Kratice

Od tod do točke brez vrnitve

Aleksandar Jurišić in Klemen Klanjšček

→ **V vsakdanjem življenju pogosto *krajšamo* dolga imena in nazive zaradi hitrejšega sporazumevanja. Učiteljica bo pogosto na tablo ali novinarka v članku prvič zapisala neskrajšano ime, nato bo dodala tudi *kratico* in jo kasneje uporabljala. V tem besedilu bomo predstavili matematično ozadje krajšanja, kot se uporablja pri računalniški varnosti, s poudarkom na kriptografiji (npr. če želi novinar prikriti ime neke osebe, bo praviloma uporabil začetnici imena in priimka – ne nujno pravih).**

Za nas bo **krajšanje** postopek, pri katerem poljubnemu nizu znakov dodelimo neko **kratico** oz. **okrajšavo**, kratko oznako, npr. začetnice uporabljenih besed, glej sliki 1 in 2 ter tudi nalogo 2.

Primeri krajšanja

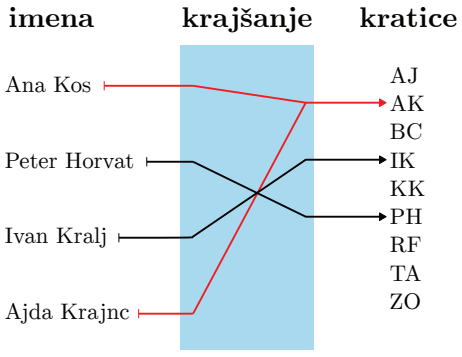
Primer 1. (Oznake) Ko učence na začetku šolskega leta razporedimo v razrede, npr. 100 učencev v 4 razrede A–D (24+23+27+26), je pogosto posamezni učenec na kratko označen npr. »A-jevec«.

Zgornji primer omogoči kuharici, da v jedilnici hitro pokliče del učencev ali pa večje skupine razdeli na manjše in jih zato lažje razporedi za deljenje malice.

Primer 2. (Začetnice) Ko se v jedilnici štirje posedejo k eni mizi, ugotovijo, da dogovor o klicanju po začetnicah imena in priimka zanje žal ne pride v poštev, glej sliko 2.

V zadnjem primeru vidimo, da je treba biti pri krajšanju precej previden.

SLIKA 1.
Stiskanje, zmešnjava, razpršenost, kot pri stiskanju grozdja ali jabolk, pri tem pa gredo nekatere stvari tudi med odpadke.



SLIKA 2.
Trčenja pri začetnicah.

Primer 3. (Konice – gr. akronimi), in (radijske) kode. Krajšanje se pogosto uporablja za hitrejše posredovanje informacij.

Omenili smo že novinarje, a tudi policija, vojska in interventne službe uporabljajo kratice, s katerimi lahko zelo hitro opišejo stanje na terenu. Če pa želimo ohraniti posredovane informacije še tajne, v kriptografiji pravimo, da gre za uporabo **kodnih knjig**. Čeprav tak pristop štejemo v kriptografiji za zastarelega, se je splošno uporabljal v drugi svetovni vojni (glej npr. nalogo 3) in je v nekaterih primerih še vedno aktualen (npr. ko celotne fraze nadomestimo s kraticami iz kodnih knjig).

Namesto gesel v računalniških sistemih hranimo le njihove »okrajšave«. **Zakaj?**. Pri shranjevanju podatkov pogosto poleg samega podatka hranimo še njegovo »okrajšavo«. **Zakaj?**.

Primer 4. (Serijske številke) Za evidentiranje naprav (knjig, ljudi ...) namesto dolgih opisov, hranimo le pripisano številko vsake naprave (npr. evidenčna, registrska, telefonska, matična, davčna, datum).

Poznamo pa tudi druge tehnično bolj zapletene primere uporabe krajšanja, kot so digitalni podpisi, denar ali zaprisege, tvorjenje naključnih števil in podobno. Ti primeri pridejo na vrsto v kakšnem nadaljevanju.

Vsaka tehnika krajšanja ima svoje prednosti in slabosti, zato po navadi ni namenjena splošni uporabi. Besedi krajšanje in kratica nista niti nujno najbolj primerni za vsako tehniko, zato omenimo še nekaj možnosti. Najbolj pogosta je **zgoščevanje** in **zgostitev** (angl. hash). Za rezultat zgoščevanja pa so tu še *prstni odtis*, *izvleček* in *povzetek*.

Zaželene lastnosti krajšanj

Predstavljamo nekaj zaželenih lastnosti tehnike krajšanja, ki so uporabne na splošno:

1. **Določenost** — *krajšanje istih izrazov vedno pripelje do iste okrajšave.* Enostavna posledica te lastnosti je, da različni okrajšavi dveh (neznanih) izrazov pomenita, da morata biti izraza zagotovo različna.
2. **Učinkovitost** — *postopek krajšanja je hiter in enostaven (vsaj za računalnike).* Daljše izraze bomo sicer običajno krajšali dlje, npr. računanje vsote števk števila za potrebe ugotavljanja njegove deljivosti s 3.
3. **Enosmernost** — *iz same okrajšave (brez dodatnih informacij) je težko najti izraz, ki smo ga krajšali.* Iz okrajšave ne moremo sklepati, kako se glasi neokrajšani izraz. Posledično mora biti vsak delček okrajšave odvisen od vseh delov vhoda ali pa od nobenega (v kriptografiji tej lastnosti pravimo *zmešnjava* – angl. confusion), npr. zadnja števka vsote števk neznanega števila.
4. **Skoraj brez trčenj** — *možnost, da imata dva različna izraza enako okrajšavo, je majhna.* Tako je npr. pri pravih/fizičnih prstnih odtisih. Dodatna zahteva: *najti dva različna izraza z isto okrajšavo je težko.* Posledično mora celoten izraz vplivati na izbrano okrajšavo (v kriptografiji tej lastnosti pravimo *razpršenost* – angl. diffusion).

Lastnosti 1 in 2 držita za večino krajšanj, če pa k temu dodamo še lastnosti 3 in 4, potem smo z eno nogo že vstopili v deželo (tete) kriptografije. A preden gremo po tej poti, omenimo še nekaj zanimivih primerov.

Hranjenje velikih datotek

Najprej smo na telefonu shranjevali samo številke in še to kar v SIM-kartici. Ko pa je telefon postal še fotoaparat, predvajalnik glasbe ter celo filmov, lahko hitro zapolnimo še tako velik pomnilnik. Ker so za nas podatki pomembni (po navadi šele, kadar jih izgubimo), smo se naučili delati varnostne kopije. Če pri tem nismo pazljivi, zelo hitro porabimo ves prostor tako na telefonu kot na računalniku ali v oblačni storitvi in se pri tem ne zavedamo, da hranimo vse več kopij z isto vsebino. Pojavi se še problem različnih verzij in kaj hitro se nam lahko zgodi, da s staro verzijo »povozimo« novejšo datoteko. Pri reševanju teh problemov si pomagamo tako, da poleg datotek hranimo še njihove zgostitve, preko katerih znamo učinkovito (lastnost 2) ugotoviti:

1. zelo verjetno imamo na disku več kopij iste datoteke (lastnost 4),
2. pri prenosu datotek je/ni prišlo do napak (lastnosti 1 in 4).

Ker je hitrost prenosa pogosto omejena, je hitrost primerjanja še bolj pomembna. V nekem smislu je

prva zgostitev ime datoteke, naslednji sta njena velikost in datum zadnje spremembe (celo kreiranja ali dostopa), nato pa si pomagamo še s kakšnim bolj zapletenim zgoščevanjem.

Preverjanje celovitosti podatkov

Kako bi preverili, ali manjka kakšna stran iz dnevnika ali kakšno sporočilo (recimo pri nekem dopisovanju)? Običajna rešitev je, da uvedemo števce, a se nato zaplete, če želimo dodati novo stran ali kaj izbrisati. Na nivoju besedila je lahko pomemben že en sam dodani, izbrisani ali spremenjeni znak (npr. ocena v spričevalu) oz. beseda/stavek (npr. v pogodbi). Vse te probleme celovitosti podatkov rešujemo z zgostitvami. V tem primeru je zelo pomembno, da tudi najmanjša sprememba v vhodnem nizu vrne drugačno zgostitev (npr. t. i. *vsota za preverjanje* – angl. checksum). Uporablja se pri prenosu datotek (npr. z Bittorrent protokoli), upravljanju z varnostnimi kopijami, hitrem iskanju po zbirkah podatkov/dokumentih itd. Verjetno pa je najbolj pomembna uporaba pri digitalnih podpisih, saj daljša sporočila podpisujemo tako, da v resnici podpišemo njihove zgostitve.

Hranjenje (uporabniških) gesel

O geslih smo že pisali v članku Ključi (v Preseku), vendar takrat še nismo omenili postopkov zgoščevanja. Tokrat pojdimo korak dlje. Namesto da bi računalniški sistemi (ki nam omogočajo dostop do osebnega računalnika, spletne učilnice, forumov ...) shranili uporabniška gesla, hranijo le njihove zgostitve. Na ta način gesla ostanejo skrita, tudi če si napadalec pridobi dostop do sistema. Ker bodo ista gesla imela iste zgostitve (lastnost 1), je tak načni shranjevanja gesel nepopoln. Napadalec lahko vnaprej izračuna/sestavi tabelo vseh možnih parov

zgostitev gesla g_1	geslo g_1
⋮	⋮
zgostitev gesla g_n	geslo g_n

za zbirko pogostih gesel g, urejeno po zgostitvah. Tako zlahka razbije manj varna gesla – za vsako zgostitev v shrambi preveri, ali se nahaja v tabeli. Če njegova zbirka vsebuje vse besede iz slovarja, v kriptografiji rečemo, da gre za **napad s slovarjem**. Omenimo še napad z *mavričnimi tabelami* (velikosti nekaj 100 GiB), s katerim lahko razbijemo skoraj vsa krajša gesla (recimo z manj kot 9 znaki – tudi posebnimi).

Kako preprečimo možnost hitre primerjave takšnih gesel prek pripadajočih zgostitev?

Ena možna rešitev je, da za vsakega uporabnika izberemo drugačen postopek zgoščevanja, ki ga tudi hranimo. V tem primeru bi bilo potrebno za ugibanje gesel sestaviti zgornjo tabelo za vsakega uporabnika posebej.

Igra kamen-škarje-list

Okoli 2000 let staro kitajsko igro po navadi igrata dva igralca takole:
(1) vsak od igralcev si na skrivaj izbere eno izmed treh možnosti *kamen/škarje/list*,
(2) igralca drug drugemu istočasno razkrijeta izbiro,
(3) zmagovalca se določi po pravilu:

- *kamen* uniči *škarje*,
- *škarje* prerežejo *list*,
- *list* pokrije *kamen*,
- v vseh drugih situacijah imamo izenačenje.

Za poštenost igre je ključnega pomena istočasnost. Lahko jo zagotovimo s sodnikom.

Kaj pa če nimamo sodnika ali pa želimo igro igrati preko telefona/interneta?

Podobna vprašanja si je postavljal že leta 1982 Manuel Blum — glej nalogo 5.

Problem istočasnosti lahko rešimo z uporabo **kriptografsko varne zgoščevalne funkcije**. Takšna funkcija za poljubno zaporedje znakov x učinkovito (z računalnikom) izračuna zgostitev $z := F(x)$. Zaradi predstavitve medijev v digitalni obliki je lahko x tudi slika, video ... Veljati mora tudi, da je praktično nemogoče uganiti/izračunati dva različna izraza z enakima okrajšavama ali pa vrednost x iz zgostitve z (tudi za najzmogljivejše računalnike) [5].

Primer poteka igre med Ano in Bojanom (brez sodnika):

[5] Ti dve možnosti sta manjši, kot da 3-krat zapored zadenemo sedmico na Lotu, in to s prvimi tremi kupljenimi srečkami!

(1) Ana na skrivaj izbere (dovolj dolg) naključni niz `H6FUcfqtAGf3` in *list*, Bojan pa izbere `uAD22TWTNTz2` in *škarje*.
(2) Ana izračuna (z računalnikom) zgostitev $z_A = F($`H6FUcfqtAGf3 list`$)$, Bojan pa izračuna $z_B = F($`uAD22TWTNTz2 škarje`$)$.
(3) Izmenjata si zgostitvi z_A in z_B.
(4) Izmenjata si naključna niza (lahko pa tudi še izbiri).
(5) Določita izid igre. (**Kako?**)

Hranjenje množic (na papirju oz. računalniku – beri: digitalnem mediju)

Pred več kot pol stoletja smo v šole hkrati s števili (aritmetiko) začeli uvajati še množice (glej `https://en.wikipedia.org/wiki/New\_Math`). Zato je čisto prav, da nanje pomislimo danes, ko smo že povsem obkroženi z računalniki, še iz tega zornega kota. Tu se namreč pojavijo zanimivi problemi, ki so povezani z našim krajšanjem.

Kaj odgovorimo na vprašanje, ali je neki element v množici?

Če je množica majhna in »vidimo« vse njene elemente, potem to najbrž sploh ni težko. Če je elementov več, moramo izbrati neko drugo primerno strategijo. Kako bi na primer ugotovili, ali je neka beseda slovenska oziroma, da pripada množici vseh slovenskih besed. Kako bi na to vprašanje odgovoril stršček Google, če nanj ne bi bil pripravljen? Kako naj zbere vse slovenske besede, če vseh sploh ne pozna. Za občutek omenimo še, da imajo najbolj bogati jeziki tudi do četrt milijona besed (pa pri tem ne štejemo sklonov).

Če se da elemente urediti po vrsti, potem jih radi hranimo urejene, kot je to narejeno z besedami nekega jezika v slovarjih (*leksikografska ureditev*). V tem primeru je za preverjanje prisotnosti elementa v množici potrebno le nekaj primerjav, da preverimo, ali je neka beseda v slovarju (npr. metoda bisekcije). Na srečo ni potrebno prelistati vseh strani. **Zakaj?**[6]

[6]Po omenjeni metodi bisekcije bi slovar odprli na sredini in če tam ne bi našli besede, bi nadaljevali v prvi ali drugi polovici – po istem postopku vse do konca, ko bi ugotovili, ali je sploh iskana beseda v slovarju.

Kaj pa če elementov sploh ne znamo enostavno primerjati?

Recimo, da iščemo nekega vojaka, ki je storil nekaj slabega, pa se nam zdijo vsi vojaki na prvi pogled zelo podobni. Upajmo, da imamo na voljo vsaj kakšen prstni odtis nepridiprava, pravzaprav kar vseh vojakov. Lahko so na voljo le njihove slike, ki jih je posnela skrita kamera. Kako urediti po vrstnem redu slike? Če bi vojake slikali ob vstopu v vojsko, potem bi najbrž vsak dobil številko in k tej številki pripeto sliko. Če pa ne gre za naše vojake, potem je stvar še nekoliko bolj zapletena (sicer bi lahko šlo tudi za bakterije, viruse itd., a ne želimo zaiti na področje biologije ali medicine), saj smo rekli, da je naš zorni kot računalništvo. Predpostavimo, da primerjamo slike vojakov (a tu ne bomo reševali problemov računalniškega vida).

Ena možna rešitev je, da namesto elementov primerjamo njihove zgostitve. Za hitro iskanje in dodajanje elementov najprej izračunamo zgostitve vseh elementov ter sestavimo tabelo na naslednji način. Element spravimo v tisto vrstico tabele, ki jo določa njegova zgostitev. Vsaki zgostitvi namenimo svoj prostor (to je lahko list papirja, del pomnilnika ali vrstica v tabeli). Zgostitve igrajo vlogo kazala (npr. rokovnik z naslovi, zgostitev predstavljata prvi dve črki priimka – trki so tukaj dopustni, ampak nezaželeni).

Podobno tehniko uporablja sistem za spletno izmenjevanje datotek prek t. i. *magnetnih povezav* (angl. Magnet Link), ki so pretežno zgostitve h_{magnet}. Če poznamo neko zgostitev, lahko prek *Bittorrent protokolov*, prenesemo želeno datoteko (ali pa mapo z datotekami). Najprej poizvemo, kdo si trenutno deli pripadajočo datoteko. Podatek o tem, kateri računalniki si trenutno delijo datoteko vezano na dano magnetno povezavo, se hrani v velikem skupnem »imeniku« na spletu, kjer za »začetnico« imenika vzamemo zgostitev h_{magnet}, za »naslov« pa IP naslov in vrata (angl. port) računalnikov. Ker je imenik velik in se neprestano spreminja, noben računalnik nima celotnega imenika, le nekaj »strani«.

Ko najdemo računalnike, ki trenutno delijo želeno datoteko, najprej prejmemo od njih t. i. *torrent* datoteko. Ker so datoteke, ki jih na ta način izmenjujemo, ponavadi velike, jih prenašamo po manjših delih/koščkih. Zaradi tega prejeto torrent datoteko sestavljajo zgostitve vseh koščkov datoteke in njihove

velikosti, ter ime datoteke. Skupna zgostitev vseh teh treh informacij je h_{magnet} in osnovni del magnetne povezave, glej sliko 3. Ker se zgoščevanje izvaja s kriptografsko varno zgoščevalno funkcijo H, so zgostitve datotek praktično unikatne. Prednost takega načina prenosa datotek je, da lahko datoteke prejemamo od več računalnikov hkrati (od vsakega le nekaj koščkov) in smo zato neodvisni od mrežnih problemov (npr. prekinitve, napake). Namenoma nismo podali celotne zgodbe, saj je še veliko bolj zapletena. Primer magnetne povezave: `magnet:?xt=urn:btih:e0d9af6671db5bc4fc77ab5462c50e2f3545dad9`. Število s 40-timi znaki v šestnajstiškem zapisu `e0d9a`... (20 »bajtov«) tukaj predstavlja zgostitev (SHA-1) za prenos DVD-ja (velikosti 3,59 GiB = 14741 koščkov velikosti 256 KiB, torrent datoteka je velikosti 288 KiB $\approx$ 14741 $\times$ 20 B) za namestitev operacijskega sistema Ubuntu 20.04.5 (za celoten vpogled v vsebino te torrent datoteke glej povezavo `https://gist.github.com/kllemen/47c7d8782-c81afbb76bc7ff19d172588`).

Zaključek

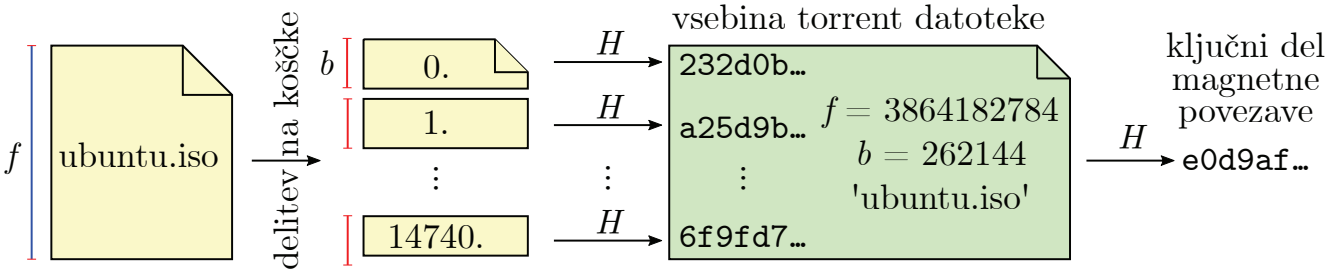
Če boste rešili vsaj nekaj spodnjih nalog, boste opazili, da se zdi na prvi pogled konstrukcija kriptografskih zgoščevalnih funkcij (pre)zahtevna naloga. V resnici sploh ni lahko najti (sestaviti) takšno funkcijo. Tudi če nam jo nekdo predlaga, ne moremo biti prepričani, da se ne bo v prihodnosti izkazalo, da ena od zahtevanih lastnosti ne drži. Prve zasnove kriptografskih zgoščevalnih funkcij segajo 45 let nazaj (glej Merkle 1979 in npr. Damgärd). V devetdesetih letih prejšnjega stoletja je zelo hitro naraslo število modelov zgoščevalnih funkcij, vendar so bile pri številnih od teh predlogov ugotovljene varnostne pomanjkljivosti, npr. najbolj popularnim Rivestovim zgoščevanim funkcijam MD4 in MD5 (angl. Message Digest) se je po več 10-letni uporabi zgodilo prav to.
Odprt problem: ali kriptografske zgoščevalne funkcije sploh obstajajo?

Ta čas sicer imamo kar nekaj kriptografskih zgoščevalnih funkcij, za katere se smatra (ali vsaj upa), da navedene lastnosti držijo in jih zato uporabljamo v praksi: BLAKE, RIPEMD (Bitcoin), SHA (za digitalno potrdila v osebni izkaznici oz. v splošni uporabi), Whirlpool itd. Teh funkcij nikoli ne računamo na roke (`https://md5calc.com/hash/sha1` bralec lahko poskusi preveriti njihovo delovanje oziroma se poigrati s simulacijo računanja funkcije `https://sha256algorithm.com/`). Kakšno med njimi bi si enkrat celo lahko pobližje ogledali. Že sedaj pa lahko omenimo:
Odprt problem: najdi zalogo vrednosti zgoščevalne funkcije MD4 ali SHA-1.

Lahko pa se vprašamo tudi, ali obstajajo kriptografsko varne zgoščevalne funkcije primerne za »ljudi«? Za zdaj kaže, da brez pomoči računalnika, vsaj za večino nas, žal ne gre.

Naloge

1. Možni ostanki pri deljenju celega števila s 3 so 0, 1 in 2. Krajšanje, ki nam za naravno število vrne njegov ostanek pri deljenju s 3, ima očitno prvi dve lastnosti (*določenost* in učinkovitost). Prepričaj se, da ni tako pri zadnjih dveh lastnostih (enosmernost oz. iskanje *praslike* in *problem trčenja*).



SLIKA 3.
Ustvarjanje magnetne povezave.

2. Teta kriptografija se je pogovarjala z mlajšo kolegico in izvedela, da bodo mladi bralci ob besedi krajšanje najprej pomislili na krajšanje ulomkov. Pri tem postopku običajno iz velikega imenovalca in števca izpostavimo njun največji skupni delitelj D in končamo po krajšanju z manjšim imenovalcem in števcem (če le ni $D = 1$), medtem ko vrednost ulomka ostane ista. Tudi pri našem krajšanju si želimo podobno – krajši zapis. Katerim od omenjenih štirih lastnosti ustreza omenjeno krajšanje ulomkov?

3. Ameriška mornarica je zbrala 29 mož indijanskega plemena Navajo (`https://www.intelligence.gov/people/barrier-breakers-in-history/453-navajo-code-talkers`) in ustvarili kode, zasnovane na njihovem zapletenem, nenapisanem jeziku. Koda je v prvi vrsti uporabljala povezovanje besed tako, da je ključnim frazam in vojaškim taktikam dodelila besedo Navajo. Če se želite preizkusiti v odšifriranju kode v jeziku Navajo, uporabite `https://www.cia.gov/stories/story/navajo-code-talkers-and-the-unbreakable-code/` za odšifriranje naslednje kode:

   `TKIN-KE-YAH YIL-DOI AH-JAH YEH-HES KLESH DIBEH-YAZZIE`

   `WOL-LA-CHEE TSAH BE TKIN YIL-DOI WOL-LA-CHEE`

   oziroma odšifrirajte spodnjo kodo, da ugotovite, v kateri bitki je koda pomagala doseči zmago ZDA:

   `Tkin-Gloe-Ih-A-Kha Ah-Ya-Tsinne-Tkin-Tsin-Tliti-Tse-Nill`

4. (a) Kaj bi bil primeren način zgoščevanja glasbe, ki bi ljudem še vedno omogočal prepoznavanje/ločevanje? (b) Katere izmed tistih štirih lastnosti si tukaj želimo? (c) Kako bi tako prepoznavanje naredil z računalnikom (torej brez uporabe naših možganov)? Na telefonu bi lahko uporabili aplikacijo *Shazam*. (d) Ali mogoče veš, kako *YouTube* portal ve, da si uporabil avtorsko zaščiteno glasbo v svojem videu (ker jih očitno ne pregledujejo »na roke« – razen morda čisto na koncu, ko pride do kakšne tožbe)?

5. Kako varno vreči kovanec prek telefona (iz članka M. Bluma, Coin Flipping by Telephone, 1982)? Ana in Bojan želita vreči kovanec prek telefona. (Nasledni mesec bosta delala v različnih mestih in želita sprejeti odločitev, kdo bo dobil skupni avto.) Bojan noče povedati Ani »*cifra*« in slišati (na drugi strani linije), kako ona odvrne »Vržem kovanec ... Izgubil si!«. Ali bi znal predelati igro *kamen-škarje-list z uporabo zgoščevalnih funkcij* tako, da rešiš Anin in Bojanov problem?

6. Kako zaščititi idejo, da nam je kdo ne ukrade in predstavi kot svojo? Recimo, da je sošolec prišel do naše seminarske naloge, še preden smo jo oddali in se je odločil, da jo bo oddal s svojim imenom. Ali obstaja način, da prepričamo učiteljico o našem avtorstvu?

7. Katere lastnosti želimo pri zgoščevalnih funkcijah za hranjenje gesel v računalniških sistemih?

   Kako bi problem primerjav gesel rešil le z eno zgoščevalno funkcijo (torej v nasprotju s predlogom v samem članku)?

8. Zakaj pri shranjevanju podatkov pogosto poleg podatka hranimo še njegovo *okrajšavo*? (Namig: kako preverimo, ali podatek ni pokvarjen, zakaj je iskanje po velikem disku lahko dolg proces, kako lahko na hitro primerjamo, kaj imamo shranjeno v oblaku in kaj lokalno na svojem domačem računalniku — ali lahko ugotoviš, kaj ti sistemi v ozadju počnejo.)

9. Poišči način krajšanja, ki ima lastnost 3, ne pa tudi lastnosti 4.

10. Poišči način krajšanja, ki ima lastnost 4, ne pa tudi lastnosti 3.

11. Zakaj lahko pri prenosu datotek prek BitTorrent protokola, sproti preverjaš ali si doslej prejel pravilne/nepokvarjene kose datotek.

12. Kako se pri igri kamen-škarje-list odkrije goljufanje? Čemu služi naključen niz, ki si ga Ana

in Bojan izbereta na začetku? Kaj gre lahko narobe, če ta niz ni naključno izbran? Zakaj Ana ne more iz vrednosti zgostitve z_B ugotoviti Bojanove izbire? Zakaj lahko izmenjavo zgostitev z_A in z_B izvedemo z zamikom? Kako bi se dalo goljufati, če za funkcijo F lastnost 4 ne bi držala?

13. Kako so shranjena gesla za dostop do spletne učilnice na vaši šoli oz. tvojega računalnika?

14. Za kriptografske zgoščevalne funkcije skoraj vedno velja, da če se spremeni en sam bit vhoda, se mora spremeniti približno polovica bitov na izhodu. Zakaj je to pomembno?
Ali bi bil problem, če bi se v izhodu spremenili skoraj vsi biti?

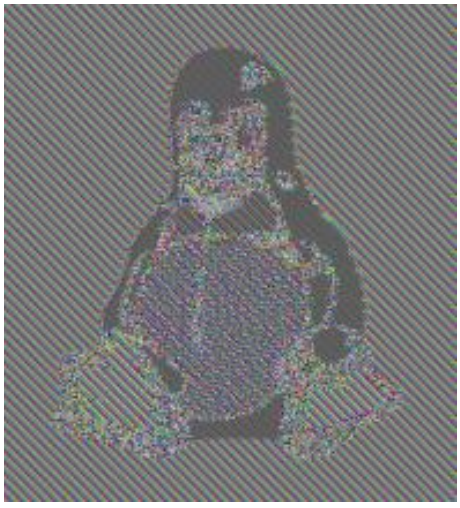
15. Uporabi seštevanje ali množenje števil (v $\mathbb{Z}$ ali $\mathbb{R}$) za sestavo nove zgoščevalne funkcije in jo analiziraj. Enosmernost nam zagotavlja problem iskanja seštevancev ali faktorjev. Ali je uporaben poseben primer, kadar vemo, da gre za enaka števila?

16. Ali je krajšanje, ki vrača začetnici imena, kriptografsko varna zgoščevalna funkcija? Možnost, da imata dva izmed petih učencev enake začetnice, je sicer majhna, 2,7 %, a v svetu zgoščevalnih funkcij je to ogromno. Za primer lahko naredimo tudi scenarij, da je v razredu 20 učencev. Takrat je možnost, da imata dva učenca enak izhod, že 57,1 % (z upoštevanjem, da niso vsa imena in priimki enako verjetni; če tega ne upoštevamo, zmanjšamo možnosti na 1,0 % pri 4 učencih in 26,5 % pri 20 učencih). Kako smo prišli do teh ocen?

17. Bločne šifre (najbolj slavna je danes AES) razdelijo tekst, ki ga želimo šifrirati, na bloke enake dolžine (npr. 128 bitov), nato pa šifrirajo vsak blok posebej Če to naredimo z enim samim ključem K, se identični bloki sporočila $m = m_1 m_2 \ldots m_t$ zašifrirajo v identične bloke tajnopisa:

   - $c = c_1 c_2 \ldots c_t$, kjer je $c_i = \mathrm{AES}_K(m_i)$,

   kar je lahko problem, glej (zašifrirano) sliko 4.



SLIKA 4.
Zašifrirana slika znanega Linuxovega pingvina.

Zato včasih šifriranje nekoliko prepletemo, npr. po postopku

- $c_0 = \mathrm{IV}$, $c_i = \mathrm{AES}_K(c_{i-1}) \,\mathrm{XOR}\, m_i$, za $i \geq 1$,

kjer je IV neka (javno) znana konstanta, bitni XOR pa v računalništvu zelo znana operacija, ki ji v slovenščini rečemo *ekskluzivni ali*. Postopek se je dolgo uporabljal za preverjanje celovitosti sporočila. Kako? Bi ga znal zapisati? Običajno zaključimo pismo s kakšno običajno frazo (Lep pozdrav) in če to napadalec ve, potem se da to izkoristiti (brez poznavanja ključa K). V bistvu lahko zadnji blok zamenja s poljubnim svojim. Bi znal pokazati, kako to stori?

18. Nad enosmernostjo se zamislimo še zunaj sveta računalnikov. Kateri pojavi iz našega običajnega življenja imajo lastnost enosmernosti?

× × ×