno od rješenja koje možete predložiti je da se šalje nazad potvrda prijema i da se onda čekaju ovakve potvrde prije nego bi se ponovo poslala ista pločica– to onda znači da i učenik koji prima poruke treba imati prazne pločice da bi slao svoje poruke, a sam par se treba dogovoriti šta ovih 6 slova u odgovoru znače prije nego se igra počne igrati ponovo.

Potrebna su najmanje dva učenika za ovu igru ali mi preporučujemo da ih imate što je više moguće. Ukoliko imate veliki razred razmislite o tome da imate više kurira. Još jednom, sve ovo treba diskutovati u razredu. Šta se događa ako imate više kurira? Šta se događa ako imate samo jednog?

Isporuči ovu pločicu sada	Isporuči ovu pločicu nakon sljedeće pločice
Isporuči ovu pločicu sada	Isporuči ovu pločicu nakon sljedeće pločice
Isporuči ovu pločicu sada	Isporuči ovu pločicu nakon sljedeće pločice
Isporuči ovu pločicu sada	Ne isporučuj ovu pločicu
Isporuči ovu pločicu sada	Ne isporučuj ovu pločicu

<p>Za:</p> <table border="1"><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table> <p>Od:</p>							<p>Za:</p> <table border="1"><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table> <p>Od:</p>						
<p>Za:</p> <table border="1"><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table> <p>Od:</p>							<p>Za:</p> <table border="1"><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table> <p>Od:</p>						
<p>Za:</p> <table border="1"><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table> <p>Od:</p>							<p>Za:</p> <table border="1"><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table> <p>Od:</p>						
<p>Za:</p> <table border="1"><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table> <p>Od:</p>							<p>Za:</p> <table border="1"><tr><td></td><td></td></tr><tr><td></td><td></td></tr><tr><td></td><td></td></tr></table> <p>Od:</p>						

Kamene Pločice

U jednom drevnom gradu bilo je nekoliko jako važnih Ministara. Ovi Ministri su odlučivali kako će funkcionisati cijeli grad i donosili su važne odluke. Svaki od njih je živio u svojoj zasebnoj vili koje su bile razasute po cijelom gradu.

Ministri su često željeli komunicirati između sebe, trebali su slati i primate poruke poslane sa jednog na drugi kraj grada. Ministri su bili poznati i po brojevima vila gdje žive i imali su mogućnost korištenja grupe kurira čiji jedini posao je bio prenošenje poruka.

Jedini način za slanje poruka je bilo da se one napiše na velikim četvrtastim kamenim pločicama koje bi onda kuriri nosili na njihovo odredište. Kamene pločice su bili tačno određene veličine I mogle su sadržavati samo 6 dijelova jedne informacije na sebi. Jedan dio informacije je bilo jedno slovo ili jedna cifra. Jedna poruka je obično bila razbijena i napisana na nekoliko kamenih pločica a kako su pločice bile prilično teške bilo je moguće nositi samo jednu odjednom.

Kuriri nisu bili potpuno pouzdani i nije im se moglo vjerovati da će isporučiti svaku poruku jer su bili i zaboravni a ponekad i lijeni. Često su se zaustavljeni i pravili velike pauze tokom svog radnog vremena a nekada su pokušavali pobjeći iz grada i od svog teškog posla.

Ministri su željeli naći koji je dobar i pouzdan način komunikacije, željeli su razviti jedan kompletan sistem pravila koji bi onda doslovno primjenjivali u svojoj komunikaciji. Prateći takva pravila željeli su biti sigurni da mogu reći da li je ili ne njihova poruka isporučena i da li je njen sadržaj tačan. Ministri su već odlučili da se adresa primaoca (broj kuće) mora nalaziti na pločici.

Vaš zadatak je da u svojoj grupi razvijete komplet pravila koji bi omogućio Ministrima da komuniciraju između sebe....

Zašto je ovo sve važno?

Na Internetu su podaci koji se prenose razbijeni u pakete za prenos. S druge strane, kanali kojima putuju ovi paketi nisu uvijek pouzdani. Pojedini paketi su ponekad oštećeni, izgubljeni ili su izgubili svoj dobar redoslijed.

U Kamenim Pločicama, pločice predstavljaju pakete a njihov sadržaj su podaci. Pakteti sadrže u isto vrijeme i podatke i informacije u zaglavljiju (eng. header). Veličina zaglavljaja utječe na to koliko stvarnih podataka može biti preneseno – pa je potrebno naći dobar odnos između ovih veličina jer su paketi konačne, fiksne veličine.

Učenici će zaključiti da je potrebno zamjeniti neke dijelove pravih podataka i ustupiti mjesto informacijama kao što su redni broj paketa i ukupan broj paketa, i da li ili ne je to paket o potvrdi prijema (eng. an acknowledgement packet). Iz razloga što ove informacije zauzimaju prostor pravim porukama ukupan broj paketa potreban za slanje jedne poruke će se povećati.

Internet protokoli kao što su TCP i UDP uspješno balansiraju ove faktore kako bi omogućili pouzdan i efikasan prenos podataka.

Ova aktivnost je adaptirana na osnovu jedne druge aktivnosti dostupne kroz program “Computing Science Inside” (csi.dcs.gla.ac.uk).

DIO III

Saopštiti Računarima Šta Raditi— Predstavljanje Procedura

Saopštiti Računarima Šta Raditi

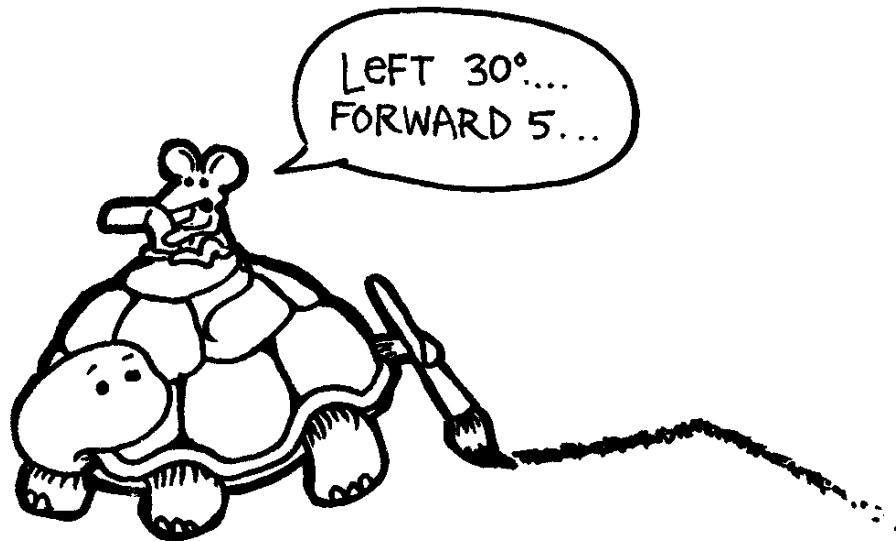
Računari prate i izvršavaju naredbe—na milione naredbi u samo jednoj sekundi. Da bi saopštili jednom računaru što uraditi sve što treba je dati mu prave naredbe. S druge strane, to nije uvijek tako jednostavno kako izgleda!

Kada mi sami dobivamo naredbe koje treba izvršiti onda koristimo “zdravi razum” njihovo značenje. Ako neko kaže “prođi kroz ta vrata,” onda on ne misli da treba stvarno probity vrata—misli da treba proći kroz vratnicu i ako je potrebno prvo otvoriti vrata! Računari su drugačiji. Zaista, kada su spojeni na jedan pokretni robot onda treba pažljivo razmisliti o svim sigurnosnim aspektima upravljanja kako ne bi došlo do uništavanja dobara ili izazivanja nesreća ako bi se naredbe izvršavala na “bukvalan” način—kaon a primjer probati proći kroz vrata. Treba malo vremena da se naviknete raditi sa nečim što slijepo, bez razmišljanja, i tačno izvršava svaku naredbu.

Dvije aktivnosti predložene u ovom dijelu će nam dati ideju na šta liči i kako je, koristeći unaprijed dat skup naredbi, komunicirati sa mašinom koja bespogovorno i slijepo sluša sve što joj se kaže.

Prva aktivnost će nas naučiti o “mašini” koju računari koriste da bi prepoznali riječi, brojeve ili stringove simbola sa kojima onda računar može dalje raditi. Ove “mašine” se nazivaju konačni automati.

Druga aktivnost uvodi nas i predstavlja nam način na koji je moguće komunicirati sa računarima. Jedan dobar programmer mora naučiti kako saopštiti računaru što treba uraditi i to koristeći unaprijed dat skup naredbi koje će se poslije slijepo izvršavati.



Aktivnost 12

Lov na blago—*Konačni Automati*

Sažetak

Računarski programi često moraju obraditi (procesirati) niz simbola kao što su slova ili riječi u jednom dokumentu, ili poneka trebaju analizirati tekst nekog drugog računarskog programa. Računarski stručnjaci često koristi konačne automate da bi to uradili. Jedan konačni automat (eng. finite-state automaton) (skraćeno FSA) prati dati skup naredbi da vidi da li računar može prepoznati riječ ili niz slova (string). Mi ćemo ovdje raditi sa nečim što je potpuno jednako jednom Konačnom Automatu—mapi blaga!

Veze sa Curriculum-om

- ✓ Matematika: Razvoj logike i logičkog razmišljanja – koristiti riječi i simbole kako bi se opisali i produžili i kompletirali dati uzorci
- ✓ Socijalne nauke
- ✓ Maternji jezik

Vještine

- ✓ Jednostavno čitanje mapa
- ✓ Prepoznavanje uzoraka
- ✓ Logika
- ✓ Praćenje naredbi

Dobna/starosna grupa

- ✓ 9 godina i više

Materijali

Nastavniku će trebati:

- ✓ Jedan komplet otočkih karata (naredbe moraju biti sakrivene i nepoznate od onih učenika koji će pokušati nacrtati mapu!)
Kopirajte Uzorak za fotokopiranje: Otočke karte (od stranice 110) i izrežite ih.
Presavijte ih duž isprekidanih linija i zalijepite, tako da prednja strana karte ima ime otoka a stražnja strana sadrži naredbe.

Svaki učenik će trebati:

- ✓ Radni list za aktivnost: Pronađite svoj put do bogatstava otočke riznice (strana 109)
- ✓ Olovka

Postoje je i dodatne aktivnosti za koje će svaki učenik trebati:

- ✓ Radni list za aktivnost: Blago otoka (strana 115)
- ✓ Radni list za aktivnost: Misteriozna igra novčića (strana 116)

Otok sa blagom

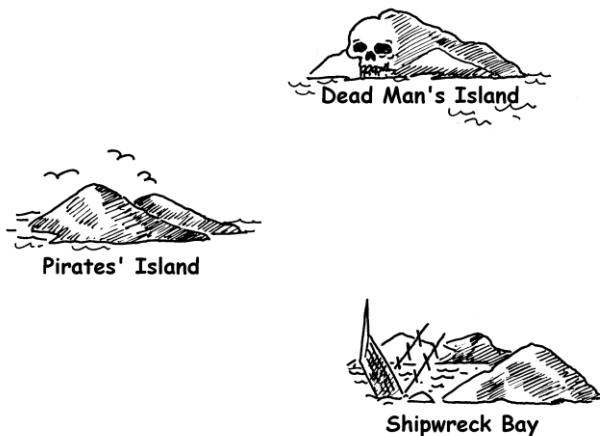
Uvod

Vaš zadatak je naći otok sa blagom. Prijateljski gusarski brodovi plove duž zadanih ruta među otocima u ovom dijelu svijeta i nude besplatan prevoz svim zainteresovanim putnicima. Svaki otok ima dva broda koja isplovjavaju sa njega, A i B, I možete izabrati bilo koji od njih za nastavak svog putovanja. Vi trebate naći najbolji put do otoka sa blagom. Na svakom otoku na koji stignete možete pitati ili za brod A ili za brod B (ali ne za oba). Osoba koju vidite na otoku će vam reći gdje će vas vaš sljedeći brod odvesti ali sami pirati nemaju mapu otoka kod sebe. Koristite svoju sopstvenu mapu kako bi bilježili gdje idete kao i na kojim brodovima ste putovali.

Demonstracija

(Primjedba: Ovo je različita mapa od one koja se koristi u samoj aktivnosti.)

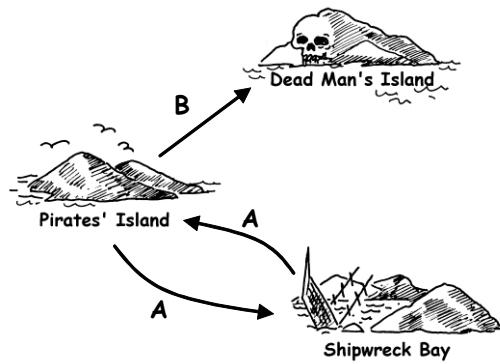
Na table nacrtajte dijagram sa tri otoka kao što je prikazano ovdje dolje:



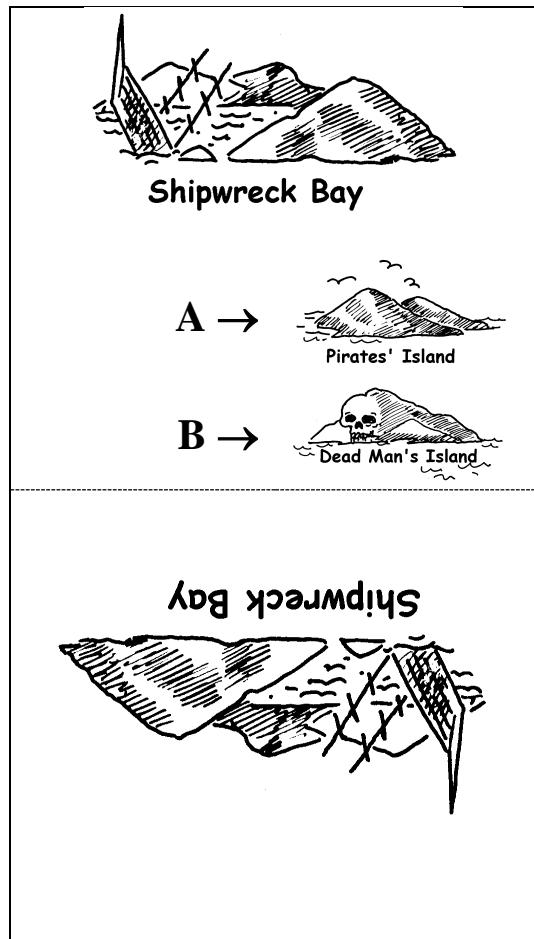
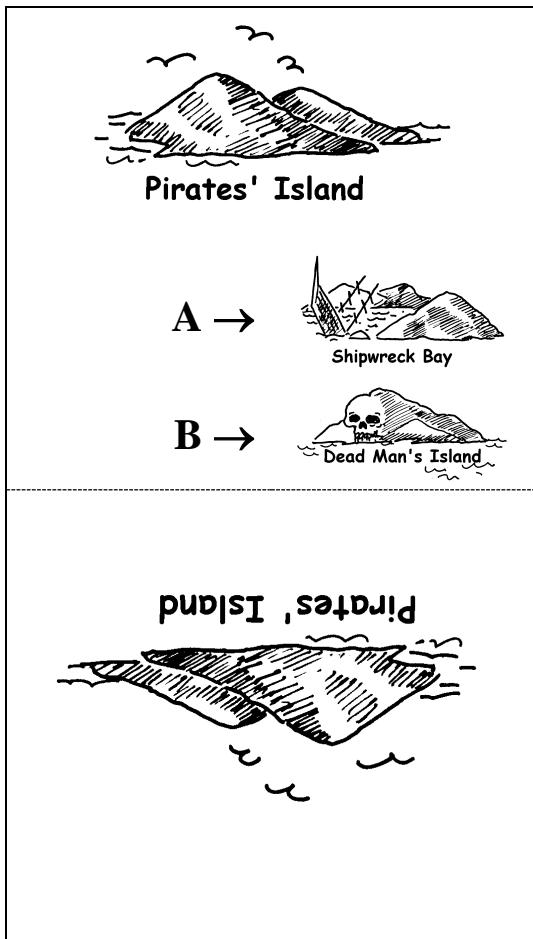
Kopirajte tri karte sa sljedeće dvije stranice i izaberite učenike koji će držati po jednu kartu. Ne zaboravite da su putevi na ovim kartama različiti od onih koje imamo u glavnoj aktivnosti.

Započinjući na Otku gusara (Pirates' island) zatražite brod A. Učenik će vas uputiti prema Zaljevu brodoloma (Shipwreck Bay). Označite ovaj dio puta na mapi. U Zaljevu brodoloma ponovo zatražite brod A. Sada ćete biti vraćeni nazad na Otok gusara (Pirates' island). Označite to na mapi. Sada zatražite brod B. Označite to na mapi. Ovaj put vas vodi do Otoka mrtvog čovjeka (Dead Man's Island), i sad u ovom slučaju ste zaglavljeni na otoku!

Na kraju, vaša mapa treba da izgleda ovako:



Karte za demonstraciju aktivnosti



Karte za demonstraciju aktivnosti

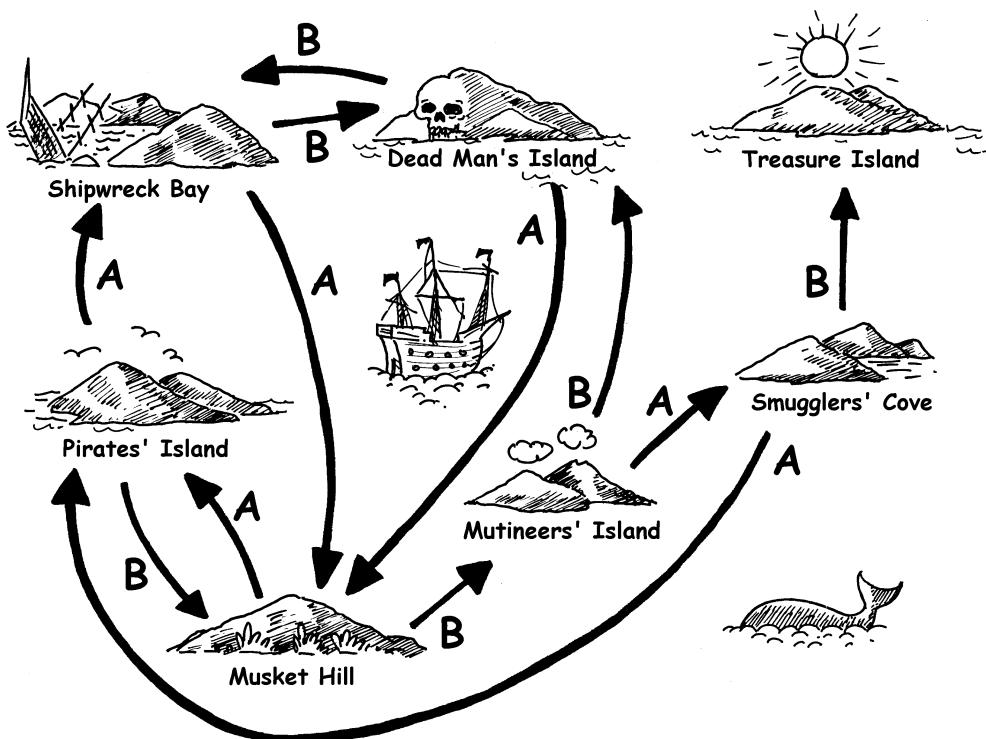


Aktivnost

Izaberite 7 učenika koji će biti "otoci". Učenici će držati karte koje odgovaraju njihovom otoku sakrivajući precizne naredbe koje se nalaze na stražnjoj strani. Neka se učenici na slučajan proizvoljan način rasporede u učionici ili igralištu. Ostali učenici će dobiti prazne mape i trebaju da putuju od Otoka gusara (Pirates' Island) do Otoka sa blagom (Treasure Island) bilježeći pažljivo pređeni put na svojim mapama. (Dobra je ideja slati učenike jedan po jedan kako ne bi mogli čuti od drugih učenika unaprijed o mogućim putevima.)

Za one koji brzo završe: Pokušajte pronaći više od jednog puta.

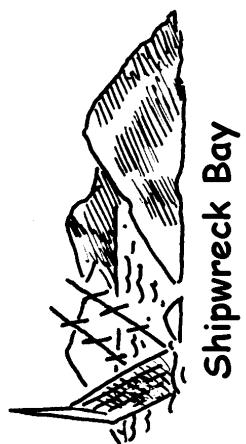
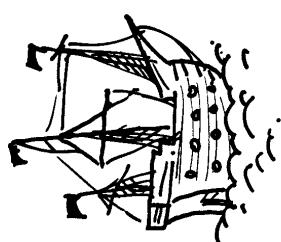
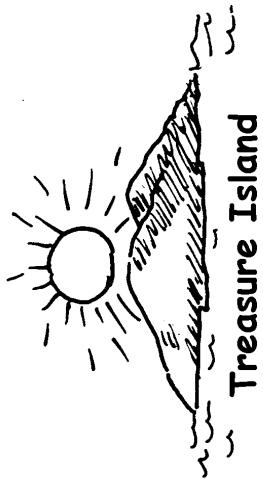
Kompletna mapa izgleda ovako:



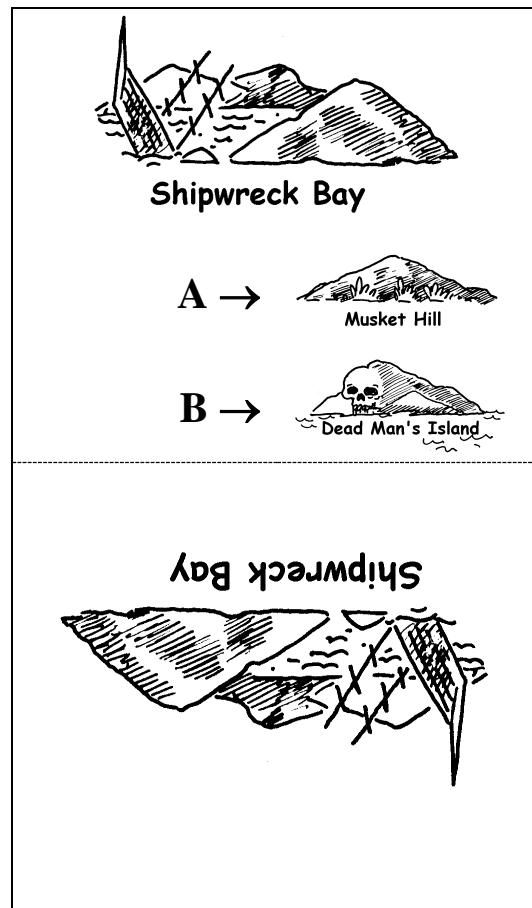
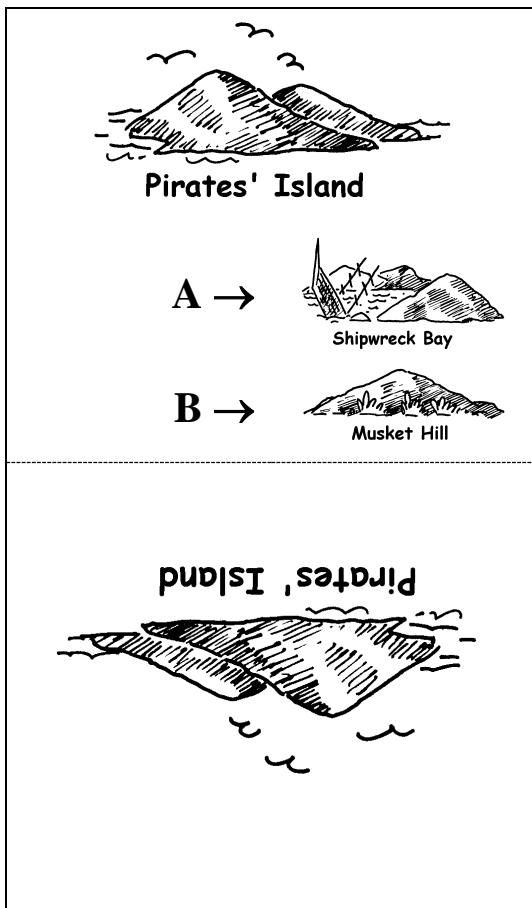
Diskusija koja slijedi aktivnost

Koji je najbrži put? Koji bi bio najsporiji put? Neki putevi mogu sadržavati i konture. Možete li pronaći primjer takvog puta? (Na primjer, BBBABAB i BBBABBABAB su dva puta koja oba vode do Ostrva sa blaogom (Treasure Island).)

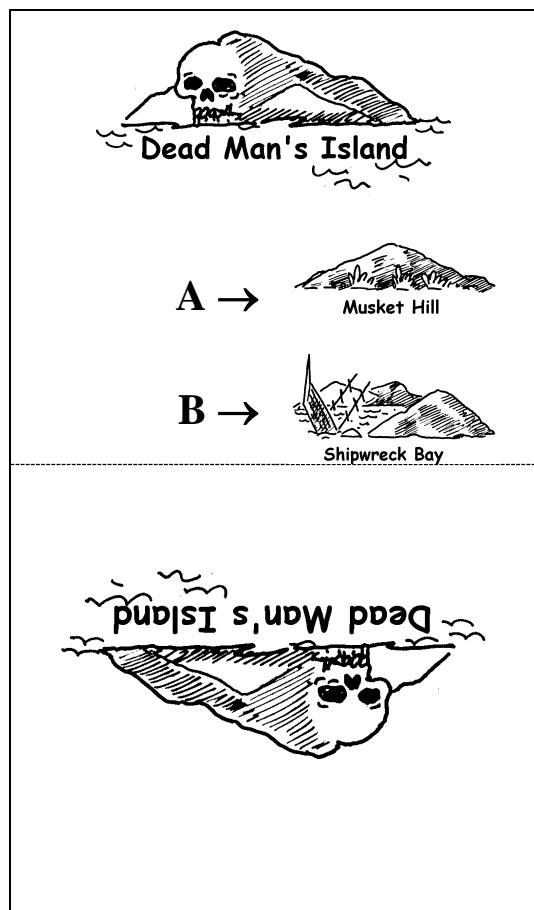
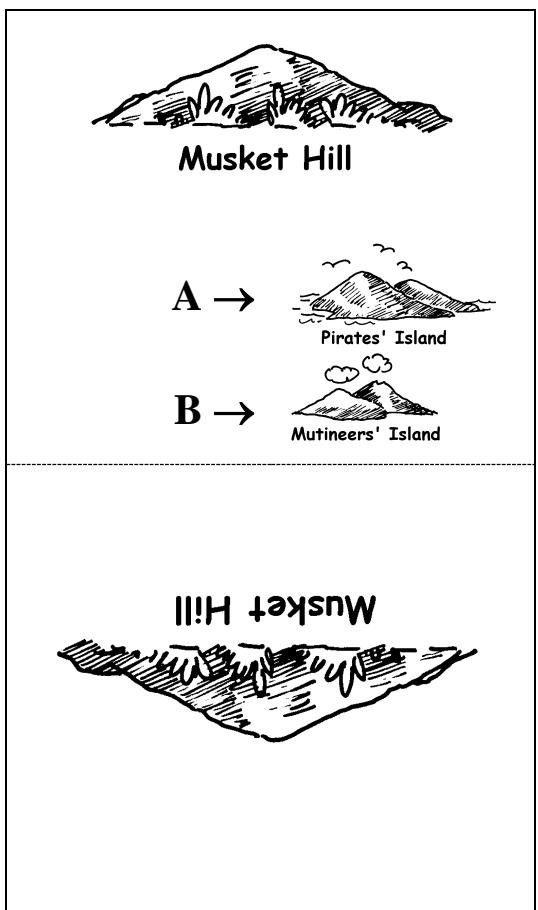
Radni list za aktivnost: Pronadite svoj put do trezora Otoka sa blagom



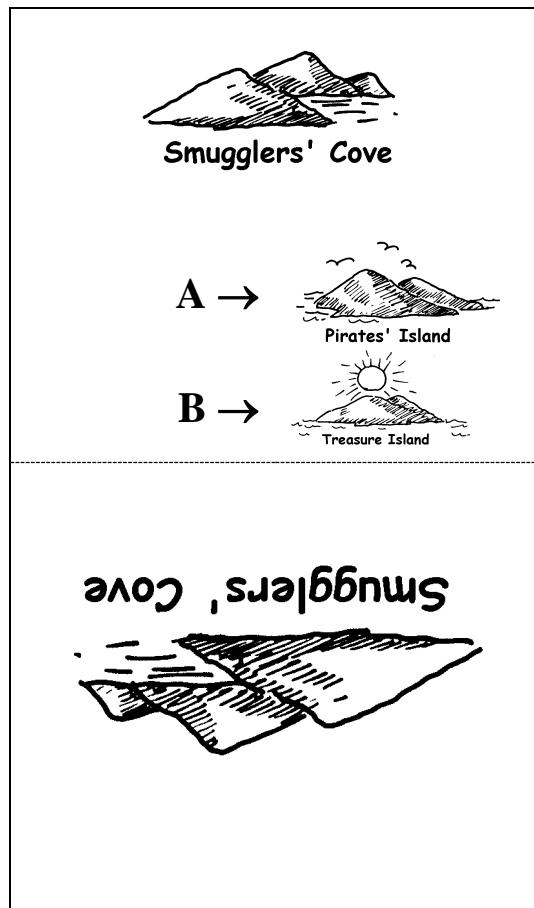
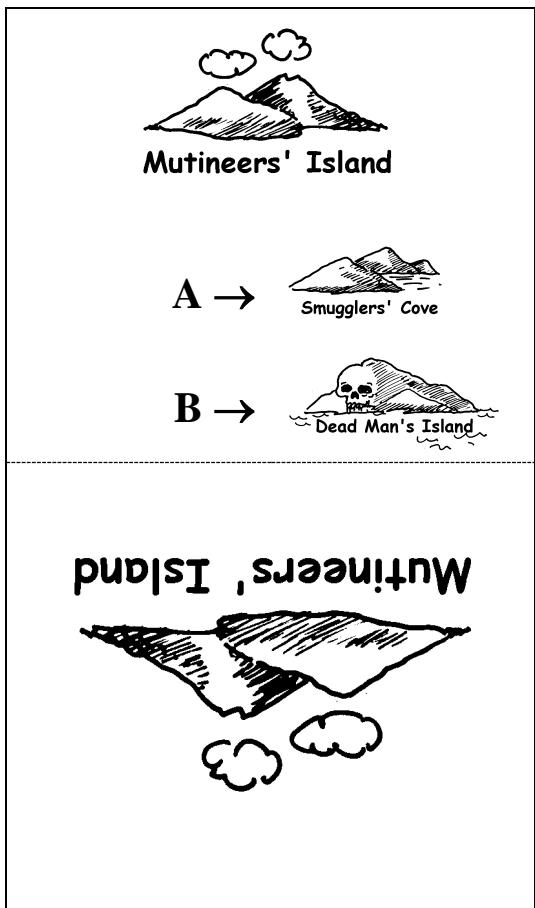
Uzorak za kopiranje: Karte za otoke(1/4)



Uzorak za kopiranje: Karte za otoke (2/4)



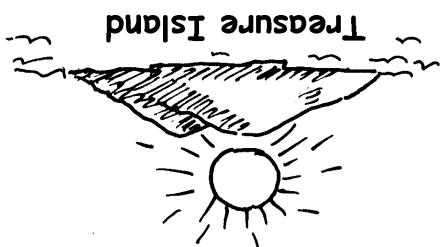
Uzorak za kopiranje: Karte za otoke (3/4)



Uzorak za kopiranje: Karte za otoke (4/4)

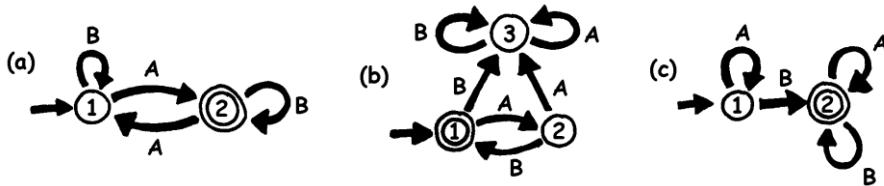


Congratulations!



Konačni automati

Jedan drugačiji način crtanja ovakvih mapa je sljedeći:



Otocci su predstavljeni numerisanim krugovima, i traženi otok (onaj sa blagom) je označen sa dva kruga. Kojim putevima možemo putovati kako bi stigli do traženog otoka? (Dobra ideja je istražiti ovo pitanje probajući različite primjere, na primjer Da li nas "A" void do stanja sa duplim krugom? "AA"? "ABA"? "AABA"? Koji je uopšteni uzorak puta?)

Rješenja:

Na mapi (a) ćemo završiti na duplom krugu (otok 2) samo ako niz ima neparan broj slova A (na primjer AB, BABAA, ili AAABABA).

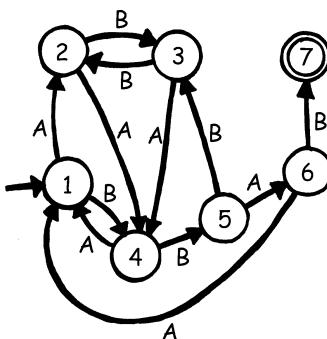
Na mapi (b) ćemo završiti na duplom krugu samo sa naizmjeničnim nizom slova A i B (AB, ABAB, ABABAB, ...).

Na mapi (c) je potrebno da niz sadrži najmanje jedno slovo B B (jedini nizovi koji nisu pogodni su A, AA, AAA, AAAA, ...).

Radni list za Aktivnost: Otok sa blagom

Možete li dobro sakriti vaš trezor sa blagom? Koliko teško možete učiniti traženje blaga? Sada je vrijeme da napravite svoju sopstvenu mapu!

- Ovdje dolje imamo malo složeniju verziju iste ideje predstavljanja jedne mape. Ova mapa je ista kao ona u prethodnom zadatku. Računarski stručnjaci koriste ovaj brzi i lagani način za predstavljanje potrebnih uzoraka.

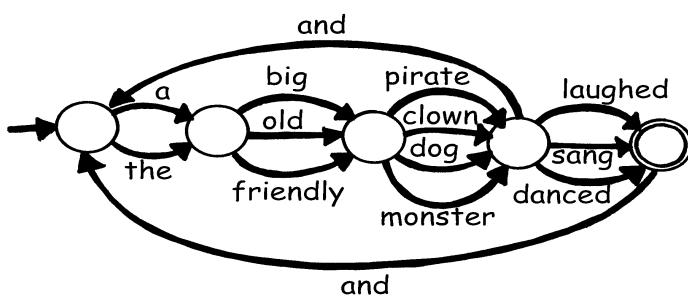


Nacrtajte vaš osnovni plan na isti način tako da jasno možete vidjeti kojim putevima vaši brodovi treba da idu pa onda napravite odgovarajuću praznu mapu sa kartama za svaki otok. Koji je najefikasniji niz puteva da bi došli do Otoka sa blagom?

- Koliko dobro vaši prijatelji mogu slijediti vašu mapu? Dajte im jedan niz slova A i B, i provjerite da li mogu doći do traženog otoka.

Možete napraviti više različitih igara i mozgalica zasnovanih na ovoj istoj ideji konačnih automata.

Ovo je jedan od načina konstruisanja rečenica izabirući na slučajninačin puteve kroz datu mapu i bilježeći riječi koje smo susreli uz put.



Sada pokušajte koristiti istu ideju sami. Možda je moguće napisati malu smiješnu priču koristeći isti metod!

Radni list za Aktivnost: Misteriozna Igra Novčića

Nekoliko prijatelja je preuzeo sa Interneta jednu igru gdje robot baca novčić a oni moraju odgjetnuti da li će sljedeći rezultat biti pismo ili glava. Na početku igra izgleda zaista jednostavna i lagana. Imaju najmanje 50/50 šansi za pobjedu—ili oni samo tako misle! Nakon nekog vremena provedenog u igri počinju sumnjati da je to tako. Izgleda da postoji neki sakriveni uzorak po kojem se novčići pojavljuju. Možda je cijela igra namještена. Naravno da nije! Odlučili su istražiti šta je istina. Armin je zapisao sve rezultate od dosadašnjih pokušaja u igri I ovo je ono što sada ima: (p = pismo, g = glava)

p p g p p g p p p g g p p p g g p g g p p p g p p p g g p g g p p p p g g p
g g g g p g g p g g p p g g p p p g p p p p g g p p p g g g p g p p p g g g
g g g

Da li možete pronaći sakriveni uzorak za dobro pogadanje bacanja?

Postoji izuzetno jednostavna mapa koja može opisati niz bacanja novčića. Pokušajte naći koja je to mapa. (**Pomoć:** ova mapa ima samo 5 ‘otoka’)

Zašto je ovo sve važno?

Konačni automati (eng. finite-state automata) se koriste u računarskim naukama da bi pomogle računarima obraditi niz karaktera ili događaja.

Jedan jednostavan primjer je kada nazovete jedan telefonski broj I dobijete poruku koja kaže "Pritisnite 1 za ovo ... Pritisnite 2 za ono ... Pritisnite 3 da bi razgovarali sa osobom." Vaše pritiskanje tastera na telefonu je ulaz za konačni automat na drugoj strani telefonske veze. Ovakav dijalog može biti jako jednostavan ali i prilično komplikovan. Ponekad ste vođeni u krug jer ima neka čudna kontura u pridruženom konačnom automatu. Ukoliko se to stvarno dogodi onda postoji greška u dizajniranju konačnog automata—i to je onda naravno veoma frustrirajuće za onoga ko poziva!

Drugi primjer je kada uzimate novac sa bankomata. Program koji se nalazi u računaru bankomata vas vodi kroz cijeli niz radnji. Unutar programa svi mogući nizovi radnji su zapisani kao končani automat. Svako dugme koje pritisnete vas vodi do sljedećeg stanja u automatu. Neka stanja imaju naredbe za računar u bankomatu, kao što je « isplatite 100 KM u novčanicama », ili « odštampajte račun » ili « izbacite bankarsku karticu ».

Neki računarski programi zaista rade i obrađuju rečenice (na engleskom ili nekom bosanskom ili nekom drugom jeziku) koristeći mapu koja je data na strani 115. Oni takođe mogu i sami kreirati rečenice kao što mogu obrađivati rečenice koje korisnik unosi (kuca ili izgovara). Šezdesetih godina prošlog vijeka računarski naučnici su napisali poznati program pod imenom "Eliza" (prema Eliza Dolittle) koji je mogao razgovarati sa ljudima. Program se pretvarao da je psihoterapeut i seansu je započinjao sa pitanjem kao što je: "Recite mi nešto o svojoj porodici" ili "Nastavite tako." Iako računar nije mogao "razumjeti" ništa bio je dovoljno vješt—a njegovi korisnici su bili dovoljno naivni—tako da su neki ljudi zaista povjerovali da razgovaraju sa pravim psihoterapeutom.

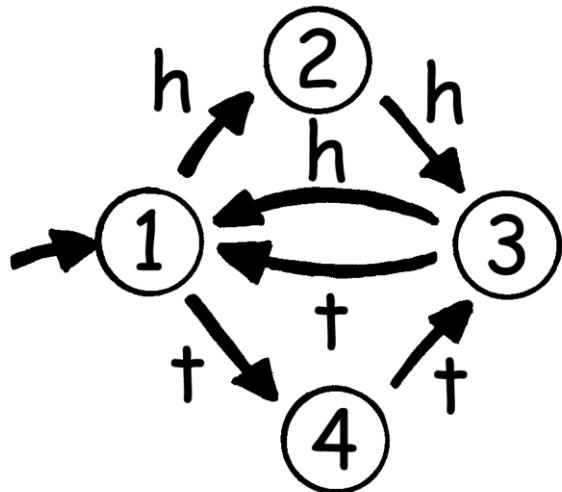
Iako računari nisu uvijek bili dobri u razumjevanju ljudskog govora uvijek su bili u stanju relativno dobro analizirati I obrađivati umjetne jezike. Jedan naročito važan tip umjetnih jezika su i programski jezici. Računari I ovdje koriste konačne automate kako bi pročitali programme I preveli ih u jednu formu elementarnih računarskih naredbi koje će se onda izvršiti direktno na samom računaru.



Rješenja i pomoć

Misteriozna igra novčića (strana 116)

Misteriozna igra novčića koristi ovu mapu za kreiranje rezultata bacanja novčića:



Ako bi je pomno pratili vidjeli bi da prva dva novčića od data tri imaju uvijek istu vrijednost.

Aktivnost 13

Marching Orders—Programski jezici

Sažetak

Računari su obično programirani koristeći neki “jezik” što je u stvari ograničen mali skup izraza – naredbi koje će računar izvršavati. Jedna od najfrustrirajućih stvari u vezi sa računarima da oni uvijek izvršavaju naredbe doslovno, čak i kada to proizvodi potpuno lude rezultate. Ova aktivnost će ponuditi učenicima iskustvo sa ovim posebnim aspektom programiranja.

Veze sa Curriculum-om

- ✓ Maternji jezik – uzajamno slušanje

Vještine

- ✓ Davati i izvršavati naredbe.

Dobna/starosna grupa

- ✓ 7 godina i više

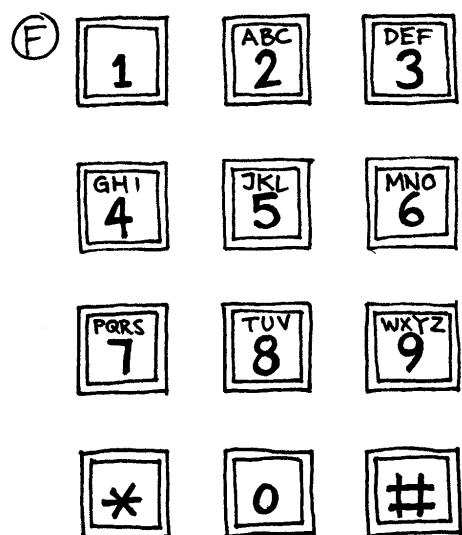
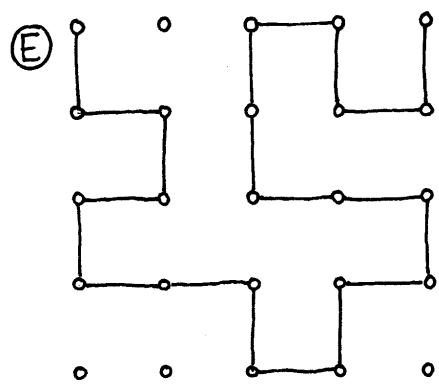
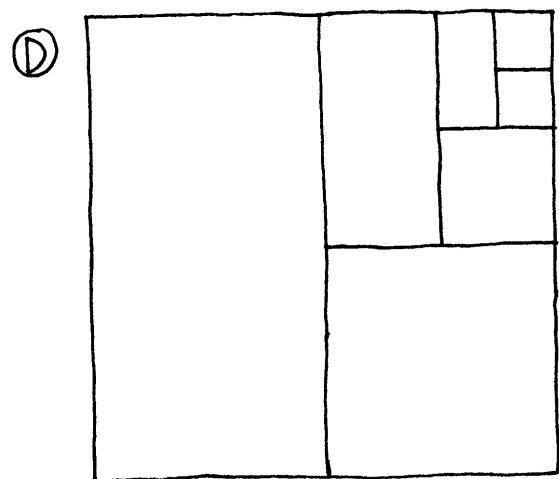
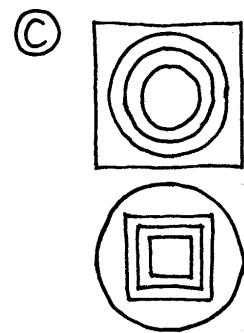
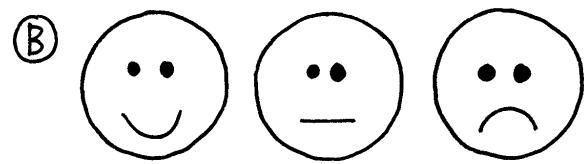
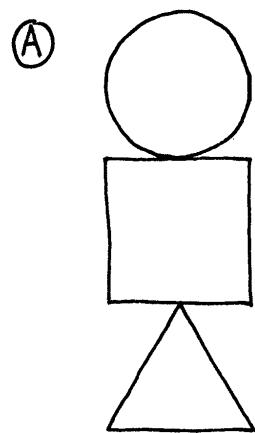
Materijali

Nastavniku će trebati:

- ✓ Karte sa slikama kao ona što je prikazana na sljedećoj stranici.

Svaki učenik će trebati:

- ✓ Olovku, papir i linijar



Marching Orders

Uvod

Razgovarajte sa učenicima koliko bi bilo dobro ako bi ljudi izvršavali sve naredbe tačno kako su date. Na primjer, šta bi se dogodilo ako bi pokazali na zatvorena vrata i rekli, "Prođite kroz ta vrata"?

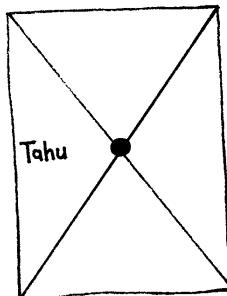
Računari rade prateći i izvršavajući niz instrukcija-naredbi, I oni izvršavaju tačno ono što svaka naredba kaže—čak i kada to nema nikakvog smisla!

Primjer za Demonstraciju

Provjerite da li učenici mogu nacrtati ovu sliku prateći date naredbe.

1. Nacrtajte tačku u centru vašeg papira.
2. Počevši od gornjeg lijevog ugla papira nacrtati pravu liniju kroz tačku i koja završava u donjem desnom uglu.
3. Počevši od lijevog donjeg ugla papira nacrtati pravu liniju kroz tačku i koja će završiti u gornjem desnom uglu.
4. Napišite svoje ime u trouglu u centru na lijevoj strani papira.

Rezultat bi trebao izgledati ovako:



Aktivnosti

Izaberite jednog učenika i dajte mu jednu sliku (jednu od onih primjera datih na strani 120). Učenik će opisati sliku cijelom razredu kako bi je svako mogao nacrtati. Učenici mogu postavljati pitanja kako bi dobili jasna uputstva. Cilj je provjeriti kako brzo i tačno zadatak može biti završen.

Ponovite zadatak ali ovaj put učenici niće biti u mogućnosti postavljati pitanja. Najbolje je koristiti jednostavnije slike za ovaj zadatak jer se učenici mogu vrlo brzo i lako izgubiti i zbuniti.

Sada probajte zadatak učenikom sakrivenim iza ekrana, bez mogućnosti postavljanja pitanja, na način da je sva moguća komunikacija u obliku naredbi.

Ukažite na to da ovaj vid komunikacije najviše liči na komunikaciju koju programmer ima kada piše jedan program za računar. Programeri daju jedan skup naredbi računaru, I ne poznaju tačno šta je rezultat datog skupa naredbi sve dok se te naredbe ne izvrše na računaru.

Zatražite sada od učenika da nacrtaju svoju sliku i da napišu niz naredbi za njeno reproduciranje. Pokušajte raditi u parovima ili sa cijelim razredom.

Varijacije

1. Napišite naredbe kako bi konstruisali avion od papira.
2. Napišite naredbe kako doći do tajne lokacije u školi koristeći naredbe kao što su “Idi naprijed x metara”, “skreni lijevo” (90 stepeni), i “skreni desno” (90 stepeni).

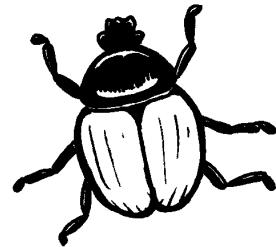
Učenici treba da testiraju i poboljšavaju svoj skup naredbi sve dok ne postignu željeni rezultat.
3. Slijepa igra. Zavežite crnim povezom jednog učenika i zatražite od ostalih da ga vode po učionici.

Zašto je ovo sve važno?

Računari funkcionišu tako što izvršavaju listu naredbi koju nazivamo program. Program je napisan da bi uradio računar uradio jedan zadatak. Programi se pišu u jezicima koji su specijalno dizajnirani, sa ograničenim skupom dostupnih naredbi, da bi saopšteli računarima šta uraditi. Neki programski jezici su pogodnije za određene potrebe od drugih programskega jezika.

Bez obzira na to koji je programski jezik koriste programeri moraju naučiti specificirati tačno ono što žele da računar uradi. Za razliku od ljudi računari će izvršavati naredbe doslovno čak i kad su te naredbe potpuno besmislene.

Jako je važno da su programi dobro napisani. Jedna mala greška može prouzročiti mnogo problema. Zamislite koja su posljedice jedne male greške u programu koji služi za lansiranje svemirskog broda, ili za upravljanje nuklearnom elektranom, ili za upravljanjem željezničkim saobraćajem! Greške u programima se ubičajeno nazivaju "bugs" (na engleskom buba) u čast (tako se barem kaže) jednog moljca koji je jednom bio uklonjen ("debugged") iz jednog električne sklopke ranih 1940 godina iz jedne elektronske mašine za računanje.



Što je program kompleksniji mogućnost postojanja grešaka je veća. Ovo je postalo veliki problem kada su SAD radile na programu razvoja Strateške Inicijative za Odbranu (eng. Strategic Defence Initiative) poznatije pod imenom Rat zvijezda (eng. "Star Wars"). Sistem je bio računarski kontrolisan i upravljan i trebao je biti neprobojan odbrambeni štit protiv bilo kakvog nuklearnog napada. Neki računarski naučnici su tvrdili da to nikada ne može dobro funkcionišati zbog kompleksnosti i unutrašnje, urođenje nesigurnosti i nesavršenosti potrebnog software-a. Software mora biti testiran vrlo pažljivo kako bi se pronašlo što je moguće više bug-ova, a nije moguće testirati jedan takav sistem jer bi se morale ispaljivati rakete na Sjedinjene Američke Države kako bi bili sigurni da sve radi kako treba!



Part IV

Zaista teški problemi—*Nerješivost*

Nerješivost

Da li postoje problem koji su teški čak i za same računare? Odgovor je da. Vidjećemo u Aktivnosti 20 da vođenje jednostavnog razgovora—neobavezan razgovor (eng. Chatting)—je nešto što računari ne mogu raditi, i ne zato što ne mogu razgovarati nego jer ne mogu razumjeti ili razmišljati o razumnim stvarima koje treba reći. Ipak to nije osnovni tip teškog problema koje računari ne znaju raditi i o kojima pričamo ovdje. Mi sami ne znamo opisati na koji način vodimo razgovor pa dakle nismo u stanju ni reći računarima kako da to urade. S druge strane, u ovom dijelu ćemo govoriti o problemima za koje je lagano reći računarima šta da rade—pišući za to prigodan program—ali računari ne mogu uraditi ono što želimo jer bi taj rad uzimao zaista puno vremena: možda i milione vjekova. Neće pomoći ni ako bi kupili brže računare: kada bi računar bio 100 puta brži još uvijek bi bili potrebne milioni godina, a čak i kad bi mogli kupiti računar milion puta brži potrebno bi bilo stotine godina da dobijemo rezultat. To je ono što nazivamo teškim problemom—problem za čije rješavanje nam treba mnogo više nego je prosječan ljudski vijek čaki i kada bi imali najbrži mogući računar!

Aktivnosti u Dijelu II o algoritmima sun am pokazale kako je moguće naći bolje i efikasnije načine da računar završi neki zadatak. U ovom dijelu ćemo razmatrati problem za koje nijedno efikasno rješenje nije poznato, probleme za čije rješavanje bi nam trebali milioni godina računanja. Takođe ćemo vidjeti šta je to jedna od najvećih tajni u računarskim naukama danas: to je da *niko ne zna* da li postoji efikasniji način za rješavanje ovih problema! Moguće je da jednostavno do sada još niko nije došao sa nekim dobrim načinom njihovog rješavanja, a moguće je takođe da ne postoji neki dobar način. Mi sada jednostavno ne znamo šta je tačno. I to nije sve. Postoje hiljade problema, koji iako svi izgledaju potpuno različiti, koji su ekvivalentni u smislu da ako postoji efikasan metod za rješavanje jednog takvog problema onda se taj metod može pretvoriti u efikasan metod za rješavanje svih ovih problema. U ovim aktivnostima ćete nešto naučiti o takvim problemima.

Za nastavnike

Predviđene su ukupno tri aktivnosti u ovom dijelu. Prva se bavi bojenjem karata i određivanjem koliki broj boja je potreban tako da su boje susjednih zemalja različite. Druga aktivnost podrazumjeva mogućnost korištenja jednostavne mape grada sa ulicama i postavljanje prodavača sladoleda na uglovima tako da niko nema potrebu da ide predaleko kako bi kupio sebi sladoled. Treća aktivnost se izvodi napolju i koristi konopce i štipaljke istražujući kako kreirati malu mrežu koja spaja dati skup tačaka.

Ove aktivnosti na praktičan i slikovit način predstavljaju osnovnu ideju kompleksnosti—kako problem koji su izuzetno jednostavni za opisati i zadati mogu biti jako, za ne povjerovati, teški za rješavanje. I ovi problemi nisu uopšte teški za razumjevanje. To su vrlo često praktična pitanja koja vrlo često dolaze sama u svakodnevnom životu kao što je izrada karata, izrada školskog rasporeda ili izgradnja mreže puteva. Kompjutaciona teoretska osnova cijele teorije se zasniva na pojmu koji nazivamo “NP-kompletost” koja je objašnjena u dijelovima *Zašto je ovo sve važno?* na kraju sveke od tri aktivnosti. Iako se sve ove aktivnosti mogu raditi u proizvoljnem redoslijedu dijelovi koji objašnjavaju širu sliku su predviđeni za čitanje I studiranje u redoslijedu u kojem se pojavljuju. Kada stignete do kraja imaćete solidno znanje i razumjevanje o tome šta su najznačajnija otvorena pitanja računarskih nauka danas.

Tehnički naziv za ovaj dio je “nerješivost” (eng. “intractability”) jer se problemi koji su teški za rješavanje nazivaju nerjesivi (eng. *Intractable*). Sama engleska riječ potiče od latinske riječi *tractare* što označava crtati ili vući pa tako dolazimo do njene moderne upotrebe u riječi *tractable* koji označava lagan za upravljanje, savijanje, ili onaj što se povinuje naredbama. Nerjesivi (eng. Intractable) problemi su oni koji nisu lagani za manipulisanje jer bi bilo potrebno mnogo vremena da dobijemo odgovor (rješenje) za njih. Iako može izgledati nerazumljivo i rijetko, nerješivost je od velikog praktičnog interesa jer bi bilo kakva nova otkrića u ovoj oblasti imala velike posljedice na mnogim poljima istraživanja i nauke. Na primjer, većina kriptografskih kodova i Sistema se zasniva na nerješivosti nekih problema pa bi tako kriminalac koji uspije naći efikasno rješenje za njih mogao lagano dekodirati sve tajne poruke i prodavati ih, ili—još jednostavnije—samo napraviti krivotvorene bankarske transakcije. Više ćemo o ovome reći u dijelu V—Kriptografija.

Aktivnost 14

Slabi kartograf—Bojenje grafova

Sažetak

Mnogi problem optimizacije uključuju situacije kada se određeni događaji ne mogu desiti u isto vrijeme, ili kada neki elementi datog skupa ne mogu biti susjedni jedan drugom. Na primer, bilo ko ko je probao da napravi raspored časova ili rasporedi sastanke suočio se sa problemom zadovoljavanja uslova za sve uključene učesnike. Mnoge od ovih poteškoća se jasno mogu predstaviti koristeći model bojenja karata gdje je potrebno izabrati određenu boju za svaku zemlju na mapi tako da su susjedne zemlje obojene različitim bojama. Ova aktivnost je o tom problem.

Veze sa Curriculum-om

- ✓ Matematika: Broj – Izražavanje brojeva u drugim bazama. Predstavljanje brojeva u bazi dva.
- ✓ Matematika: Algebra – Nastaviti dati niz prema uzorku, I opisati pravilo za dati uzorak. Uzorci i relacije u stepenu broja dva.

Vještine

- ✓ Rješavanje problema
- ✓ Logičko razmišljanje
- ✓ Algoritamske procedure i kompleksnost.
- ✓ Saopštavanje svojih osobnih otkrića.

Dobna/starosna grupa

- ✓ 7 godina i više

Materijali

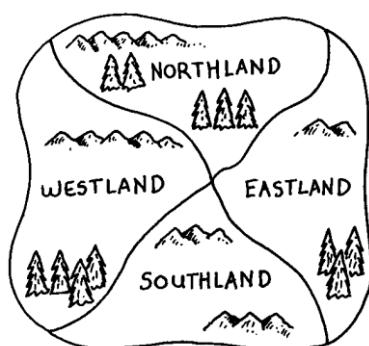
- ✓ Školska table, ili slična površina za crtanje.
Svaki učenik će trebati:
- ✓ Jednu ili više kopija radnog lista,
- ✓ Mali označavači boja (na primjer jednostavnii brojači, ili counters or poker chips), and
- ✓ Četiri olovke različitih boja (ili krede u boji, ili flomastera itd.)

Bojenje grafova



Uvod

Ova aktivnost se odvija oko priče u kojoj učenici treba da pomognu jednom kartografu, ili drugim rječima dizajneru mapa, koji pokušava da oboji zemlje na jednoj mapi. Nije bitno koje je boje pojedina zemlja, jedino je važno da je ta boja različita od boja svih susjednih zemalja.



Na primjer, ova mapa prikazuje četiri zemlje. Ukoliko obojimo Northland crveno, pa onda Westland i Eastland ne mogu biti crveni jer se onda njihove granice sa Northland ne bi mogle jasno vidjeti. Možemo dakle Westland obojiti u zeleno i isto tako bi bilo prihvatljivo obojiti Eastland zelenom jer ta zemlja nema zajedničke granice sa Westland. (Ako se dvije zemlje susreću samo u jednoj tački onda se ne smatra da imaju zajedničku granicu pa prema tome mogu biti obojene istom bojom.) Southland se može obojiti crvenom pa bi tako na kraju završili sa dvije boje koje su dovoljne da se oboji cijela mapa.

U našoj prići kartograf je siromašan i ne može sebi priuštiti mnogo različitih boja tako da je poželjno da koristi što je moguće manje boja.

Diskusija

Opisati problem na kojem će učenici raditi pokazujući process bojenja jedne mape na školskoj tabli.

Podijelite po jednu kopiju prvog radnog lista svakom učeniku. Ova mapa se može obojiti koristeći samo dvije boje. Iako ograničenje da se koriste najviše dvije boje može izgledati

naročito izazovno i teško taj zadatak je mnogo lakši ako se usporedi sa bojenjem mape koja zahtjeva više od dvije boje. Osnovni razlog za lakoću tog uproštenog problema je mali izbor boja za svaku zemlju (jedna od dvije boje).

Zatražite od učenika da probaju obojiti mapu koristeći samo dvije boje. U toku procesa bojenja moguće je da će sami otkriti pravilo “mora-bit”: nakon što obojimo jednu zemlju jednom bojom svi njeni susjadi moraju biti obojeni drugom bojom. Ovo pravilo se može primjenjivati neprestano sve dok se ne oboje sve zemlje. Najbolje je ako učenici mogu sami otkriti ovo pravilo umjesto da im vi to kažete jer će im onda taj process razmišljanja dati mogućnost dodatnog razumjevanja samog problema.

Nakon što učenici završe neki zadatak mogu odmah nakon toga pokušati uraditi i sljedeći.

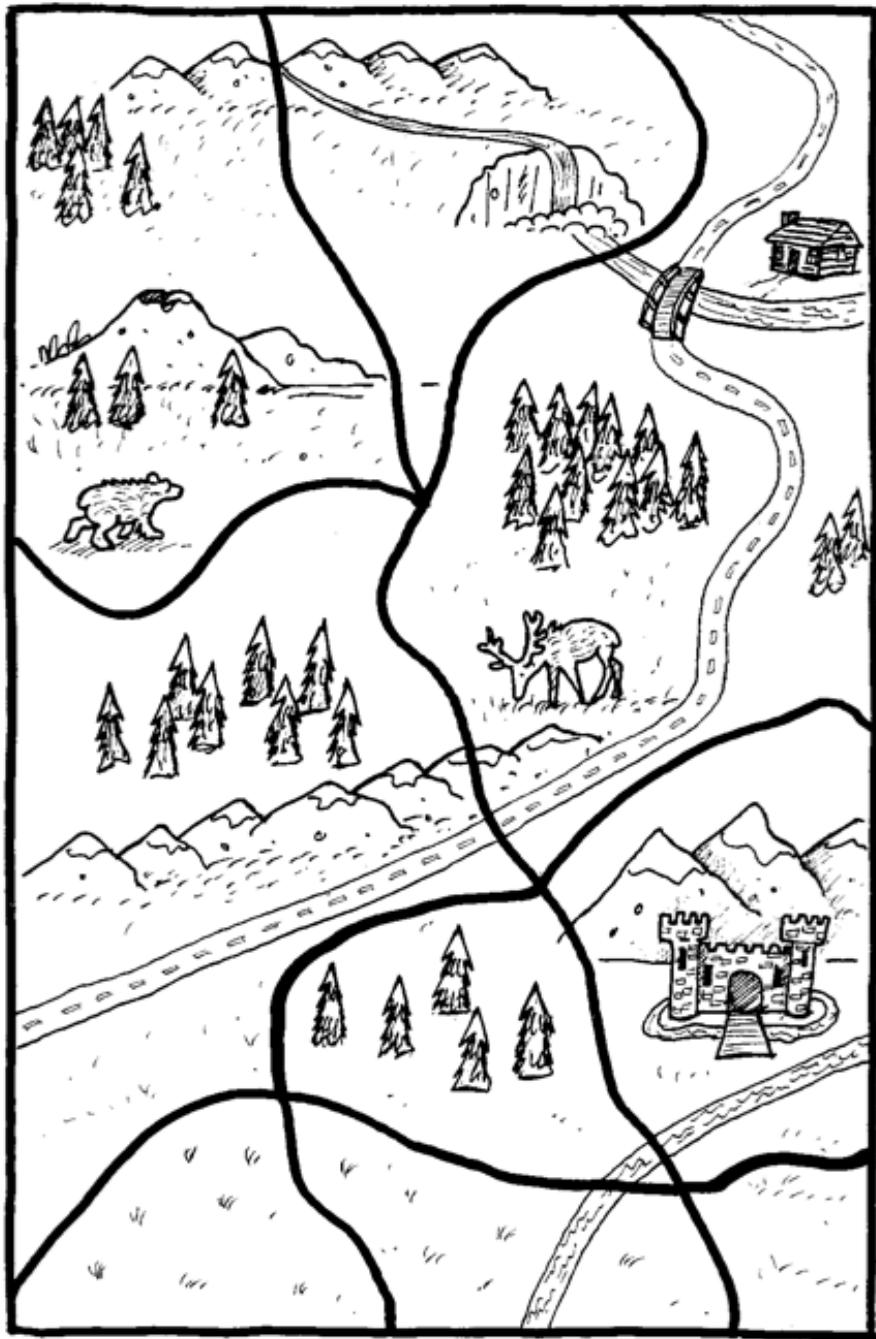
Učenici takođe mogu sami otkriti da je puno bolje koristiti jednostavno oznake za boje, na primjer numerisane figure, umjesto da se zemlje zaista i boje jer to omogućava da promijenimo već dodjeljene boje ako nam se neki novi način učini boljim.

Starije učenike možete pitati da objasne kako znaju da su dostigli minimalan broj boja potrebnih za bojenje jedne mape. Na primjer, najmanje tri boje su neophodne za bojenje ove mape jer ona sadrži jednu grupu od tri zemlje tako da svaka od njih ima granicu sa ostale dvije.

Ukoliko učenik završi sa svim zadacima ranije nego ostali zatražite od njih da kreiraju mapu koja zahtjeva korištenje pet različitih boja. Dokazano je (matematički) da se bilo koja mapa može obojiti koristeći samo četiri boje tako da će ih ovaj zadatak držati “zaposlene” neko vrijeme! Prema našem iskustvu, učenici će brzo pronaći mapi za koje vjeruju na početku da zahtjevaju pet boja ali naravno da je uvijek moguće pravilno obojiti takvu mapu koristeći samo četiri boje.

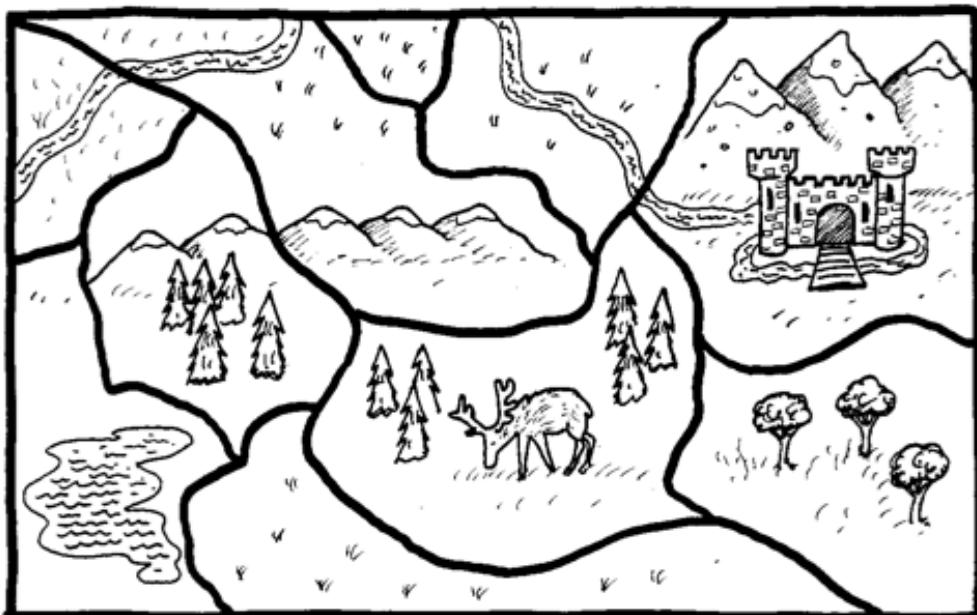
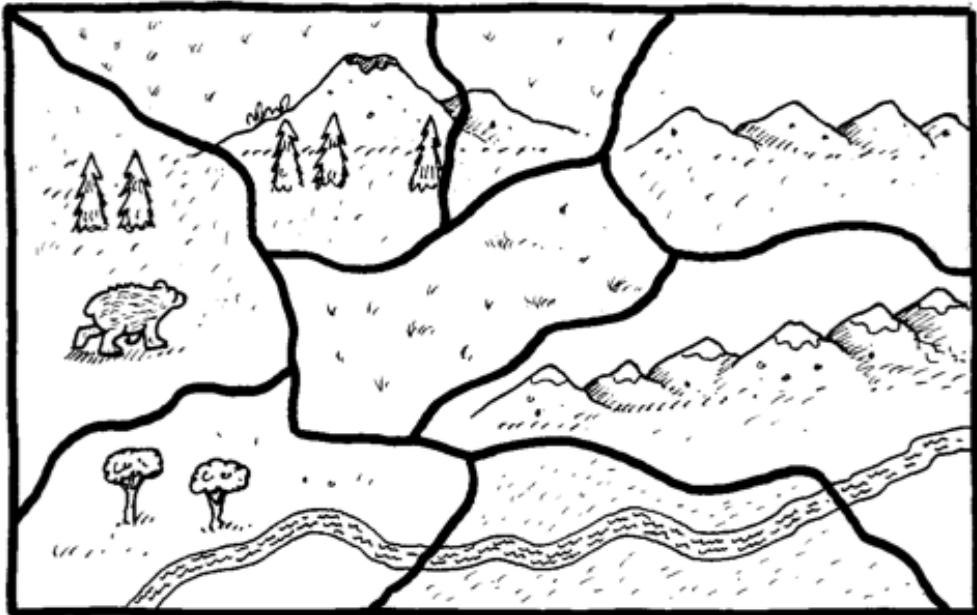
Radni list za Aktivnost: Bojenje grafova 1

Obojiti sve zemlje ove mape koristeći najmanji mogući broj boja pazeci pri tome da nikoje dvije zemlje koje imaju zajedničku granicu nisu obojene istom bojom.



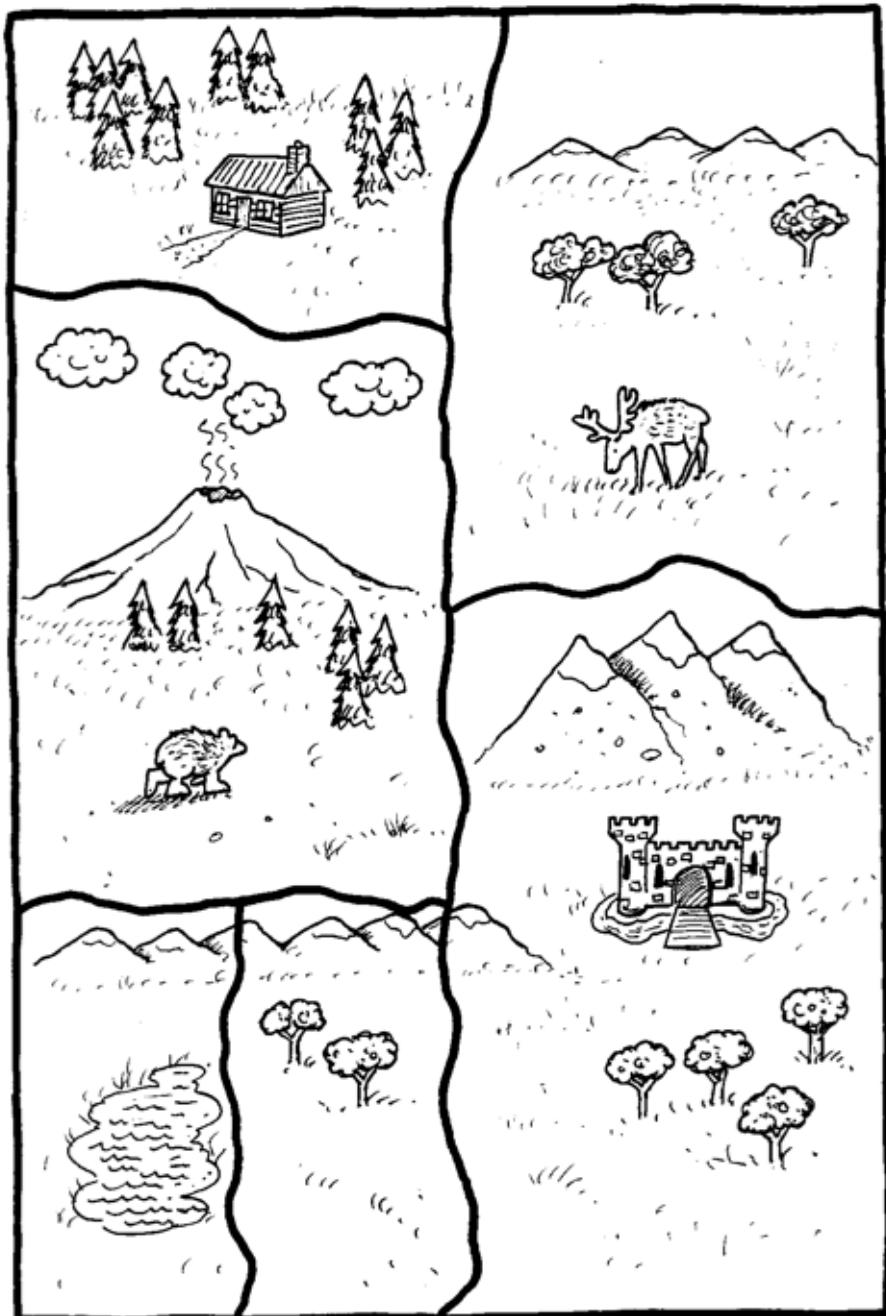
Radni list za Aktivnost: Bojenje grafova 2

Obojiti sve zemlje ove mape koristeći najmanji mogući broj boja pazeći pri tome da nikoje dvije zemlje koje imaju zajedničku granicu nisu obojene istom bojom.



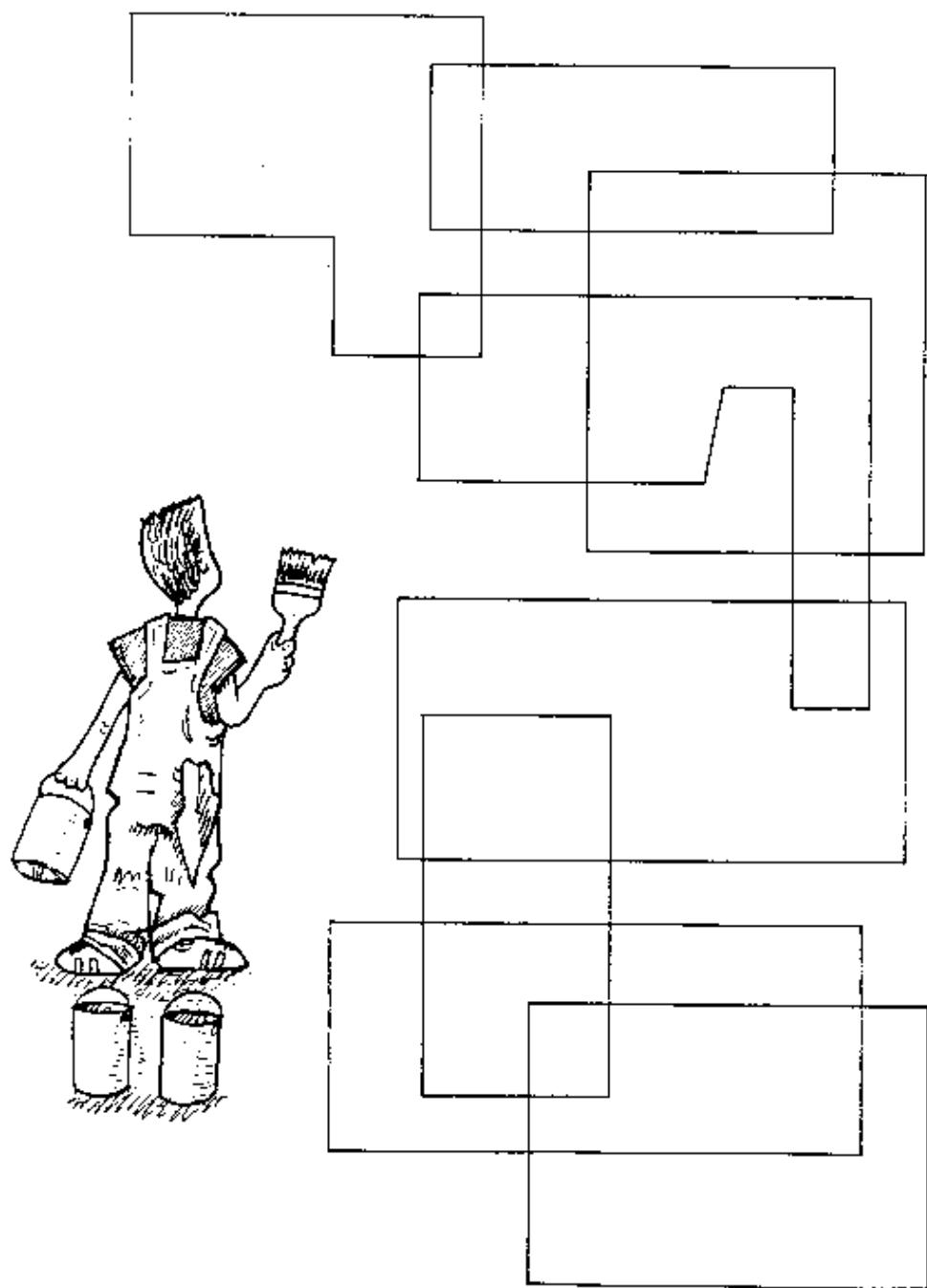
Radni list za Aktivnost: Bojenje grafova 3

Obojiti sve zemlje ove mape koristeći najmanji mogući broj boja pazeći pri tome da nikoje dvije zemlje koje imaju zajedničku granicu nisu obojene istom bojom.



Radni list za Aktivnost: Bojenje grafova 4

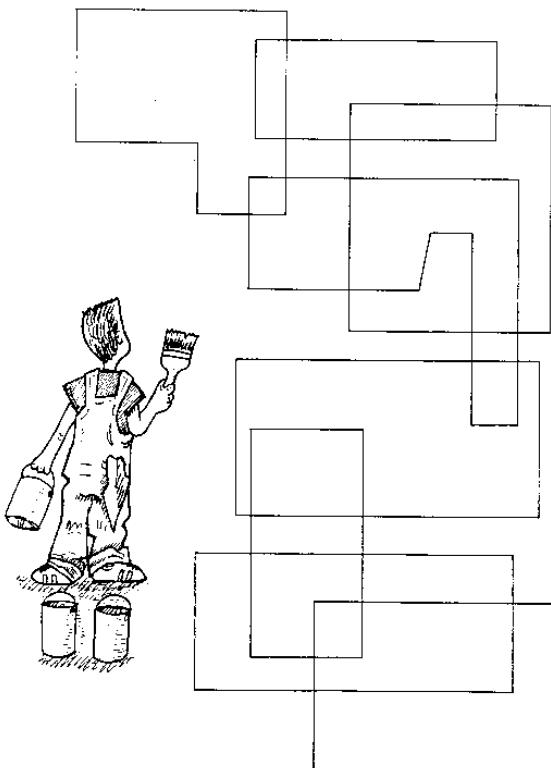
Obojiti sve zemlje ove mape koristeći najmanji mogući broj boja pazeći pri tome da nikoje dvije zemlje koje imaju zajedničku granicu nisu obojene istom bojom.



Varijacije i Proširenja

Postoji jedan jednostavan način da se konstruišu mape koje zahtjevaju samo dvije boje, upravo kako je to pokazano ovdje. Ova mapa je nacrtana tako što se preklapaju zatvorene krive linije (linije čiji početak se sastaje sa njenim krajem). Možete nacrtati bilo koji broj ovih krivih, u bilo kojem obliku koji želite, i slagati ih jednu na drugu kako želite i uvijek ćete završiti sa mapom koju možete obojiti samo sa dvije boje. Učenici mogu eksperimentisati sa ovim tipom mapa kreirajući ih po svojoj volji.

Četiri boje su ovdje dovoljne da se pravilon oboji mapa nacrtana na listu papira ili na lopti (to jest, jedan globus). Neko se može upitati (i naučnici su plaćeni da se pitaju takve stvari) koliko boja je potrebno da se oboje mape na nekoj čudnoj, neobičnoj površini (na primjer, na jednoj krofni sa rupom u sredini). U tom slučaju potrebno bi bilo najviše pet boja za bilo kakvu mapu. Učenici mogu probati sami eksperimentisati sa ovim.



Ima mnogo zanimljivih varijacija problema bojenja mape gdje svaki takav problem void u smjeru gdje je malo šta poznato sa sigurnošću. Na primjer, ukoliko bojim neku mapu na jednom listu papira onda sam siguran, ako sve radim na pametan način, da su mi četiri boje dovoljne. Ali prepostavimo da umjesto da radim sam bojim mapu sa svojim partnerom koji nije tako iskusan i izvježban (ili čak želi da mi oteža zadatak) i to tako da naizmjenično bojimo po jednu zemlju na mapi. Prepostavimo da ja radim na pametan način i najbolje što mogu dok moj partner radi samo ono što je "dozvoljeno" kada svakome dođe njegov red za bojenje. Koliko boja je potrebno imati na stolu da bi ja, u svom svom znanju i izvježbanosti, mogao primorati svog partnera da uvijek ima izbor nekog legalnog bojenja i da ne može na pametan način opstruirati cijeli process bojenja. Maksimalan broj boja nije poznat! Godine 1992 dokazano je da će uvijek 33 boje biti dovoljne a u 2008 godine ovaj broj (i dokaz) je poboljšan tako da znamo da će 17 boja biti dovoljno. Još uvijek ne znamo da uvijek postoji slučaj kada nam treba upravo 17 boja. (Eksperti, poznavaoци problema osjećaju i procjenjuju da je dovoljno 10 boja ali nemaju dokaz.) Učenici se mogu zabaviti imitirajući ovu situaciju na svojim mapama i radeći u parovima. Suparniku je cilj da maksimizira broj boja koje su potrebne protivniku da oboji cijelu mapu.

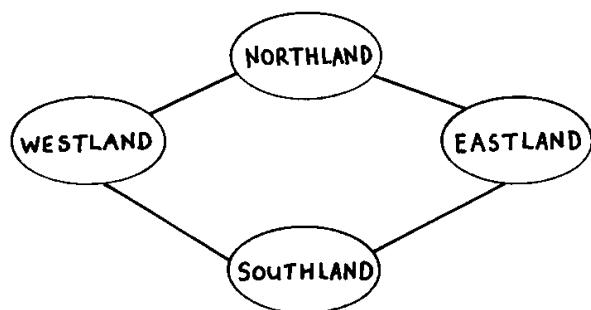
Jedna druga varijacija problema bojenja mape je poznata I pod nazivom bojenje imperije (eng. *empire coloring*), i ovdje počinjemo sa dvije različite mape na dva lista papira tako da imamo jednak broj zemalja na obje mape. Svaka zemlja na jednoj od mapa (nazovimo je na primjer Zemaljska kugla) je pridružena, uparena sa tačno jednom zemljom na drugoj

mapi (što može predstavljati kolonije na Mjesecu). Pored uobičajenih zahtjeva i uslova za bojenje mapa, to jest bojenje dvije susjedne zemlja različitim bojama, dodajemo i uslov da svaka zemlja na Zemaljskoj kugli mora biti obojena istom bojom kao i njena uparena zemlja na Mjesecu. Koliko boja nam je potrebno za rješenje ovog problema ? Odgovor na ovo pitanje je, trenutno, nepoznat.

Zašto je ovo sve važno?

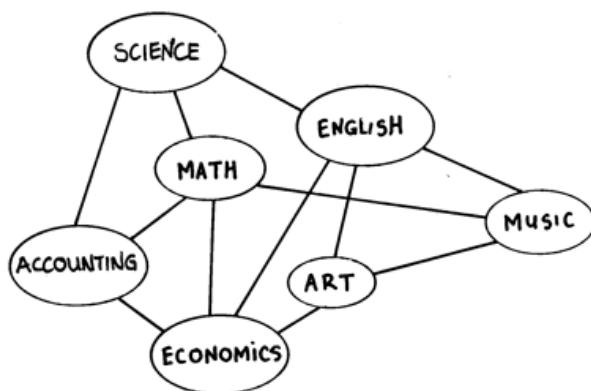
Problem bojenja mape koji smo istraživali u ovoj aktivnosti se, u principu, sastoji od traženja minimalnog broja boja—dvije, tri, ili četiri—koje su potrebne da bi pravilno obojili neku određenu mapu. Pretpostavka (konjektura, eng. conjecture) da svaku mapu možemo pravilno obojiti koristeći samo četiri boje je formulisana već 1852 godine ali nije bila dokazana sve do 1976 godine. Računarske nauke su izuzetno bogate nerješenim problemima a podatak da je teorem o četiri boje dokazan tek nakon 120 godina pažljivog rada istraživača je ohrabrujući za one koje rade na drugim problemima čija rješenja nisu bili u stanju naći desetljećima.

Problem bojenja mape pripada jednoj uopštenijoj klasi problema koje nazivamo “bojenje grafova.” (eng. graph coloring). U računarskim naukama jedan graf je abstraktan prikaz skupa relacija kao što je to prikazano ovdje na dijagramu.



Kao što je već rečeno u Aktivnosti 9, Blatnjavi grad, pojam grafa se koristi i u matematici da bi se označili dijagrami koji u koordinatnom sistemu predstavljaju neke brojevne podatke ali za računarske stručnjake pojам grafa ima jedno drugačije značenje. U računarskim naukama, grafovi se predstavljaju koristeći kružiće ili velike tačke, koje onda tehnički nazivamo čvorovima (ili vrhovima), i predstavljaju objekte a linije koje povezuju čvorove predstavljaju određenu vrstu relacije između tih objekata. Graf na slici gore predstavlja mapu sa početka ove aktivnosti. Čvorovi predstavljaju zemlje a linija između dva čvora označava da te dvije zemlje imaju zajedničku granicu. Na ovom grafu pravilo za dobro bojenje čvorova je da nikоja dva povezana čvora ne mogu imati istu boju. Za razliku od problema bojenja jedne mape, ne postoji procjene ili ograničenja na broj boja potrebnih da se dobro oboji jedan graf. Naime, na jednom grafu je moguće dodavajući veze među čvorovima izraziti veliki broj ograničenja dok crtanje jedne mape na dvodimenzionalnom papiru značajno ograničava tipove i broj uslova koji se mogu iskazati. Problem bojenja grafa je problem nalaženja najmanjeg broja boja potrebnih da se oboji određeni graf.

Na grafu datom ovdje sa desne strane čvorovi odgovaraju predmetima u školi. Linija između dva predmeta označava da ima najmanje jedan učenik koji prati oba predmeta pa onda oni ne mogu biti u isto vrijeme. Koristeći ovakvo predstavljanje, problem nalaženja rasporeda časova koji koristi najmanji broj različitih časova (perioda kada se drži nastava) u potpunosti odgovara



(ekvivalentan je) problemu bojenja pridruženog grafa. U tom slučaju, jedna boja odgovara jednom času to jest jednom vremenskom periodu kada se odvija nastava. Algoritmi za bojenje grafova su od velikog interesa za računarske nauke i koriste su u mnogim i različitim problemima iz stvarnog svijeta (eng. real-world problem) iako se vjerojatno nikada ne koriste i za bojenje pravih mapa, geografskih karata!—naš jadni kartograf je samo čista fikcija.

Postoji doslovno na hiljade drugih problema vezanih za grafove. Neki od njih su opisani na drugim mjestima u ovoj knjizi, kao što je problem najmanjeg pokrivačeg drveta u Aktivnosti 9 ili dominirajući skup u Aktivnosti 15. Grafovi su vrlo uopšten način predstavljanja podataka i mogu se iskoristiti za predstavljanje raznolikih situacija i problema kao što su mape puteva i raskrsnica, veze između atoma u jednoj molekuli, putevi koje jedna poruka može preći kroz datu računarsku mrežu, veze između elektronskih komponenti na nekoj štampanoj ploči, odnose između skupa zadatka koje treba završiti da bi se okončao jedan veliki projekt. Upravo iz ovog razloga problemi koji uključuju modele sa grafovima već dugo vremena zadržavaju i interesuju računarske naučnike.

Mnogi od ovih problema su jako teški—nisu teški za razumjevanje i odgovarajući koncepti nisu komplikovani, ali su teški jer njihovo rješavanje zahtjeva jako puno vremena. Na primjer, da bi odredili najbolje moguće rješenje za problem bojenja grafa srednje veličine—kao što je na primjer nalaženje rasporeda časova u jednoj školi gdje imamo 30 nastavnika i oko 800 učenika—mogu trebati godine, pa čak i stotine godina, na najbržem dostupnom računaru koristeći najbolji dostupan algoritam. Rješenje za problem bi bilo potpuno besmisленo u trenutku kada se nađe—a i to pod pretpostavkom da se računar u međuvremenu ne pokvari ili jednostavno istroši prije nego završi rješavanje. Ovakve probleme u stvarnosti rješavamo, i ima smisla rješavati, samo zato što smo zadovoljni i sa podoptimalnim (slabijim od optimalnog, eng. sub-optimal) ali još uvijek relativno dobrim rješenjima. Ukoliko bi strogo zahtjevali garancije da imamo najbolje moguće rješenje za problem onda bi takav problem postao praktično potpuno nerješiv za nas.

Vrijeme potrebno računaru da riješi problem bojenje raste eksponencijalno u odnosu kako raste veličina grafa. Posmatrajmo problem bojenja mape. Jedan od načina za njegovo rješavanje je isprobati sva moguća obojenja mape. Kako znamo da su potrebne najviše četiri boje za obojenje mape onda je potrebno evaluirati svaku moguću kombinaciju pridruživanje te četiri boje pojedinim zemljama. Ukoliko imamo ukupno n zemalja onda ima ukupno 4^n kombinacija. Ovaj broj raste jako brzo: svaki put kada dodamo jednu zemlju mapi broj kombinacija se poveća četiri puta pa prema tome učetverostručava kompjutaciono vrijeme potrebno za nalaženje rješenja. Čak i kada bi bili u stanju napraviti računar koji može rješavati problema sa, recimo, pedeset zemalja za samo jedan sat, dodavanje samo jedne zemlje bi povećalo potrebno vrijeme na četiri sata i dovoljno bi bilo dodati samo 10 novih zemalja pa bi onda ovaj super dobar računar trebao više od cijele godine da nađe najbolje rješenje. Možemo dakle zaključiti da problem sa vremenom potrebnim za izračunavanje svih kombinacija neće nestati samo zato što pravimo sve brže i brže računare!

Problem bojenja grafova je jedan dobar primjer problema čije kompjutaciono vrijeme rješavanja raste eksponencijalno. Za neki vrlo jednostavan primjer problema (kažemo za instancu problema) kao što su bile sve mape koje smo koristili u ovoj Aktivnosti prilično je lagano naći optimalno rješenje ali čim broj zemalja poraste iznad deset problem postaje jako težak za rješavanje ručno a sa stotinama i više zemalja i jedan jako dobar računar

može trebati mnogo godina da isproba sve moguće načine bojenja takve mapu kako bi onda mogao izabrati najbolji, optimalan način.

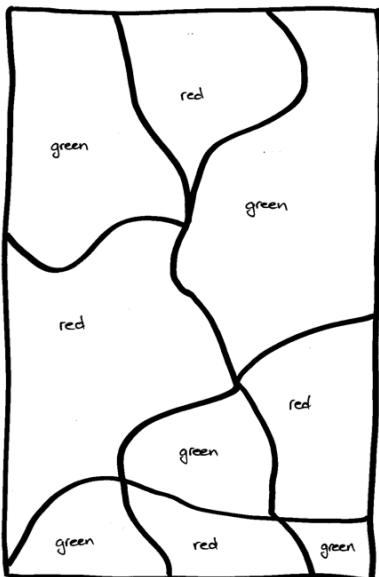
Mnogo problema iz stvarnog života su takvi ali se ipak moraju rješiti na neki način. Računarski stručnjaci onda pribjegavaju metodama koje daju dobre, ali ne možda i ne uvijek, i optimalne rezultate. Ove heurističke (eng. *heuristic*) tehnike daju rješenja koja su jako blizu optimalnim, vrlo brzi za izračunavanje. Takođe tako dobivena rješenja su tako blizu optimalnim da su skoro uvijek dovoljno dobra za sve praktične primjene. Na primjer, škola može sebi priuštiti korištenje jedne dodatne učionice u odnosu na broj učionica ako bi raspored časova bio optimalan, a isto tako jadni kartograf može sebi jednostavno priuštiti jednu dodatnu boju iako možda ona ne bi bila neophodna u optimalnom rješenju.

Niko do sada nije pokazao da ne postoji efikasan (brz) način da se riješi ovakav tip problema na uobičajenim računarima ali niko nije ni pokazao da postoji takav način. Većina računarskih naučnika ne vjeruje da će takav jedan efikasan metod ikada biti nađen. Naučićemo više o ovom tipu problema u sljedeće dvije aktivnosti.

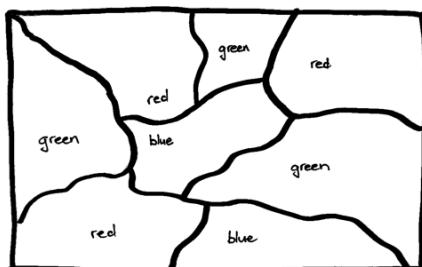
Dodatno čitanje

Harel je diskutovao problem četiri boje zajedno sa njegovom istorijom u časopisu *Algorithmics*. Više aspekata problema bojenja mapu je razmatrano u *This is MEGA-Mathematics!* od autora Casey i Fellows. Kubale-ova knjiga iz 2004, *Graph Colorings*, sadrži cijelu istoriju problema. Postoji veliki broj website-ova koji pokrivaju ovu temu.

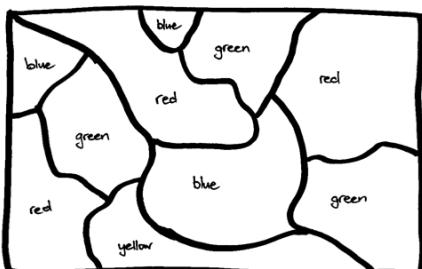
Rješenja i Pomoć

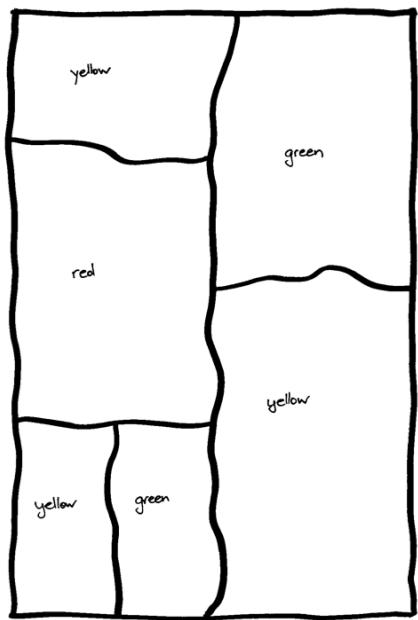


Ovo je jedino moguće rješenje bojenja mape sa radnog lista 1 (naravno, izbor samih boja može zavisiti od učenika ali ukupno samo dvije boje su potrebne).

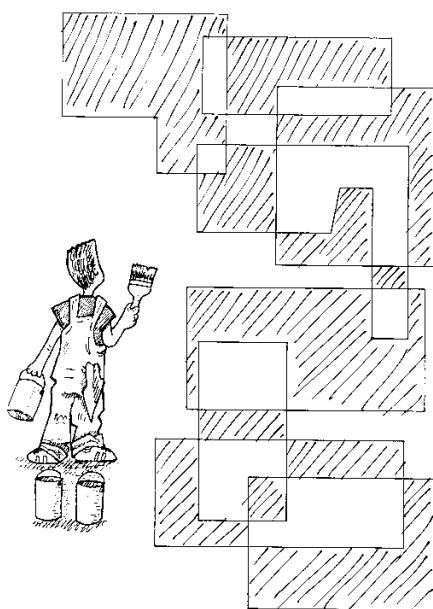


Mapa na vrhu radnog lista 2 može biti pravilno obojena koristeći tri boje dok za mapu na dnu trebaju četiri boje. Ovdje su predstavljena dva moguća rješenja.





Mapa sa radnog lista 3 je jedna jednostavnija mapa obojiva sa tri boje I jedno moguće rješenje je prikazano ovdje.



Ovo je jedno rješenje za radni list 4 koje koristi samo dvije boje (osjenčenu i bijelu).

Aktivnost 15

Turistički grad—Dominirajući skupovi

Sažetak

Mnoge situacije iz stvarnog života se mogu predstaviti i modelirati u formi mreže ili grafa kao što smo to uradili tokom aktivnosti bojenja mapa. Mreže pružaju mnogo mogućnosti i prilika za razvoj algoritama koji su korisni u praksi. U ovoj aktivnosti želimo označiti neke od raskrsnica, ili "čvorova", tako da su svi ostali čvorovi udaljeni najviše za jedan korak od nekog označenog čvora. Pitanje je koji je najmanji broj čvorova koje treba označiti? Ovaj problem, lagan za razumjevanje, je iznenađujuće težak za rješavanje.

Veze sa Curriculum-om

- ✓ Matematika – Postavaljanje and orjenacija
- ✓ Matematika – Logičko razmišljanje

Vještine

- ✓ Mape
- ✓ Odnosi i relacije
- ✓ Rješavanje mozgalica
- ✓ Iterativno traženje rješenja

Dobna/starosna grupa

- ✓ 7 godina i više

Materijali

Svaka grupa učenika će trebati:

- ✓ Jednu kopiju uzorka sa crnim linijama *Ice Cream Vans*, i
- ✓ Nekoliko brojača ili žetone za poker sa dvije različite boje.

Nastavniku će trebati

- ✓ Projektor kako bi se rješenje za uzorka sa crnim linijama *Ice Cream Vans* predstavilo na tabli, ili jednostavno tabla da se crta na njoj.



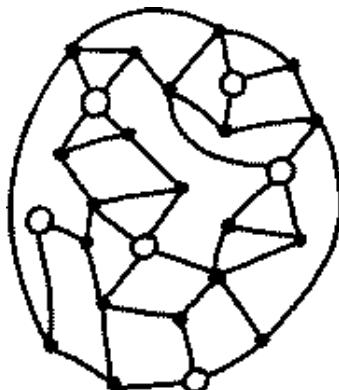
Dominirajući Skupovi

Uvod

Prodavači Sladoleda radni list predstavlja mapu Turističkog Grada. Linije predstavljaju ulice a tačke predstavljaju uglove na ulicama. Grad se nalazi u jednoj jako toploj zemlji I tokom ljeta prodavači sladoleda parkiraju svoja kolica na uglovima ulica i nude i prodaju sladoled turistima. Naša želja je postaviti kolica sa sladoledom tako da svako može doći do jednih kolica tako što će prošetati do kraja ulice i onda najviše jedan blok zgrada dalje. (Sve bi bilo lakše zamisliti ako bi ljudi živjeli na raskrsnicama ulica nego duž ulica; onda bi bili u mogućnosti doći do sladoleda hodajući najviše duž jednog bloka zgrada.) Pitanje je koji je najmanji broj kolica sa sladoledom i prodavačem koji trebamo i na kojim raskrsnicama ih treba postaviti.

Diskusija

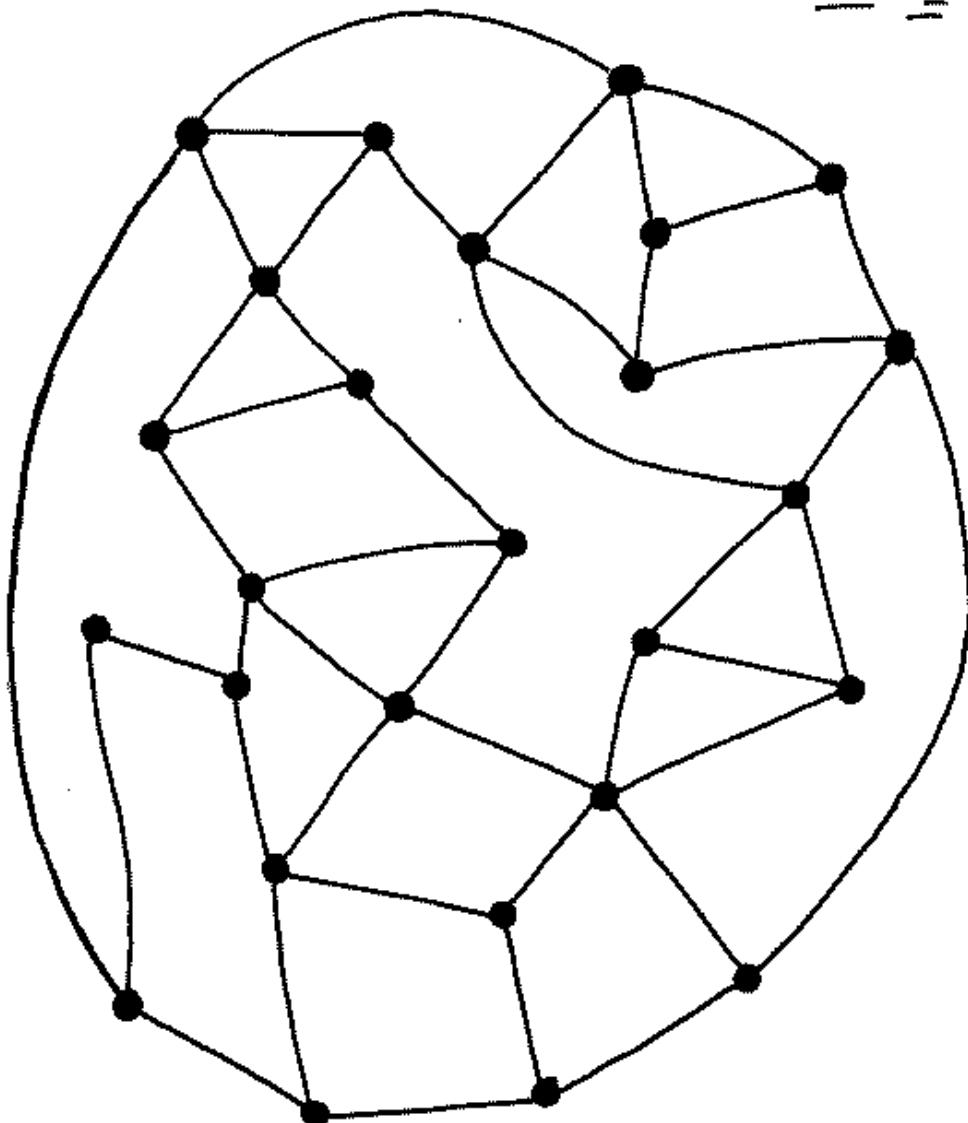
1. Podijelite učenike u male grupe i dajte svakoj grupi mapu Turističkog Grada kao I neke brojače, i ispričajte i objasnite cijelu priču.
2. Pokazati učenicima kako postaviti brojače na raskrsnicama kako bi označili jedna kolica sa sladoledom, a onda postavite brojače druge boje na raskrsnicama jednu ulicu dalje. Ljudi koji žive na ovim raskršćima (ili duž ulica koje dolaze do tog raskršća) će moći pronaći svoj sladoled na tim kolicima sa sladoledom.
3. Dajte učenicima da eksperimentišu sa različitim pozicijama kolica sa sladoledom. Kada pronađu jednu konfiguraciju koja uspješno zadovoljava sve kuće podsetite ih da su kolica sa sladoledom skupa i da je ideja da postavimo što je moguće manje kolica. Jasno je da je lagano ispuniti sve uslove problema ako imamo dovoljan broj kolica da ih postavimo na svaku raskrsnicu—interesantno je pitanje koliko ukupno kolica možemo uštedjeti.
4. Minimalan broj kolica za Turistički Grad je šest i odgovarajuće rješenje je prikazano ovdje sa strane. Ali s druge strane nije nimalo lagano naći ovo rješenje! Nakon nekog vremena saopštite razredu da je šest kolica sa sladoledom dovoljno za jedno rješenje i izazovite ih da nađe mjesta gdje ih treba postaviti. I ovo ostaje prilično težak problem: kolik grupa će ipak na kraju odustati. Čak i rješenja sa osam ili devet kolica mogu biti dosta teška za nalaženje.



5. Mapa Turističkog Grada je konstruisana tako što se krenulo sa postavljenih šest kolica kao što je prikazano na dnu radnog lista Rješenje Kolica sa Sladoledom I onda spajanjem tih mesta sa mnogim novim ulicama kako bi se pravo rješenja problema dobro sakrilo. Najvažnije je paziti da ne postavite novu ulicu između praznih kružića gdje se nalaze kolica nego samo između dodatnih crnih kružića. Pokažite ovu tehniku kreiranja problema cijelom razredu koristeći pripremljen projektor.
6. Tražite od učenika da naprave svoje mape koje su teške za rješavanje koristeći prezentiranu tehniku. Oni onda mogu izazvati svoje prijatelje ili roditelje da riješe problem–razumjeće na kraju da su u stanju napraviti mozgalicu (problem) koji oni rješavaju sa lakoćom dok ih drugi ne mogu rješiti! Ovo su sve primjeri koji opisuju funkciju u jednom smjeru (eng. “one-way function”): lagano je napraviti mozgalicu koja je jako teška za rješavanja—samo ukoliko niste vi onaj koji ju je kreirao. Ove funkcije jednog smjera su od krucijalne važnosti u kriptografiji (pogledati Aktivnosti 17 i 18).

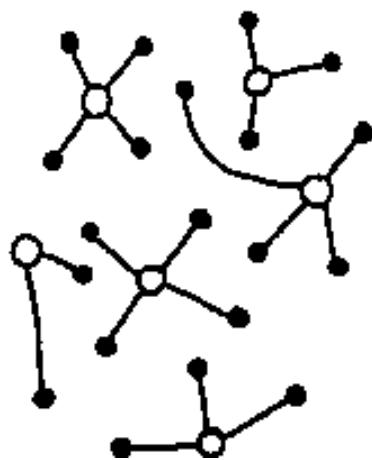
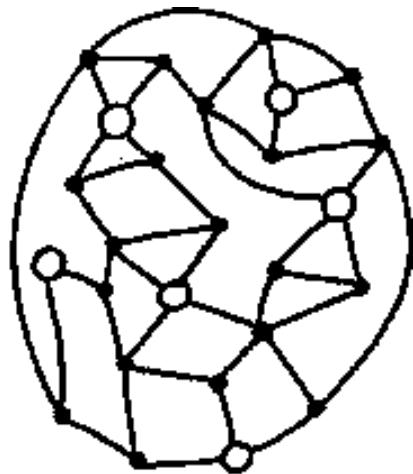
Radni List za Aktivnost: Kolica sa Sladoledom

Pronađite način kako rasporediti kolica sa sladoledom na raskrsnicama ulica tako da je svaka raskrsnica povezana sa jednom koja ima parkirana kolica sa sladoledom.



Radni List za Aktivnost: Rješenje za Kolica sa Sladoledom

Pokažite cijelom razredu ove dvije šeme kako bi svi vidjeli kako je problem napravljen.



Varijacije i proširenja

Ima mnogo situacija u kojima neko može biti suočen sa sličnim problemom prilikom planiranja grada: postavljanje poštanskih sandučića, fontana sa vodom za piće, vatrogasnih stanica, i tako dalje. Ali u stvarnom životu I sa stvarnim problemima mapa neće biti napravljena pomoću trika koji osigurava lagano nalaženje rješenja. Ukoliko stvarno trebate riješiti jedan problem poput ovog kako bi postupili?

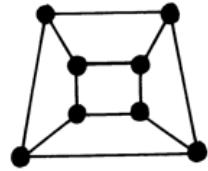
Ima jedan prilično jasan i neposredan način: posmatrajmo sve moguće načine postavljanja kolica sa sladoledom i provjerimo koja od njih je najbolja. Sa ukupno 26 uglova ulica u Turističkom Gradu postoji 26 načina da postavimo jedna kolica sa sladoledom. Lagano je provjeriti svih 26 mogućnosti kao što je jasno da nijedna od njih ne zadovoljava tražene uslove. Ako imamo dvoja kolica sa sladoledom na raspolaganju, onda postoji 26 mesta da postavimo prva kolica, i onda, koje god mjesto je izabранo za prva kolica preostaje 25 mesta za druga kolica (naravno da ne želite postaviti dvoja kolica sa sladoledom na isto mjesto): $26 \times 25 = 650$ mogućnosti koje treba provjeriti. Ponovo, svaka od ovih provjera je lagana ali može biti naporno (za čovjeka) da ih sve uradi. U stvari, potrebno je provjeriti samo pola od njih (325) jer nije bitno koja kolica su koja: ako provjerite kolica broj 1 na raskršću A i kolica broj 2 na raskršću B onda nije potrebno provjeravati kolica 1 na B i kolica 2 na A. Sada treba nastaviti provjere sa troja kolica sa sladoledom (2600 mogućnosti), četvera kolica (14950 mogućnosti), i tako dalje. Jasno, 26 kolica će biti dovoljno jer imate samo 26 raskrsnica i nema nikakvog razloga da imate više od jednih kolica na jednoj te istom mjestu. Drugi način da nađete broj mogućnosti je da posmatramo koji je ukupan broj konfiguracija (mogućnosti rješenja) kada imamo 26 raskrsnica i bilo koji broj kolica. Kako imamo tačno dvije mogućnosti za svaki ugao ulice—kolica sa sladoledom su tu ili nisu—broj konfiguracija je 2^{26} , što je 67,108,864.

Ovaj način rješavanja problema se naziva algoritam grube sile ili čiste sile (eng. “brute-force”) ili enumeracija i može zahtjevati zaista veliko kompjutaciono vrijeme. Uvriježeno je (**pogrešno**) mišljenje da su računari tako brzi da mogu riješiti bilo koji problem brzo bez obzira koliko je rada potrebno za to rješavanje. Ali to naravno nije tačno. Koliko vremena je potrebno jednom algoritmu čiste sile da riješi problem zavisi i od vremena koje mu je potrebno da provjeri da li je neka konfiguracija zaista rješenje problema. Kako bi to provjerili u našem problem potrebno je provjeriti za svaku raskrnicu najmanju udaljenost do nekih kolica sa sladoledom. Prepostavimo da je moguće svaku konfiguraciju provjeriti za jednu sekundu. Koliko je onda vremena potrebno za sve konfiguracije Turističkog Grada? (Odgovor: 2^{26} je oko 67 miliona; imamo 86,400 sekundi u jednom danu, pa tako znamo da je 2^{26} sekundi nešto oko 777 dana, ili približno dvije godine.) Sada prepostavimo da je umjesto jedne sekunde dovoljan samo hiljaditi dio sekunde da bi se provjerila jedna konfiguracija. Onda bi za spomenute dvije godine koristeći novi računar mogli riješiti samo neki 36-raskrsnica grad, jer je 2^{36} oko 1000 puta više od 2^{26} . Čak i kad bi neki računar bio i million puta brži, tako da je moguće provjeriti million konfiguracija u samo jednoj sekundi, dvije godine bi bile dovoljne da se riješi problem u gradu sa samo 46 raskrsnica. Ovo sigurno nisu neki veliki gradovi! (Koliko raskrnicu imate u vašem gradu?)

Kako vidimo da je algoritam čiste sile prespor onda treba pitati da li postoje drugi načina da se riješi isti problem? Pa recimo mogli bi probati neki pristup pohlepnim algoritmom koji je bio uspješan za rješavanje problema blatnjavog grada (Aktivnost 9). Prvo bi trebalo razmislići šta znači biti pohlepan sa kolicima sa sladoledom—drugim rječima kako primjeniti pohlepni pristup na problem kolica sa sladoledom. Jedan način da se to uradi je da postavimo prva kolica sa sladoledom na raskrsnicu u kojoj se spaja najviše ulica, pa onda druga kolica na sljedeću raskrsnicu sa najviše ulica, i tako dalje. Ipak, ovo

neće uvijek (u stvari, rijetko) proizvesti minimalan skup kolica sa sladoledom koje je rješenje problema—u stvari, raskrsnica sa najviše povezanih ulica u Turističkom Gradu, koja ima pet ulica, nije dobro mjesto za postavljanje kolica sa sladoledom (provjerite ovu činjenicu sa cijelim razredom).

Pogledajmo sada jedan lakši problem. Umjesto da se bavimo pitanjem nalaženja najmanje konfiguracije, pretpostavimo da nam je data jedna konfiguracija i da se od nas traži da odredimo da li je ona minimalna ili nije. Na primjer, na dijagramu ovdje sa strane prikazana je mnogo jednostavnija mapa čije rješenje je prilično jasno. Ako zamislimo ulice kao bridove jedne kocke onda je jasno da su dvoja kolica sa sladoledom na suprotnim vrhovima kocke dovoljna za rješenje. Pored toga, potrebno je da pokažete da nije moguće riješiti ovaj problem sa manje od dvoja kolica sa sladoledom. Mnogo teži zadatak je—iako nije nemoguć—da pokažete da Turistički Grad nije moguće dobro opslužiti sladoledom sa manje od šest kolica sa sladoledom. U opštem slučaju jako teško je pokazati da je neka data konfiguracija minimalna za datu mapu.



Zašto je ovo sve važno?

Jedna interesantna činjenica vezana za problem sa kolicima sa sladoledom je da *niko* ne zna da li postoji neki algoritam za nalaženje minimalnog skupa lokacija koji bi bio značajno brži od algoritma čistom silom! Kompjutaciono vrijeme koje treba algoritmu čistom silom raste eksponencijalno sa brojem raskrsnica—takav algoritam se naziva eksponencijalan algoritam (eng. *exponential-time algorithm*). S druge strane, polinomijalna algoritam (eng. *polynomial-time algorithm*) je onaj čije vrijeme izvršenja raste sa kvadratom, ili kubom, ili sa sedamnaestim stepenom, ili bilo kojim drugim konstantnim stepenom broja raskrsnica. Polinomijalan algoritam će uvijek biti brži za dovoljno velike mape—čak i onaj (recimo) algoritam koji je sedamnaestog stepena—jer jedna funkcija eksponencijalnog rasta uvijek nadmaši bilo koju polinomijalnu funkciju za neki dovoljno veliki argument. (Na primjer, može se pokazati da je za svaki n veći od 117 vrijednost funkcije n^{17} manja od vrijednosti funkcije 2^n). Da li postoji polinomijalan algoritam za nalaženje minimalnog skupa lokacija?—to niko ne zna iako je mnogo istraživača naporno radilo da nađe jedan takav algoritam. Pored toga, isto vrijedi i za naizgled laksi problem provjere da li je određeni dati skup lokacija minimalan: algoritam čiste sile koji probava sve mogućnosti koje imaju manje lokacija je takođe eksponencijalan u odnosu na broj raskrsnica, a polinomijalan algoritam nije nikada bio pronađen niti je pokazano da takav algoritam ne postoji.

Da li vas ova situacija podsjeća na bojenje mapa (Aktivnost 13)? Trebalo bi. Problem postavljanja kolica sa sladoledom koji je poznat i kao problem “najmanjeg dominirajućeg skupa” (eng. “minimum dominating set”) je samo jedan od velikog broja problema—na hiljadu—za koje ne znamo da li postoji polinomijalan algoritam. Takvi problemi se pojavljuju u oblastima kao što su logika, slagalice pa bojenje mapa, nalaženje optimalnih ruta u datoj mreži, raspoređivanju zadataka (eng. *scheduling*). Začuđujuće je da je za sve ove probleme pokazano da ako bi postojao jedan polinomijalan algoritam za neki od ovih problema onda bi se on mogao pretvoriti ili prilagoditi za sve ostale—pa prema tome se kaže da su ili svi rješivi ili nijedan nije rješiv.

Ovakvi problem se nazivaju *NP-kompletni* (eng. *NP-complete*). NP označava “nedeterministički polinomijalan” (eng. “non-deterministic polynomial”). U ovakvoj terminologiji to znači da taj problem može biti rješen u razumnom vremenu ukoliko bi vam bio na raspolaganju računar koji može isprobati proizvoljno veliki broj rješenja u isto vrijeme (to je onaj dio o nedeterminističkom u imenu). Možete misliti da je to prilično nerealna pretpostavka, kao što u stvari i jeste. Nije moguće konstruisati u stvarnosti takav računar jer bi onda on morao biti proizvoljno velik (fizički). S druge strane, koncept postojanja takve maštine je važan u principu jer se pokazuje da NP-kompletni problemi nisu rješivi u razumnom vremenu ako nisu dostupni takvi nedeterministički računari.

Pored toga, ova grupa problema se naziva *kompletni* (eng. *complete*) jer iako sus vi ovi problemi na prvi pogled jako različiti—na primjer, problem bojenja mapa je jako različit od problema postavljanja kolica sa sladoledom—pokazuje se da ako bi se našao jedan efikasan način za rješenje jednog takvog problema onda bi se taj metod mogao prilagoditi da riješi i bilo koji drugi problem u ovoj grupi problema. To je ono na šta smo mislili kad kažemo oni su svi “ili rješivi ili nerješivi” (eng. “standing or falling together.”)

Postoji na hiljade NP-kompletnih problema, i različiti istraživači su naporno pokušavali riješiti svaki od njih desetinama godina ali bez uspjeha. Ukoliko bi se našlo efikasno

dobro rješenje za samo jedan od njih to bi značilo da bi imali efikasno rješenje za sve njih. Upravo iz tog razloga postoji velika sumnja da postoji jedno takvo efikasno rješenje. U isto vrijeme, dokazati da se ovi problemi mogu riješiti samo nekim eksponencijalnim algoritmom je jedno od najpoznatijih otvorenih pitanja računarskih nauka—a vjerovatno I cijele matematike—danас.

Dalje čitanje i literatura

Harel-ova knjiga *Algorithmics* predstavlja nekoliko različitih NP-kompletnih problema I razmatra pitanje postojanja polinomijalnih algoritama za njih. Dewdney-eva knjiga *Turing Omnibus* takođe razmatra NP-kompletnost. Najpoznatija i najstandardnija knjiga računarskih nauka na ovu temu je knjiga autora Garey & Johnson pod nazivom *Computers and Intractability*, koja predstavlja nekoliko stotina NP-kompletnih problema zajedno sa načinima za dokazivanje njihove NP-kompletnosti. Ipak, ova knjiga je možda dosta teška za čitanje i pogodna je samo za specijaliste u računarskim naukama.

Activity 16

Ledeni putevi —*Steiner stabla*

Sažetak

Ponekad je moguće da mala, skoro zanemariva promjena u specifikacije samog problema može rezultirati velikom razlikom u tome koliko je problem težak za rješavanje. Ova aktivnost, kao i problem Blatnjavi Grad (Aktivnost 9) je o nalaženju najkraćih puteva u jednoj mreži. Razlika je da je ovdje moguće dodati nove tačke na mreži ukoliko će to smanjiti dužinu puta. Kao rezultat dobivamo jedan puno teži problem koji nema nikakve veze sa problemom Blatnjavog Grada ali je algoritamski ekvivalentan (potpuno jednak) problemima kartografa (Aktivnost 13) I problemu Turističkog Grada (Aktivnost 14).

Veze sa Curriculum-om

- ✓ Matematika – Postavaljanje and orjenacija
- ✓ Matematika – Logičko razmišljanje

Vještine

- ✓ Prostorna vizualizacija
- ✓ Geometrijsko razmišljanje
- ✓ Algoritamske procedure i kompleksnost

Dobna/starosna grupa

- ✓ 7 godina i više

Materijali

Svaka grupa učenika će trebati

- ✓ pet ili šest štipalji ili kuka koje će postaviti na tlo (kuke za šator su dosta dobre, ali malo jača žica isječene na komade i savijena u obliku kuke može dobro poslužiti),
- ✓ nekoliko metera užeta ili gume,
- ✓ linijar ili savitljivi metar, i
- ✓ olovka i papir kako bi se mogle uzeti podaci.

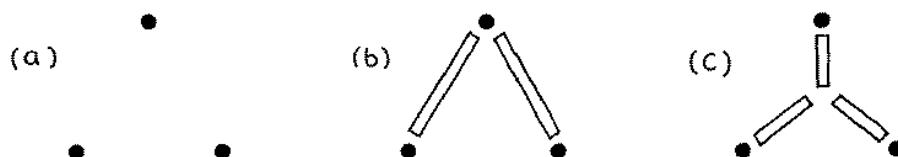
Ledeni putevi



Uvod

Prethodna aktivnost, Turistički Grad, se cijela događala u jednoj jako toploj zemlji; u ovom slučaju je sasvim suprotno. Na smrznutom sjeveru Kanade (tako nekako priča kaže), tokom zime na velikim zaleđenim jezerima, grtalice snijega imaju zadatku da naprave puteve kako bi spojili crpne stanice i kako bi se posade mogle uzajamno posjećivati. Tamo gore u zimi oni žele da naprave što je moguće manje puteva i vaš zadatku je da im pomognete pri izboru kuda i kako da naprave puteve. Ne postoje posebna ograničenja: putevi se mogu pružiti bilo gdje u snijegu—cijela jezera su zaleđena i pokrivena snijegom. Sve je savršeno ravno.

Putevi naravno treba da budu prave linije jer bi svako krivudanje povećalo ukupnu dužinu puta bez potrebe. Ali zadatku nije baš jednostavno spajanje svih crpnih stanica pravim linijama jer bi dodavanjem raskrsnica gore u bijelim bespućima Kanade moglo smanjiti ukupnu dužinu puteva—i jedino što je važno je ukupna dužina puteva a ne vrijeme putovanja od jednog mjesta do drugog.

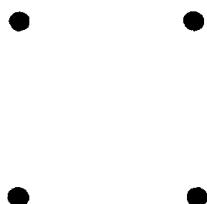


Na ovoj slici ovdje gore su predstavljene, (a) tri različite crpne stanice. Povezivanjem jedne sa ostale dvije (kao što je to u (b)) bi kreiralo jednu prihvatljivu mrežu puteva. Druga mogućnost je da napravimo raskrsnicu negdje oko centra zamišljenog trougla I da onda povežemo sve tri crpne stanice sa centrom (c). I sada ako bi izmjerili ukupnu dužinu puteva koji se trebaju očistiti vidjeli bi da je to jedno bolje rješenje. Ova dodatna raskrsnica se naziva "Steiner" (izgovara se Štajner) tačka po švicarskom matematičaru Jacob Steiner (1796–1863), koji je formulisao problem I bio prvi koji je primjetio da se ukupna dužina puteva može smanjiti dodavanjem novih tačaka. Moguće je misliti o Steiner tački kao o novoj, zamišljenoj (virtualnoj) crpnoj stanici.

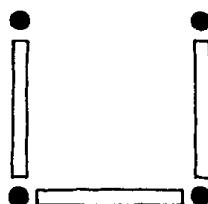
Diskusija

- Opišite problem učenicima i objasnice koje rješenje treba tražiti. Koristeći prethodni primjer pokažite da je sa tri crpne stanice dodavajući jednu novu tačku ponekad moguće poboljšati rješenje smanjujući ukupnu dužinu puteva koje treba očistiti.

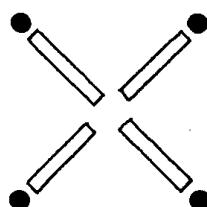
(a)



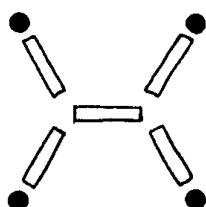
(b)



(c)

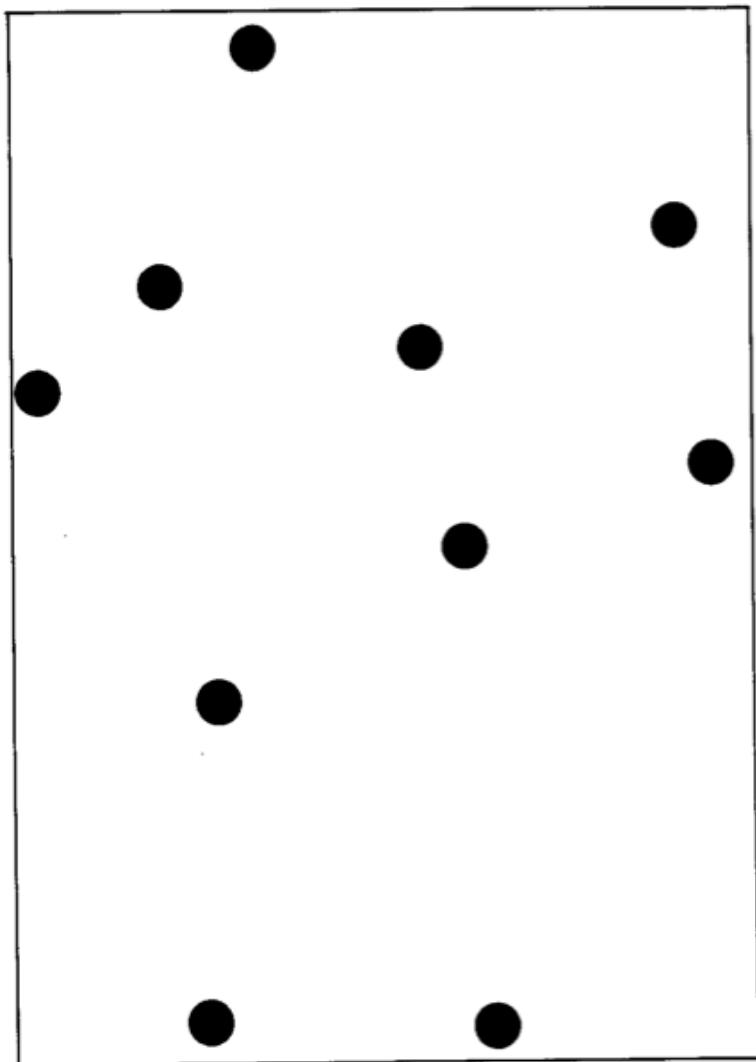


(d)



- Učenici će, za vježbu, koristiti četiri tačke raspoređene u vrhove kvadrata kao što je prikazano na slici (a). Izađite sa učenicima vani i tražite od svake grupe učenika da postave četiri kuke u travu u vrhove kvadrata dimenzija 1 metar sa 1 metar.
- Neka sada učenici počnu eksperimentisati povezujući kuke užetom ili gumom i mijereći i zapisujući minimalnu ukupnu dužinu neophodnih puteva. U ovoj fazi učenici ne bi trebali koristiti nijednu Steiner tačku. (Minimum se može postići povezujući tačke duž tri stranice kvadrata kao što je prikazano u (b), pa je ukupna dužina potrebnih puteva jednaka 3 metra.)
- Sada provjerimo da li učenici mogu poboljšati rješenje koristeći jednu Steiner tačku. (Najbolje mjesto je u centru kvadrata, (c). Ukupna dužina je sada $2\sqrt{2} = 2.83$ metra.) Predložite im da je moguće uraditi još i bolje ako bi koristili dvije Steiner tačke. (I zaista mogu postići bolju konfiguraciju ako postave dvije tačke kao u (d), formirajući uglove od 120 stepeni između puteva. Ukupna dužina je sada $1 + \sqrt{3} = 2.73$ metra.)
- Da li učenici mogu naći još i bolje rješenje ako bi koristili tri Steiner tačke? (Ne – dvije tačke su najbolje, i nije moguće poboljšati to rješenje dodajući još jednu ili više Steiner tačaka.)
- Diskutujte sa učenicima zašto ovi problemi izgledaju teški.(To je upravo zbog toga jer ne znate gdje treba postaviti Steiner tačke, a postoji jako puno različitih mogućnosti za to.)

Radni List za Aktivnost: Steiner Stablo Primjer 1



Radni List za Aktivnost: Steiner Stablo Primjer 2

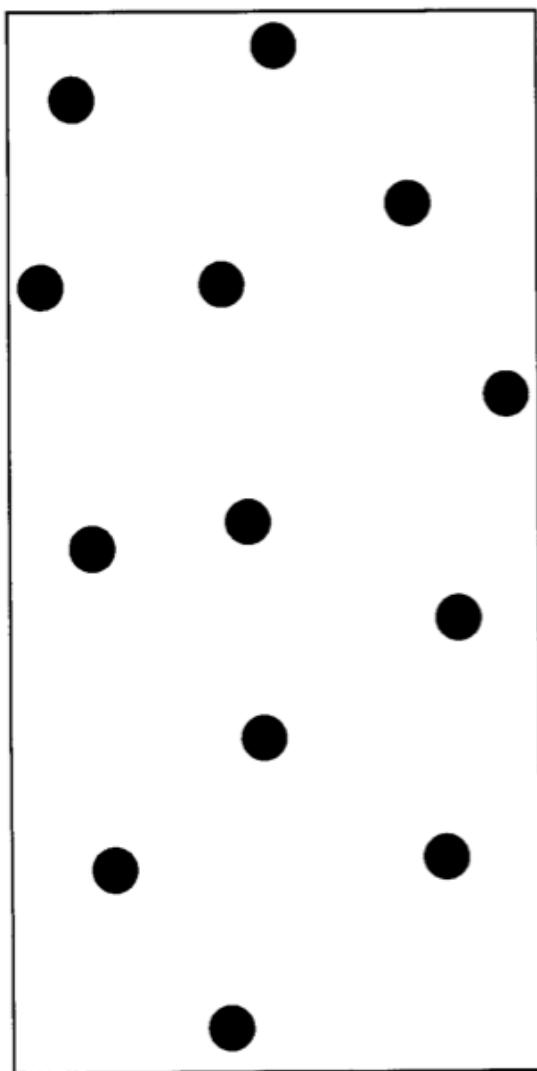


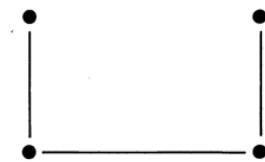
Diagram 132

Varijacije i proširenja

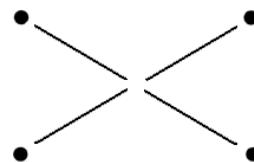
(a)



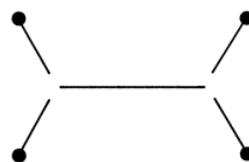
(b)



(c)

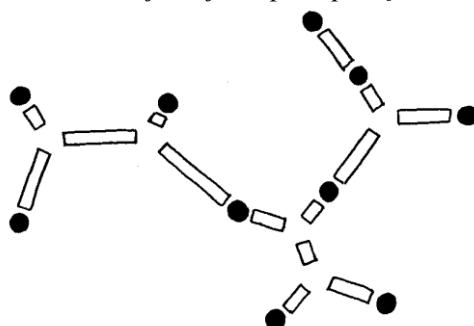


(d)

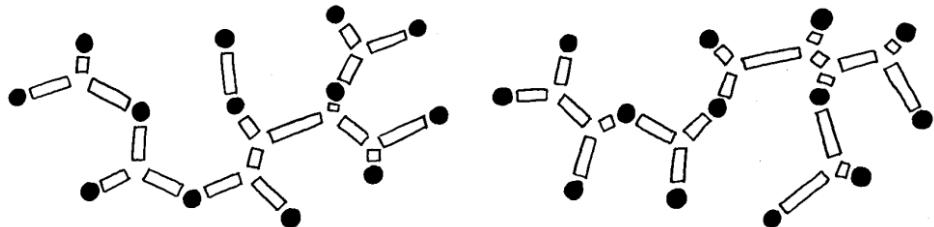


1. Jedan zanimljiv eksperiment, za one grupe koje su sve svoje eksperimente završile ranije, je da posmatramo pravougaonik dimenzija otprilike 1 metar sa 2 metra (a). Učenici će naći da bi dodavanje jedne Steiner tačke stvari učinilo gorim ali da bi dodavanje dvije tačke poboljšalo rješenje. (Ukupna dužina je 4 metra u (b), $2\sqrt{5} = 4.47$ metra za (c), i $2 + \sqrt{3} = 3.73$ metra za (d).) Potaknite ih da razmišljaju zašto konfiguracija sa jednom dodanom tačkom daje gori rezultat nego u slučaju pravougaonika za razliku od problema sa kvadratom. (To je zbog toga jer kada kvadrat rastegnemo u pravougaonik onda ovo rastezanje dodaje dužinu samo jednom putu u slučaju (b) i (d), dok obje dijagonale dobivaju na dužini u slučaju povećanja u (c).)
2. Stariji učenici mogu rješavati veće probleme. Dva rasporeda crpnih stanica koje treba povezati putevima su data u radnim listovima. Učenici mogu eksperimentisati sa različitim rješenjima bilo da koriste nove kopije radnih listova ili da iznova sebi zadaju isti problem koristeći neku transparentnost papira postavljenog iznad prvobitne kopije (i reproducirajući problem tako). Alternativa je takođe da naprave maketu crpnih stanica na tlu koristeći pripremljene kuke. Grupa koja misli da je postigla novi najbolji rezultat može to objaviti na sav glas cijelom razredu. (Figura koje se nalaze sa desne strane pokazuju koja konfiguracija postiže minimalno rješenje za prvi primjer a za drugi primjer imamo pokazana dva moguća najbolja rješenja ovje dolje čija je ukupna dužina jednak ili približno jednak.) Činjenica da imamo dva različita rješenja slične kvaliteti ilustruje zašto je ovaj tip problema tako težak—ima tako puno različitih izbora gdje postaviti nove Steiner tačke!

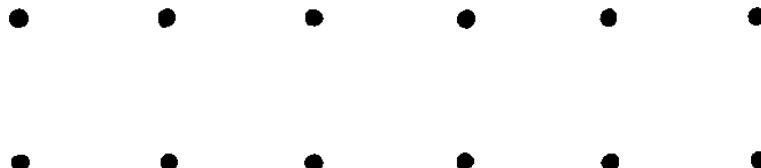
Minimalno rješenje za prvi primjer



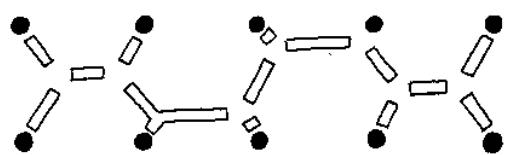
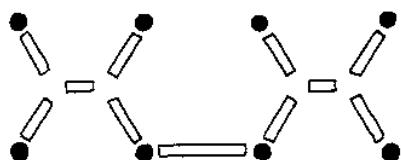
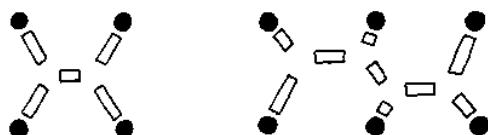
Dva moguća Steiner stabla za drugi primjer



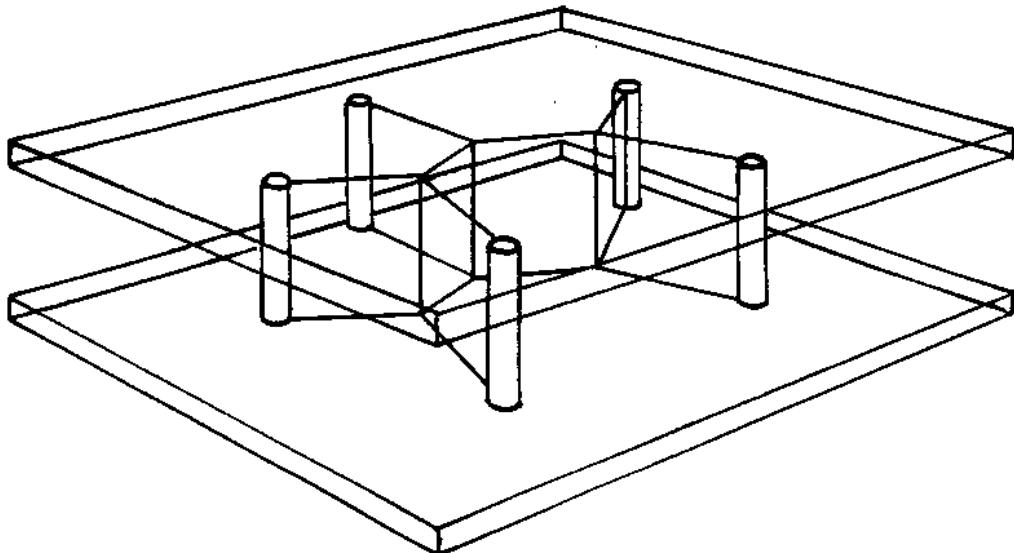
3. Stepeničaste mreže (eng. ladder network) kao ova predstavljena ovdje nude jedan drugi način na koji se problem može proširiti. Stepeničasta mreža izgleda ovako:



Neka minimalna Steiner stabla za stepeničaste mreže su pokazana ovdje dolje. Rješenje za stepenice sa samo dvije prečke je isto kao i za kvadrat. Ipak, u slučaju stepenica sa tri prečke rješenje je prilčno novo i različito—kao što ćete I sami primjetiti ako ga pokušate reproducirati samo po sjećanju! Rješenje za stepenice sa četiri prečke je kao i ono za dvoje stepenica sa po dvije prečke spojena zajedno dok rješenje za stepenice sa pet prečki izgleda kao jedno poopštenje rješenja za stepenice sa tri prečke. U opštem slučaju, može se reći da rješenje za stepeničastu mrežu zavisi da li stepenice imaju paran ili nepar broj prečki. Ako je to paran broj onda je rješenja kao da smo spojili nekoliko rješenja za stepenice sa dvije prečke. U drugom slučaju, kada je broj prečki neparan, onda je to kao ponavljanje rješenja za stepenice sa tri prečke. Treba biti oprezan jer strogi i tačan dokaz ovih tvrdnji nije ni malo trivijalan niti lagan.



4. Još jedna interesantna aktivnost je konstruisanje Steiner stable pomoću pjene od sapunice. Ovo možete uraditi tako što ćete uzeti dvije ploče od providne nesavitljive plastike i postaviti zabadače koji će predstavljati crpne stanice koje treba povezati, upravo kako je prikazano ovdje dolje.



Sada potopite cijelu konstrukciju u pjenu od sapunice. Nakon što izvučete cijelu konstrukciju vidjećete da je film od pjene povezao zabadače u prelijepu mrežu koja je, u stvari, Steiner stablo.

Na žalost, to ipak nije uvijek minimalno Steiner-ovo stablo. Pjenasti film zaista nalazi konfiguraciju koja minimizira ukupnu dužinu ali taj postignuti minimum je samo lokalni a nije uvijek u globalan. Moguće je da postoji jedan sasvim drugim način postavljanja novih Steiner tačaka koje bi dale manju ukupnu dužinu. Na primjer, moguće je zamisliti film od sapunice koje izgleda kao i konfiguracija u Ekstenziji 2 kada je jednom izvadimo iz otopine i jedna sasvim drugačija konfiguracija kada je izvadimo iz otopine drugi put.

Zašto je ovo sve važno?

Mreže na kojima smo radili u ovoj aktivnosti se nazivaju minimalna Steiner-ova stable. Nazivaju se "stabla" jer nemaju kontura upravo kao što i grane na nekom pravom stablu rastu u stranu jedna od druge i, uobičajeno, se ne sastaju ponovo da bi onda ponovo rasle zajedno. Nazivamo ih "Steiner" stabla zbog ovih novih tačaka, Steiner tačaka, koje mogu biti dodane početnim, originalnim stanicama koje ćemo onda sve zajedno povezati. I na kraju se nazivaju "minimalna" jer imaju najmanju ukupnu dužinu među svim stablima koja povezuju sve početne stanice. U problemu Blatnjavi Grad (Aktivnost 9) smo naučili da je mreža koja povezuje dati skup tačaka čija je ukupna dužina minimalna u stvari minimalno pokrivajuće stablo: Steiner-ova stable su takođe minimalna pokrivajuća stable samo što je moguće dodati nove tačke.

Interesantna je činjenica da dok postoji jako efikasan algoritam za nalaženje minimalnog pokrivajućeg drveta (Aktivnost 14)—pohlepni algoritam koji iterativno povezuje dvije najbliže do tada nepovezane tačke—ne postoji, u opštem slučaju, efikasno rješenje, algoritam, za problem minimalnog Steiner-ovog stabla. Steiner stablo je puno teži problem jer morate odlučiti gdje postaviti dodatne tačke. U stvari, i to je prilično iznenadjuće, teži dio rješenja problema minimalnog Steiner-ovog stable nije odrediti koje su tačno pozicije za nove Steiner tačke nego odlučiti otprilike gdje bi se one mogle nalaziti: ova razlika se, na primjer, dobro vidi u dva različita rješenja za Primjer 2. Jednom kada znate mjesta gdje treba postaviti nove tačke precizno određivanje gdje te tačke treba postaviti je relativno jednostavan zadatak. Film od sapunice može to uraditi na vrlo efikasan način pa tako i sami računari to mogu dobro riješiti.

Nalaženje minimalnih Steiner stabala je i dio priče koja može donijeti I velike uštede u novcu kada je riječ o telefonskim kompanijama. Prije 1967, kada su velike korporacije u Sjedinjenim Državama održavale same svoje velike korporativne private telefonske linije u isto vrijeme su i iznajmljivale te linije telefonskim kompanijama. Račune koje su ispostavljali tim kompanijama nije bila zasnovana na tome na koji način su žice bile stvarno korištene već na osnovu najmanje mreže koja bi zadovoljavala sve potrebe. Razmišljanje je bilo da korisnik ne bi trebao plaćati ništa dodatno ako bi telefonska kompanija koristila neki zaobilazan put. Na samom početku, algoritam koji je računao koju sumu treba fakturisati je radio na osnovu određivanja minimalnog pokrivajućeg stabla. Ipak, negdje oko 1967 godine primjećeno je od strane jednog korisnika—jedne avio kompanije koja je, u stvari, imala tri velika čvorišta—da ako bi oni zatražili i četvrto čvorište kao prelaznu tačku onda bi se ukupna dužina mreže smanjila. Telefonska kompanija je tako bila obavezna da reducira cijene na nivo koji odgovara nivou koji bi imali ukoliko bi postojala telefonska razmjena i u jednoj Steiner tački! Iako je tačno da je minimalno Steiner-ovo drvo samo 5% ili možda 10% manje po dužini od minimalnog pokrivajućeg stabla uštede postaju značajne kada su u pitanju velike količine novca u pitanju. Problem Steiner-ovog stable je ponekad nazivan i "problem najkraće mreže" (eng. "shortest network problem") jer uključuje nalaženje najkraće mreže koja povezuje skup stanica ili lokacija.

Ukoliko ste učestvovali u rješavanju problema iz obje prethodne aktivnosti, kartografova mozgalica i problem turističkog grada, onda vas neće iznenaditi da čujete da je problem Steiner-ovog stable NP-kompletan problem. Kako se ukupan broj lokacija povećava tako i broj mogućih pozicija za Steiner-ove tačke se povećava i isprobavanje svih mogućnosti podrazumjeva i neku vrstu pretraživanja koje ima eksponencijalan rast. To je još jedan od hiljada problema za koje jednostavno nije poznato, za sada, da li je eksponencijalno pretraživanje ono najbolje što možemo uraditi ili pak postoji još neotkriven

polinomijalan algoritam. Ipak ono što je poznato je da ako bi postojao takav polinomijalan algoritam za ovaj problem onda bi ga mogli prilagoditi i izmjeniti tako da imamo i polinomijalan algoritam i za bojenje grafova, i za problem dominirajućeg skupa—kao i za sve ostale problem za koje znamo da se nalaze u klasi NP-kompletih problema.

Već smo objasnili na kraju prethodne aktivnosti da “NP” u NP-kompletan označava “nedeterministički polinomijalan,” a da se “kompletan” odnosi na činjenicu se jedan polinomijalan algoritam za jedan NP-kompletan problem može pretvoriti redom u polinomijalne algoritme za sve ostale problem. Skup problema koji su rješivi u polinomijalnom vremenu se nazivaju P. Prema tome krucijalno pitanje, da li postoji polinomijalan algoritma za NP-kompletne probleme, se drugim riječima svodi na pitanje—da li vrijedi da je $P = NP$? Odgovor na ovo pitanje nije poznat i to je jedna od velikih misterija računarskih nauka danas.

Problemi za koje postoje polinomijalni algoritmi—čak i kada su ti algoritmi prilično spori—se nazivaju “rješivi” (eng. “tractable”) problemi. Za probleme za koje ne poznajemo takav algoritam se nazivaju “nerješivi” (eng. “intractable”) jer bez obzira sa kako brzim računarom raspolažete, ili pak koliko imate ukupno računara na raspolaganju, malo povećanje u veličini problema znači da neće biti moguće u praksi riješiti takav problem u razumnom vremenu, pa dakle neće biti rješiv u praksi. Nije poznato da li su NP-kompletni problemi—koji uključuju i kartografsku mozgalicu kao i problem turističkog grada ili problem ledenih puteva—rješivi ili nisu. Ipak treba imati na umu da je velika većina računarskih naučnika pesimistična u pogledu postojanja polinomijalnog algoritma za NP-kompletne problem, dakle ne vjeruju da će se takav algoritam ikad naći, pa onda jedan dokaz da je problem NP-kompletan nudi jedan jak argument da je praktično taj problem u suštini nerješiv.

Šta možete uraditi kada vaš šef traži od vas da kreirate jedan efikasan algoritam koji će pronaći optimalno rješenje za dati problem, i vi ipak niste u stanju pronaći takav algoritam?—ono što se sigurno događa kada avionska kompanija prepozna da se troškovi korištenja jedne mreže mogu smanjiti ako bi imali pravo uvođenja Steiner tačaka. Ukoliko ste u stanju dokazati da ne postoji efikasan algoritam koji može proizvesti optimalno rješenje za dati problem onda je to zaista sjajno. S druge strane, treba imati na umu da je izuzetno teško dokazati ovakve negativne rezultate u računarskim naukama, jer nikada ne znate koji od onih sjajnih programera može doći u nekoj budućnosti i iskoristiti neki zaista rijedak trik koji omogućava rješavanje tog problema. Prema tome, na žalost, vrlo malo je moguće da se nađete u poziciji da možete kategorično tvrditi da ne postoji nikakav efikasan algoritam za dati problem—drugim riječima da je problem potpuno nerješiv. Ipak ako možete pokazati da je vaš problem NP-kompletan onda to znači da su hiljade ljudi vrijedno radili u istraživačkim laboratorijama na različitim problemima koji su svi ekvivalentni vašem problem, i da ni oni nisu mogli naći nikakvno efikasno rješenje. Ovakav dokaz vam možda neće donijeti neki dodatni bonus u kompaniji u kojoj radite ali ćete se spasiti većih problema sa vašim šefom!



“Ja ne mogu naći neki efikasan algoritam za ovaj problem. Prepostavljam da sam prilično glup.”

“Ja ne mogu naći neki efikasan algoritam za ovaj problem. Takav algoritam ne postoji.”

“Ja ne mogu naći neki efikasan algoritam za ovaj problem ali niko ni od ovih poznatih naučnika nije našao takvo rješenje.”

Šta uraditi kada niste u mogućnosti, to jest niste znali, naći efikasan algoritam za vaš problem: tri mogućnosti

Naravno da je u stvarnom životu potrebno na neki način riješiti ove probleme pa su onda ljudi odlučili da pribjegnu heuristikama – to jest algoritmima koji ne mogu garantovati najbolje moguće rješenje ali mogu naći neko rješenje koje je za mali procenat udaljeno od optimalnog. Heuristički algoritmi mogu biti jako brzi, a takođe šteta uzrokovana nenalaženjem najboljeg rješenja može biti zaista mala skoro zanemariva, tako da su ti algoritmi dovoljno dobri za sve praktične primjene. Jedino je malo frustrirajuće znati da tamo negde postoji malo bolji raspored časova ili da postoji malo bolja mreža puteva.

Dodatno čitanje

Prethodna karikatura je zasnovana na vrlo sličnoj koja se nalazi u već spominjanoj knjizi, već klasiku, *Computers and Intractability* autora Garey i Johnson.

Rubrika Računarske rekreacije (eng. “Computer recreations”) u časopisu *Scientific American* od juna 1984 sadrži kratak opis kako napraviti Steiner-ovo stablo koristeći pjenu od sapunu zajedno sa jednim interesantnim opisom drugih sličnih “uredaja” koje možemo koristiti za rješavanje problema a koji uključuju računara od špageta za sortiranje, kolijevku za mačke od žice za nalaženje najkraćeg puta u jednom grafu kao i svjetlo-i-ogledalo uređaj koji vam može reći da li je dati broj prost ili ne. Ovakve teme se takođe pojavljuju u dijelu koji govori o analognim računarima Dewdney-evoj knjizi *Turing Omnibus*.

Dio V

**Dijeliti tajne i boriti se protiv
kriminala -*Kriptografija***