ŠTÁTNY PEDAGOGICKÝ ÚSTAV

MATURITA Z INFORMATIKY 2018/2019

Prehľad zmien a námety na tvorbu maturitných zadaní

Metodická pomôcka pre učiteľov informatiky

BRATISLAVA 2018

Mgr. Peter Kučera

Ministerstvo školstva, vedy, výskumu a športu Slovenskej republiky dňa 21. 12. 2016 schválilo nový Katalóg cieľových požiadaviek na maturitnú skúšku z informatiky s platnosťou od 1. 9. 2018 [1]. Cieľom tohto dokumentu je poukázať na najdôležitejšie zmeny v súvislosti s maturitnou skúškou a prípravou maturitných zadaní z informatiky.

V tomto dokumente nájdete:

- najdôležitejšie zmeny v porovnaní s predchádzajúcimi Cieľovými požiadavkami na vedomosti a zručnosti maturantov z informatiky (ďalej len CP),
- ukážky maturitných zadaní,
- komentáre k ukážkam maturitných zadaní.

Zmeny v CP

V nových CP [1] sú upravené názvy tematických okruhov tak, aby boli v súlade s názvami tematických oblastí štátneho vzdelávacieho programu platného od 1. 9. 2018.

Váha hodnotenia dvoch úloh maturitného zadania sa zmenila z pôvodných váh 50 % a 50 % na váhu 70 % pre prvú úlohu a 30 % pre druhú úlohu maturitného zadania. Pod zmenou váhy hodnotenia rozumieme len samotnú zmenu hodnotenia jednotlivých úloh, **nie** zmenu rozloženia ich časovej náročnosti. Hodnotenie maturitného zadania momentálne definuje príslušná vyhláška. Treba si uvedomiť, že platí vyššia právna úprava, a teda pokiaľ sa vyhláška nezmení, zostáva v platnosti pôvodná váha hodnotenia.

V CP sa uvádza: *„Žiaci na maturitnej skúške nepreukazujú zručnosti v ovládaní digitálnych technológií, ale schopnosti riešiť algoritmické problémy a úroveň vedomostí zo základov informatiky (žiak by mal vysvetliť ideu, princíp fungovania).“* To v praxi znamená, že cieľom zadania nie je overovať konkrétne digitálne zručnosti, napr. pri práci v textovom editore, grafickom editore, prezentačnom softvéri a pod..

Prvá úloha maturitného zadania

V prvej úlohe maturitného zadania (programátorskej) má byť definovaný cieľ, ktorý má žiak dosiahnuť vytvorením programu v konkrétnom programovacom jazyku. V zadaní úlohy **nemajú** byť prezradené prostriedky, ktorými sa má dosiahnuť cieľ. Výber prostriedkov si určuje žiak a ich správny výber je súčasťou hodnotenia. To znamená, že v úlohe nemá byť prezradené, či sa na riešenie má použiť napríklad cyklus a ktorý cyklus, a tiež ktorý typ premennej alebo dátovej štruktúry sa má v programe využiť.

Odporúčame prvú úlohu maturitného zadania členiť na menšie podúlohy (alebo nejaké vlastnosti programu), a tie zapísať v odrážkach. Často sú tieto odrážky gradovanými čiastkovými podúlohami celej úlohy. Toto členenie má pomôcť aj slabšiemu žiakovi vyriešiť aspoň časť úlohy. Napríklad v úlohe s čítaním textového súboru môže byť v prvej odrážke prečítanie prvého riadku súboru a jeho vypísanie, čím žiak preukáže, že vie minimálne správne otvoriť vstupný súbor a prečítať prvý riadok. Až podúloha v ďalšej odrážke môže požadovať prečítanie a spracovanie celého súboru. Tiež sa môže stať, že prvá odrážka popisuje podmnožinu úlohy, ktorú treba spraviť v ďalšej odrážke. V niektorých takýchto zadaniach, ak vieme, stačí vyriešiť náročnejšiu časť popísanú v ďalšej odrážke (čím

preukážeme aj vyriešenie elementárnejšej úlohy, o ktorú sa zachytíme v prípade, že náročnejšiu časť nevieme vyriešiť).

Zmeny v prvej úlohe maturitného zadania

1. Zmenil sa popis odporúčaných programovacích jazykov. Aj keď pôvodné CP nezakazovali konkrétny programovací jazyk, formulácia sa rozšírila. Podľa pôvodnej formulácie si niektoré školy vysvetľovali, že je možné maturovať len v jazyku Pascal. Nové CP odporúčajú Pascal alebo Python (prípadne aj C++, Java, C# a pod.). Filozofiou informatiky vždy bolo, že **škola** si vyberá konkrétne nástroje (softvér) a programovací jazyk. Najlepšie je, keď si škola vyberá jazyk, v ktorom prebiehalo vyučovanie. Niekedy sa stretávame s otázkou, či môže žiak maturovať aj v inom jazyku ako je zvyklosť alebo očakávanie konkrétnej školy. Formálne výber iného jazyka (voči skupine, ktorá maturuje na danej škole) konkrétnym študentom nie je v rozpore s CP. No je dôležité, aby o tejto skutočnosti žiaci s učiteľmi komunikovali už vopred, pretože to ovplyvňuje prípravu techniky na maturitnú skúšku, ale môže to znamenať aj zmenu formulácie zadaní. Náročnosť úlohy maturitného zadania súvisí aj s výberom programovacieho jazyka, na ktorý je úloha stavaná.
2. Obsah tematického okruhu Algoritmické riešenie problémov je podrobnejšie formulovaný ako v predchádzajúcich CP.
3. Najvýraznejšou zmenou je vyradenie dvojrozmerných polí a záznamov. V praxi to znamená, že niektoré zo zadaní obsahujúce dvojrozmerné pole alebo záznam (vytvorené podľa predchádzajúcich CP) sa dajú použiť aj teraz. Podmienkou je, aby sa dali riešiť aj bez použitia dvojrozmerného poľa alebo záznamu, a zároveň čas potrebný na ich riešenie takýmto spôsobom nepresiahol optimálnu časovú náročnosť prvej úlohy. Môže sa stať, že žiak zadanie rieši s využitím štruktúr alebo poznatkov, ktoré sú nad rámec CP. Maturitná komisia však nemôže očakávať také riešenie. Ak ho žiak predsa len použije, komisia ho nemá dôvod zakázať, no žiak ide do rizika časovej tiesne, lebo v prípade chýb v tomto riešení sa komisia bude pýtať na riešenie, ktoré je v rozsahu CP. Ak si žiak vyberie také riešenie, počas odpovede na maturitnej skúške je priestor aj na diskusiu o iných riešeniach a následne zhodnotenie požadovaného obsahu a výkonu podľa CP.
4. V niektorých výkonoch CP sú v zátvorke uvedené príklady. Ich cieľom je len ozrejmiť daný výkon a priblížiť ho čitateľovi. Neznamená to, že práve tieto úlohy má študent vedieť riešiť, resp. že práve tieto máme skúšať. Výber konkrétnych úloh sa opäť necháva na zvyklosti danej školy, napr. ak škola vo vyučovaní preferovala grafický kontext riešenia úloh, je vhodné ho preferovať aj na maturitnej skúške. Nemalo by sa stať, že charakter resp. kontext úlohy je v istom zmysle pre žiaka úplne iný ako charakter a kontext úloh počas vyučovania. Týmto neskúšame žiaka vedomosti a schopnosti, na ktoré sme ho priebežne pripravovali. Samozrejme, to neznamená, že ho skúšame presne rovnaké úlohy, s ktorými sa stretol na vyučovaní.

Zmeny v druhej úlohe maturitného zadania

V druhej úlohe zadania sú zmeny zásadnejšie. Podľa CP: „*Druhá úloha (Základy informatiky) má váhu 30 %. Má byť prierezová tak, aby zasahovala aspoň do troch oblastí. Väčšinou pozostáva z riešenia úlohy, ktorá má algoritmický charakter, pomocou rôznych nástrojov informatiky (žiak nemusí použiť programovanie).*“

1. Druhá úloha v minulosti neobsahovala obsah a výkon z tematického okruhu Algoritmické riešenie problémov. Teraz môže zadanie zasahovať do ktoréhokoľvek okruhu zo všetkých

piatich okruhov (aj do okruhu Algoritmické riešenie problémov). Väčšina úloh má pozostávať z riešenia úlohy, ktorá má algoritmický charakter. To znamená, že pri tridsiatich maturitných zadaniach by aspoň 16 druhých úloh zadaní malo spĺňať túto požiadavku, no môžu aj všetky.

2. Požiadavku na prierezové druhé úlohy zadania už obsahovali CP platné v školských rokoch 2011/2012 a 2012/2013. Rovnako mali zasahovať aspoň do troch oblastí (okruhov), ale pre druhé zadanie sa používali len štyri oblasti (okruhy). Teraz majú zasahovať do troch okruhov z piatich (pribudol medzi ne okruh Algoritmické riešenie problémov).
3. V obsahu a výkonoch sú zvyšné štyri okruhy výrazne stručnejšie ako v minulosti. Chýbajú v nich dôkladné a presné formulácie.
4. Niektoré časti obsahu a výkonov sa presunuli do iných okruhov. Napríklad počítačové siete sa presunuli z okruhu Hardvér a softvér (pôvodný názov) do okruhu Komunikácia a spolupráca.

Dôvodom zmeny požiadaviek druhej úlohy maturitného zadania bola skutočnosť, že niektoré maturitné zadania overovali najmä zručnosti v ovládaní digitálnych technológií (napríklad prácu s textovým editorom, grafickým editorom, prezentačným softvérom a podobne). Mnohé z týchto zručností má žiak zvládnuť v priebehu základnej školy, a aj preto nie je dôvod, aby sme ich overovali na maturitnej skúške.

Čo znamená, že úloha má algoritmický charakter a žiak nemusí použiť programovanie?

Z nášho pohľadu sú to úlohy, kde je potrebné vyriešiť nejaký konkrétny problém, ktorý sa dá riešiť aj bez použitia programovacieho jazyka. Napríklad riešenie sa dá zrealizovať pomocou vzorcov v tabuľkovom procesore, využitím online nástrojov na internete, využitím nástrojov na hľadanie a nahradenie textu a pod.. Zadaním môže byť napr. transformácia nejakého vstupu v konkrétnej zadanej forme do požadovaného výstupu.

Na riešenie takej úlohy môže žiak potrebovať napríklad analyzovať problém, navrhnúť postup, ktorý vedie k požadovanému výstupu, prípadne celý postup alebo jeho časť zrealizovať, pomenovať v danom postupe kritické situácie, odhadovať možné chyby a navrhnúť spôsob, ako im predchádzať. Takéto požiadavky spĺňajú aj niektoré úlohy (alebo ich časti), ktoré sa bežne používajú v prvej úlohe maturitného zadania a dajú sa riešiť aj inak ako priamo v programovacom jazyku. Tiež to spĺňajú úlohy, ktoré je náročné (či už časovo na samotný zápis programu alebo z dôvodu použitia zložitejších dátových štruktúr, náročným technickým postupom a pod.) zrealizovať v konkrétnom programovacom jazyku.

Úlohy s algoritmickým charakterom zasahujú do okruhu Algoritmické riešenie problémov a často aj do okruhu Reprezentácie a nástroje (kódovanie informácie na vstupe a výstupe). Doplnením otázky na vysvetlenie, ako sa daný postup bude vykonávať na danom hardvéri, alebo zistením informácií o vykonávaní tohto procesu sme v zadaní pokryli ďalší okruh (Softvér a hardvér). Do zadania môžeme napr. doplniť otázky, či a ako je možné riešiť tento problém s využitím kolaboratívnych nástrojov, a zadaním zasahujeme do okruhu Komunikácia a spolupráca. Prípadne sa v zadaní budeme pýtať na bezpečnostné riziká riešenia problému alebo jeho vzťah k ochrane súkromných údajov a pokryli sme oblasť Informačná spoločnosť. Takto relatívne ľahko vytvoríme zadanie, ktoré zasahuje aspoň do troch okruhov. Zároveň je zadanie komplexné a umožňuje nám odhaliť, ako sa žiak orientuje v rôznych súvislostiach. Aj keď sa zdá, že otázky z iných okruhov môžu byť niekedy veľmi podobné, odpoveď na ne vychádza z konkrétnej situácie a kontextu zadaného problému, a teda v konečnom dôsledku sú odpovede rôzne.

Algoritmický charakter má aj typ úlohy, ktorá obsahuje nejaký krátky odborný text (napríklad krátka správa z oblasti IT zverejnená na internete) (viď úloha 2.5). V texte je napríklad popis nejakej novej funkcionality softvéru (komunikačnej služby a pod.) a úlohou je analyzovať danú funkcionalitu, odhadnúť jej možné spôsoby riešenia, popísať kroky, ktoré musí funkcia vykonať, určiť ich poradie, odhadnúť potrebnú reprezentáciu dát a pod..

Pokiaľ chceme pracovať s nejakým poznatkom, ktorý žiak nemusí poznať (podľa CP), môžeme ho v úlohe aj predstaviť a následne použiť. Treba však pamätať na primeranosť dĺžky zadania a časovú náročnosť.

Samozrejmé je, že na akýkoľvek typ úloh je potrebné žiaka priebežne pripravovať a maturitné zadanie nemá byť svojou formou, štruktúrou alebo komplexnosťou pre žiaka prekvapením a zásadnou zmenou voči zvyklostiam počas celého štúdia. Opäť zdôrazňujeme, že to neznamená, že žiak sa už mal stretnúť s konkrétnou úlohou.

Ukážky prvej úlohy maturitného zadania

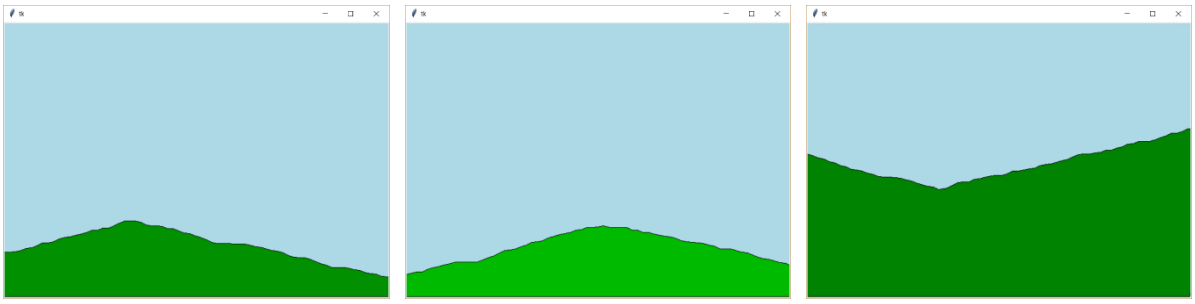
Samotné znenie úlohy je závislé od konkrétneho programovacieho jazyka. Je potrebné si uvedomiť, že použitie konkrétneho jazyka môže zmeniť aj kognitívnu obťažnosť úlohy, ale ešte výraznejšie časovú náročnosť na riešenie danej úlohy. Pripomíname, že je dôležité, aby sa žiaci s podobným kontextom úloh stretli už počas štúdia a tiež, že k príprave je potrebný aj tréning riešenia úloh s podobnou náročnosťou. Ak žiak nie je na maturitné zadania pripravený, časová náročnosť na riešenie úlohy sa môže výrazne zvýšiť. Nasledujúce ukážky sú stavané pre jazyk Python. Viac prvých úloh maturitných zadaní pre jazyk Python nájdete aj v zbierke riešených maturitných úloh s názvom Maturujeme v Pythone [4].

1.1 Krajina

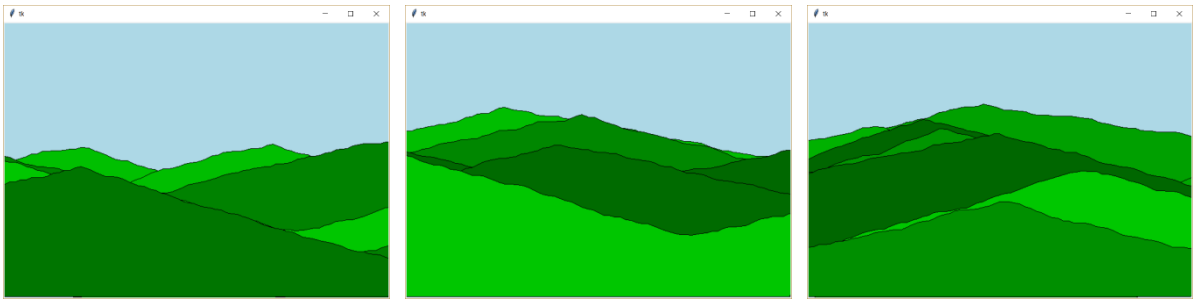
Pomocou programu môžeme generovať a vykresľovať náhodnú krajinu. Vytvorte program, ktorý:

- vygeneruje údaje pre kopec, pričom si najprv náhodne určí x-ovú pozíciu vrcholu a y-ovú súradnicu počiatočnej výšky kopca. Pre kopec platí, že jeho výška je v prvej časti (pred vrcholom) neklesajúca a za vrcholom nerastúca. Zmena v reliéfe kopca môže byť každých 10 bodov a je náhodná voči predchádzajúcemu stavu výšky.
- vykreslí kopec pomocou príkazu canvas.create_polygon(). Farba kopca je niektorý náhodný odtieň zelenej farby.
- sa náhodne rozhodne, či ide kresliť kopec alebo údolie (či je najprv neklesajúca postupnosť a potom nerastúca alebo opačne) a vykreslí jeden kopec alebo jedno údolie.
- opakovane vykreslí viac náhodných kopcov / údolí, čím vznikne vygenerovaná krajina,
- po stlačení medzery nakreslí novú sériu náhodných kopcov a údolí.

Ukážka jedného kopca alebo jedného údolia:



Ukážka viacerých kopcov a údolí:



Riešenie úlohy:

```
import tkinter, random
canvas = tkinter.Canvas(width=700, height=500, bg='lightblue')
canvas.pack()

def kresli_kopec():
    kopec = []
    smer = random.choice((1,-1))              #  {riadok A}
    kopec.append(0)
    kopec.append(random.randint(200, 500))
    vrchol = random.randint(100, 600)
    for i in range(vrchol // 10):
        nova_hodnota = kopec[-1] + smer * random.randint(0, 5)
        kopec.append(i*10+1)
        kopec.append(nova_hodnota)
    smer = -1 * smer
    for i in range((700-vrchol) // 10 + 10):
        nova_hodnota = kopec[-1] + smer * random.randint(0, 5)
        kopec.append(i*10+vrchol)
        kopec.append(nova_hodnota)

    kopec = [0, 500] + kopec + [700, 500]
    farba = '#00{:02x}00'.format(random.randint(100, 200)) #  {riadok B}
    canvas.create_polygon(kopec, fill=farba, outline='black')

def prekresli(event):
    canvas.delete('all')
    for i in range(10):
        kresli_kopec()

#kresli_kopec()
canvas.bind_all('<space>', prekresli)
```

Možné otázky počas odpovede k tomuto konkrétnemu riešeniu úlohy:

1. Čo spôsobí zápis na riadku {riadok A}? Ako inak by sme ho mohli zapísať?
2. Vysvetlite zápis na riadku {riadok B}.
3. Ako je vyriešená zmena stúpania medzi prvou časťou kopca a druhou časťou?
4. Koľko informácií o kopcoch je v pamäti počas behu tohto programu? Je možné optimalizovať využitie pamäte? Ak áno, ako?

1.2 Analýza údajov

Lokálne potraviny sa rozhodli, že zistia spokojnosť s poskytnutými službami u svojich zákazníkov. Pri východe z predajne nainštalovali box, v ktorom zákazník vyjadrí svoju spokojnosť / nespokojnosť pomocou dotykovej obrazovky. Všetky vyjadrenia sa zapíšu do textového súboru. Vyjadrenia sú v textovom súbore spokojnost_1.txt (k dispozícii sú aj súbory spokojnost_2.txt a spokojnost_3.txt). Na každom riadku je jedno vyjadrenie zákazníka. Vyjadrenie obsahuje čas zaznamenania v tvare hodina:minúta, potom nasleduje jedna medzera a text áno alebo nie podľa toho, či bol zákazník spokojný alebo nespokojný. V súbore sú vyjadrenia z viacerých dní.

Ukážka vstupného textového súboru:

```
15:38 áno
15:39 áno
14:33 áno
08:38 áno
07:42 áno
15:20 áno
```

Ukážka výstupu:

```
1. deň - počet reakcií:2
2. deň - počet reakcií:1
3. deň - počet reakcií:1
4. deň - počet reakcií:2
Počet všetkých vyjadrení: 6
Hodina:7 Reakcií zákazníkov:1
Hodina:8 Reakcií zákazníkov:1
Hodina:14 Reakcií zákazníkov:1
Hodina:15 Reakcií zákazníkov:3
Počet dní: 4
```

Vytvorte program, ktorý zistí a vypíše:

- celkový počet vyjadrení,
- počet všetkých vyjadrení v jednotlivých hodinách dňa, ale iba v tých hodinách, keď boli nejaké vyjadrenia,
- počet dní, počas ktorých sa zbierali vyjadrenia (predpokladajme, že každý deň bolo zaznamenané aspoň jedno vyjadrenie a že vstupné dáta sú zapísané v takom poradí, ako boli zrealizované),
- počet vyjadrení v jednotlivých dňoch.

Riešenie úlohy:

```
vyjadreni = [0] * 24

subor = open('subory/spokojnost_0.txt', 'r')
dni = 0
cas1 = '00:00'
pocet_v_dni = 0
for riadok in subor:
    riadok = riadok.strip()
    info = riadok.split()
    cas2 = info[0]
    cas2_roz = info[0].split(':')
    hodina = int(cas2_roz[0])
    minuta = int(cas2_roz[1])
    vyjadreni[hodina] += 1
    if cas2 < cas1:
        dni += 1
        print('{}. deň - počet reakcií:{}'.format(dni, pocet_v_dni))
        pocet_v_dni = 1
    else:
        pocet_v_dni += 1
    cas1 = cas2
print('{}. deň - počet reakcií:{}'.format(dni, pocet_v_dni))
pocet_vyjadreni = sum(vyjadreni)
print('Počet všetkých vyjadrení:', pocet_vyjadreni)
for i in range(24):
    if vyjadreni[i] > 0:
        print('Hodina:{} Reakcií zákazníkov:{}'.format(i, vyjadreni[i]))
print('Počet dní:', dni)
```

Možné otázky počas odpovede k tomuto konkrétnemu riešeniu úlohy:

1. Podľa čoho vieme, že záznam je z ďalšieho dňa?
2. Ako funguje porovnávanie časov?

Ukážky druhých úloh maturitného zadania

2.1 V súbore stretnutie.txt (stretnutie.xlsx) sú zakódované znaky textu – tajného termínu stretnutia. Rozkódujte tieto znaky a zistite dohodnutý termín. Vysvetlite princíp prevodu čísla z dvojkovej sústavy do desiatkovej a späť. Popíšte, akým spôsobom sú reprezentované v počítači znaky.

Vysvetlite, ako počítač pracuje pri sčítaní dvoch čísel. Ktoré časti počítača túto činnosť zabezpečujú? Prečo počítač spracováva naraz čísla, ktorých veľkosť je násobkom bajtu? Zistite, aké dve veľké čísla vie v jednom kroku sčítať váš počítač, a vysvetlite, od čoho to závisí.

Ukážka súboru:

1110011
1110100
1110110
1110010
1110100
1101111
1101011
110001
110100
111010
110001
110101

Poznámky k úlohe:
V úlohe treba navrhnúť a zrealizovať postup, ktorým získame text tajného termínu stretnutia. Na vstupe sú čísla v binárnej sústave, ktoré reprezentujú kód znakov.
Úlohu môžeme riešiť viacerými spôsobmi:

a) Pomocou tabuľkového kalkulátora. Najprv čísla pomocou funkcie bind2dec() zapíšeme v desiatkovej sústave a následne ich funkciou char() zameníme na znak.
b) Nájdeme nejaký online konvertor a pomocou neho získame výstupný text.
c) Postup prevodu na text môžeme popísať a zrealizovať aj „ručne“.

Touto časťou úlohy sme splnili, že má algoritmický charakter, zároveň zasahuje do okruhov Algoritmické riešenie problémov, Reprezentácie a nástroje.
Zvyšnými otázkami v úlohe zasahujeme do okruhu Softvér a hardvér. V riešení je potrebné popísať fungovanie procesora.

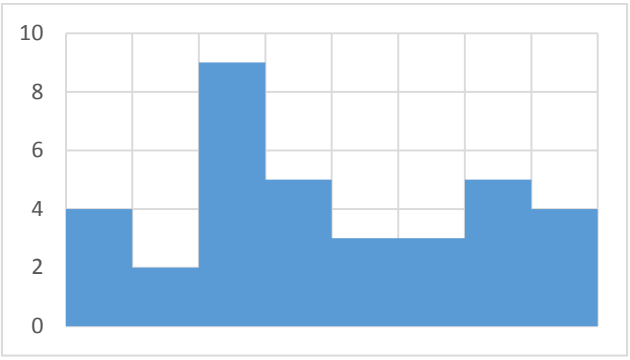
2.2

V tabuľke plan_mesta.xlsx je časť mesta rozdelená na štvorcovú sieť a v každom políčku je zapísaná informácia o výške budovy na danom mieste. Vytvorte grafické zobrazenie výškového profilu pri pohľade zo západu a z juhu. Navrhnite postup, ako by sme zadanú úlohu mohli riešiť naraz s viacerými používateľmi.

Ukážka tabuľky:

1	2	4	4	3	1
0	0	0	0	2	2
0	8	0	9	0	1
1	5	0	0	0	0
2	0	2	0	1	3
1	2	3	0	0	0
0	0	0	5	2	0
0	1	3	3	4	0

Ukážka možnej vizualizácie pri pohľade zo západu:



Poznámky k úlohe:
Analyzovaním vstupných informácií a ukážky výstupu je potrebné odhaliť, že hľadáme maximálne čísla v riadkoch a v stĺpcoch. Ich hodnoty graficky znázorníme pomocou grafu. Na riešenie môžeme využiť tabuľkový kalkulátor, cloudové nástroje alebo aj nejaký online nástroj.

V úlohe treba objaviť, navrhnúť a zrealizovať postup, ktorým získame grafický výstup. Úloha má algoritmický charakter. Prvá časť zasahuje do okruhov Algoritmické riešenie problémov a Reprezentácie a nástroje. V závere máme navrhnúť postup, ako by sme zadanú úlohu mohli riešiť naraz s viacerými používateľmi. Touto časťou zasahujeme do okruhu Komunikácia a spolupráca.

2.3 V tabuľke chyba_v_datach.xlsx sú zakódované nejaké neznáme informácie v dvojkovej sústave. Pri prenose informácií sa používajú v kódovaní aj nadbytočné informácie, ktoré pomáhajú zistiť isté množstvo chýb pri prenose, prípadne ich opraviť.

Doplňte do tabuľky k jednotlivým riadkom a stĺpcom tzv. paritný bit. Paritný bit v riadku bude obsahovať nulu alebo jednotku doplnenú tak, aby počet jednotiek v riadku bol párny. Paritný bit pre stĺpec bude nastavený podľa počtu jednotiek v danom stĺpci. Doplňte do tabuľky aj paritný bit pre diagonálu zľava doprava.

1	0	1	1	0	1	0
1	1	0	1	1	1	0
1	1	1	1	1	1	1
1	1	1	0	0	1	0
0	1	0	1	0	0	0
1	1	0	1	0	1	1
1	0	1	1	0	1	0

Vysvetlite, kde v počítači a ako sa vykonávajú operácie, ktoré ste použili vo svojom riešení.

Poznámky k úlohe:
Na riešenie úlohy nie je potrebné vopred vedieť, čo je paritný bit, krátko sme ho v zadaní vysvetlili.
V úlohe treba porozumieť definícii paritného bitu, navrhnúť a zrealizovať postup, ktorým získame výstup. Na riešenie môžeme využiť tabuľkový kalkulátor, v ktorom sčítame príslušné čísla a vypočítame zvyšok po delení dvomi (napr.: =MOD(SUM(A1:G1);2)).
Úloha má algoritmický charakter. Prvá časť zasahuje do okruhov Algoritmické riešenie problémov a Reprezentácie a nástroje. V závere sa pýtame na princíp fungovania výpočtov v počítači. Touto časťou zasahujeme do okruhu Softvér a hardvér.

2.4 V tabuľke chyba_v_datach.xlsx sú zakódované nejaké neznáme informácie v dvojkovej sústave. Pri prenose informácií sa používajú v kódovaní aj nadbytočné informácie, ktoré pomáhajú zistiť isté množstvo chýb pri prenose, prípadne ich opraviť.

V tabuľke sú k jednotlivým riadkom a stĺpcom doplnené aj tzv. paritné bity (8. stĺpec a 8. riadok). Paritný bit v riadku obsahuje nulu alebo jednotku doplnenú tak, aby počet jednotiek v riadku bol párny. Paritný bit pre stĺpec je nastavený podľa počtu jednotiek v danom stĺpci. V tabuľke je aj paritný bit pre diagonálu zľava doprava.

1	0	1	1	0	1	0	0
1	1	0	1	1	1	0	1
1	1	1	1	0	1	1	1
1	1	1	0	0	1	0	0
0	1	0	1	0	0	0	0
1	1	0	1	0	1	1	1
1	0	1	1	0	1	0	0
0	1	0	0	0	0	0	0

Navrhnite a vytvorte riešenie, ktoré v takejto tabuľke identifikuje chybu (maximálne jednu).

Diskutujte o vplyvoch samoopravných kódov na spoločnosť a bezpečnosť komunikácie.

Poznámky k úlohe:
Na riešenie úlohy nie je potrebné vopred vedieť, čo je paritný bit, krátko sme ho v zadaní vysvetlili.
V úlohe treba porozumieť definícii paritného bitu, navrhnúť a zrealizovať postup, ktorým identifikujeme chybu v danej tabuľke. Na riešenie môžeme využiť tabuľkový kalkulátor, v ktorom sčítame príslušné čísla a vypočítame zvyšok po delení dvomi (napr.: =MOD(SUM(A1:G1);2)). Takto vypočítame nové hodnoty paritného bitu a porovnáme ich s pôvodnými, čím odhalíme miesto chyby.
Úloha má algoritmický charakter. Prvá časť zasahuje do okruhov Algoritmické riešenie problémov a Reprezentácie a nástroje. Diskusiou o vplyvoch samoopravných kódov na spoločnosť a bezpečnosť komunikácie zasahujeme do okruhu Informačná spoločnosť.

2.5

Text k zadaniu:

Gmail pridal jednoduché blokovanie emailu od konkrétnej osoby

Spoločnosť Google pridala do svojej populárnej emailovej služby Gmail novú funkciu, jednoduché blokovanie emailov od konkrétnych osôb. Užívateľ môže v novej verzii jednoducho vo voľbách pre daný email označiť odosielateľa otvoreného emailu za blokovaného, po čom už nebude dostávať emaily z danej emailovej adresy.

Email bude automaticky končiť v spamovom priečinku užívateľa, kde si ho tento môže v prípade potreby nájsť. Správy sa zo spamu automaticky odstraňujú 30 dní po ich prijatí.

Nová funkcia je už prítomná vo webovom rozhraní, v Android aplikácii pre Gmail sa objaví v priebehu týždňa. Zablokovaných užívateľov je možné opäť odblokovať v nastaveniach vo webovom rozhraní, v sekcii filtrov.

Podobnú funkčnosť si bolo možné samozrejme zabezpečiť aj doteraz, ale pomerne komplikovane práve manuálnym vytvorením filtra.

(zdroj: http://www.dsl.sk/article.php?article=17499)

- Navrhnite a slovne popíšte postup, ktorým by ste riešili proces (novú funkciu) spomínaný v texte. Ktoré kroky a v akom poradí by mala funkcia vykonať, ktoré informácie si potrebuje na to pamätať?

- Kde sa vytvorená funkcia vykonáva? Uveďte aj potrebné hardverové zdroje a miesto vykonávania v rámci celej komunikácie pri odosielaní e-mailu zo strany odosielateľa až k adresátovi.
- Predstavme si, že článok bol zverejnený práve teraz, pár hodín po spustení novej funkcie. Ako a ktoré vlastnosti spomínanej funkcionality môžeme ihneď využívať vo svojom smartfóne?

Poznámky k úlohe:
V texte je popis novej funkcionality Gmailu. Úlohou je analyzovať danú funkcionalitu, odhadnúť jej možné spôsoby riešenia, popísať kroky, ktoré musí funkcia vykonať, určiť ich poradie, odhadnúť potrebnú reprezentáciu dát a pod..
Riešením úlohy nie je praktická realizácia nejakého postupu, ale návrh a slovný popis postupu. Úloha má algoritmický charakter. Zasahuje do okruhov Algoritmické riešenie problémov, Reprezentácie a nástroje, Komunikácia a spolupráca, Softvér a hardvér. V tejto úlohe je potrebné prepojiť poznatky a súvislosti z viacerých oblastí.

2.6 V súbore termin.txt (termin.xlsx) je text termínu stretnutia. Zakódujte tieto znaky do dvojkovej sústavy v rovnakom súbore. Vysvetlite princíp prevodu textu na číslo v desiatkovej sústave a prevodu čísla z desiatkovej sústavy do dvojkovej sústavy. Popíšte, akým spôsobom sú reprezentované v počítači znaky.

Vysvetlite, ako a kde je v počítači uložený tento vstupný a výstupný súbor. Akým spôsobom môžeme ochrániť tieto súbory pred inými používateľmi na lokálnom počítači a pri ich presune v počítačovej sieti?

Ukážka súboru:

n
e
d
e
l
a
o
1
9
:
5
0

Poznámky k úlohe:
Úloha je podobná úlohe 2.1, len je tu opačne otočený požadovaný proces (vstup / výstup). V úlohe treba navrhnúť a zrealizovať postup, ktorým získame z textu čísla v binárnej sústave, ktoré reprezentujú kód znakov.
Úlohu môžeme riešiť viacerými spôsobmi:
a) Pomocou tabuľkového kalkulátora. Najprv pre znaky pomocou funkcie code() zistíme kód v desiatkovej sústave a následne ho funkciou dec2bin() prevedieme do dvojkovej sústavy.
b) Nájdeme nejaký online konvertor a pomocou neho získame výstupné čísla.
c) Postup prevodu textu na čísla môžeme popísať a zrealizovať aj „ručne“.

Touto časťou úlohy sme splnili, že má algoritmický charakter, zároveň zasahuje do okruhov Algoritmické riešenie problémov, Reprezentácie a nástroje.
Zvyšnými otázkami v úlohe zasahujeme do okruhov Softvér a hardvér a Komunikácia a spolupráca.

2.7 Adam vytvoril jednoduchú web stránku (stranka.html) s tabuľkou so zoznamom ľudí.

1	Ľubomír Andrejčík
2	Jakub Adamjak
3	Adam Andrejčík
4	Martin Androvic
5	Jakub Almášy
6	Jakub Arvay
7	Matúš Babinský
8	Ivana Adamickova
9	František Amrich
10	Andrej Antoška
11	Dávid Aftanas
12	Michal Adamík
13	Alex Antol
14	Paul Arce
15	Peter Babinsky
16	Martin Bača
17	Adrian Babis

```
<table border="1">
<tr><td>1</td><td>Ľubomír Andrejčík</td></tr>
<tr><td>2</td><td>Jakub Adamjak</td></tr>
<tr><td>3</td><td>Adam Andrejčík</td></tr>
<tr><td>4</td><td>Martin Androvic</td></tr>
<tr><td>5</td><td>Jakub Almášy</td></tr>
<tr><td>6</td><td>Jakub Arvay</td></tr>
<tr><td>7</td><td>Matúš Babinský</td></tr>
<tr><td>8</td><td>Ivana Adamickova</td></tr>
<tr><td>9</td><td>František Amrich</td></tr>
<tr><td>10</td><td>Andrej Antoška</td></tr>
<tr><td>11</td><td>Dávid Aftanas</td></tr>
<tr><td>12</td><td>Michal Adamík</td></tr>
<tr><td>13</td><td>Alex Antol</td></tr>
<tr><td>14</td><td>Paul Arce</td></tr>
<tr><td>15</td><td>Peter Babinsky</td></tr>
<tr><td>16</td><td>Martin Bača</td></tr>
<tr><td>17</td><td>Adrian Babis</td></tr>
</table>
```

Až keď mal stránku hotovú, zistil, že nie je zoradená podľa priezviska a priezvisko nie je v ďalšom stĺpci, ako pôvodne chcel. Navrhnite a zrealizujte tieto úpravy tak, aby mu zabrali čo najmenej času aj v prípade, že zoznam má niekoľko sto ľudí. Popíšte akými spôsobmi môže túto stránku Adam zdieľať na internete a aký majú vplyv na ochranu súkromia.

Poznámky k úlohe:
V úlohe treba zoradiť zoznam ľudí podľa priezviska. Na to potrebujeme transformovať dáta do podoby, v ktorej sa to dá urobiť, zrealizovať zoradenia a následne transformovať dáta do pôvodnej podoby. Úlohu, resp. jednotlivé časti celého postupu je možné riešiť viacerými spôsobmi. Na odstránenie html tagov sa dajú využiť nástroje hľadať a nahradiť (či už v textovom editore alebo v tabuľkovom procesore). Vstupné dáta nám môže tabuľkový kalkulátor otvoriť aj rovno ako tabuľku. S využitím funkcií na prácu s textovými reťazcami v tabuľkovom kalkulátore rozdelíme meno a priezvisko do viacerých stĺpcov (prípadne tam môže byť priamo nástroj na rozdelenie stĺpcov), tabuľku správne zoradíme. Požadovaný výstup html tagy a údaje si môžeme vytvoriť v tabuľkovom kalkulátore spájaním reťazcov.
Úloha má algoritmický charakter. Prvá časť zasahuje do okruhov Algoritmické riešenie problémov a Reprezentácie a nástroje. Otázkou k zdieľaniu na internete a jeho vplyvu na ochranu súkromia zasahujeme do okruhov Komunikácia a spolupráca a Informačná spoločnosť. Úloha zasahuje do 4 okruhov.

2.8 Vysvetlite princíp digitalizácie zvuku. Zistite parametre kvality zvuku nejakej zvukovej nahrávky a vypočítajte jej predpokladanú veľkosť. Vypočítanú veľkosť porovnajte so skutočnosťou a uveďte dôvody odlišnosti. Porovnajte aj veľkosť súboru s veľkosťou, ktorú súbor zaberá na disku. Za akých okolností sa budú údaje líšiť a čo ich ovplyvňuje?

Navrhnite postup, ako by sme otočili zvukovú nahrávku (pripravili na prehratie odzadu). Predpokladajme, že program, ktorý by realizoval tento postup, by mohol pracovať s jednotlivými bajtami súboru (nahrávky). Aké informácie o súbore by sme potrebovali vedieť na realizáciu takéhoto postupu?

Poznámky k úlohe:
Prvá časť úlohy zasahuje do okruhov Reprezentácie a nástroje (zaoberá sa kódovaním zvuku), Softvér a hardvér (získavanie informácií o priečinkoch, súboroch a zariadeniach).

Druhá časť úlohy má algoritmický charakter. Na navrhnutie postupu nie je potrebné poznať presnú štruktúru daného zvukového súboru. Pri analyzovaní úlohy by sa mal žiak zamýšľať nad tým, akú by mohol mať súbor štruktúru. Napríklad môže odhadovať, že je tam nejaká hlavička súboru, môže uvažovať, v akom poradí sú asi uložené informácie popisujúce zvuk a čo to bude znamenať pre otočenie ich poradia. Riešením je slovný návrh postupu, ktorý by nás doviedol k výstupu, a tiež uvažovanie a popis možných vstupov a ich vplyvu na požadovanú transformáciu. Úloha zasahuje do troch okruhov.

2.9 Zistite parametre kvality zvuku nejakej zvukovej nahrávky a vypočítajte jej predpokladanú veľkosť. Vypočítanú veľkosť porovnajte so skutočnosťou a uveďte dôvody odlišnosti. Navrhnite nejaký postup, ktorý by viedol k zmenšeniu veľkosti tohto zvukového súboru za predpokladu, že vieme pracovať s jednotlivými bajtami súboru. Ako takéto zmenšenie nazývame a aké má mať vlastnosti, ak z výstupu chceme získať pôvodné údaje?

Aké informácie o súbore by sme potrebovali vedieť na realizáciu takéhoto postupu?

Zdôvodnite, prečo je možné na počítači súčasne počúvať hudbu a napr. tvoriť dokument v textovom editore. Vysvetlite, aký vplyv na to má použitie viacjadrového procesora?

Poznámky k úlohe:
Prvá časť úlohy zasahuje do okruhov Reprezentácie a nástroje (zaoberá sa kódovaním zvuku), Softvér a hardvér (získavanie informácií o priečinkoch, súboroch a zariadeniach).

Druhá časť úlohy má algoritmický charakter. Na navrhnutie postupu nie je potrebné poznať presnú štruktúru daného zvukového súboru. Pri analyzovaní úlohy by sa mal žiak zamýšľať nad tým, akú by mohol mať súbor štruktúru. Napríklad môže odhadovať, že je tam nejaká hlavička súboru, môže uvažovať, v akom poradí sú asi uložené informácie popisujúce zvuk. Riešením je slovný návrh postupu, ktorý by nás doviedol k výstupu, a tiež uvažovanie a popis možných vstupov a ich vplyvu na požadovanú transformáciu. Žiak môže navrhovať napríklad zníženie kvality zvuku zmenšením počtu vzoriek a znížením vzorkovacej frekvencie, ale aj nejaký spôsob kompresie. Poslednou otázkou v úlohe opäť zasahujeme do okruhu Softvér a hardvér. Úloha celkovo zasahuje do troch okruhov.

2.10 Zistite, aký operačný systém je na počítači, na ktorom pracujete. Vysvetlite, na čo nám slúži operačný systém a vybrané funkcie prakticky prezentujte.

Vysvetlite, v akom vzťahu je hardvér a výber operačného systému vzhľadom na použitý hardvér.

Nájdite na internete nejakú distribúciu Linuxu, ktorú je možné nainštalovať na váš počítač. Akým spôsobom si môžeme overiť pravosť zdrojového kódu tohto operačného systému?

Navrhnite postupnosť krokov a potrebné vstupné informácie na vytvorenie aplikácie, ktorá bude slúžiť na automatizovanie sťahovania požadovanej distribúcie Linuxu a overenie jej pravosti.

Poznámky k úlohe:
Úloha je ukážkou zadania, ktoré mohlo byť zadaním v minulosti. Je doplnená tak, aby mala algoritmický charakter. V úlohe treba navrhnúť zautomatizovanie istého procesu a popísať postup jednotlivých krokov. Úloha zasahuje do okruhov Softvér a hardvér, Algoritmické riešenie problémov a Reprezentácie a nástroje.

2.11 Zistite, aký antivírový program je nainštalovaný na počítači, ktorý používate, či je aktívny a či je aktuálny. Vysvetlite, čo je vírus a ako sa môžeme pred ním chrániť.

Chcem sa pripojiť na webový server antivírovej firmy a chcem si odtiaľ stiahnuť aktualizáciu programu. Ktoré aplikačné protokoly mi to umožnia? Porovnajte ich z hľadiska bezpečnosti. Vysvetlite, ako funguje komunikácia web klienta a web servera.

Poznámky k úlohe:
Úloha nemá algoritmický charakter. Podľa CP má mať väčšina úloh algoritmický charakter. Úloha zasahuje do okruhov Informačná spoločnosť, Softvér a hardvér, Komunikácia a spolupráca. Zrejme by sa dalo polemizovať, či úloha spĺňa CP, alebo je nad rámec CP. Z popisu okruhov Softvér a hardvér, Komunikácia a spolupráca, Informačná spoločnosť nie je presne zrejmá hĺbka požadovaných vedomostí.

2.12 Zistite, či daný počítač je pripojený do počítačovej siete. Popíšte výhody a nevýhody pripojenia do počítačovej siete. Vysvetlite pojem IP adresa, jej typy a spôsoby pridelenia. Vysvetlite, aký je rozdiel v požiadavkách na kvalitu siete a typ spojenia pri prehliadaní videa a pri prenose textových dokumentov.

Nájdite na internete prehľad záťaže liniek slovenského peeringového centra a u vybraného providera analyzujte graf ročnej záťaže. V ktorých mesiacoch je najnižšia prevádzka? Čo pravdepodobne ovplyvňuje túto prevádzku? Diskutujte, ako túto záťaž môžu ovplyvňovať bezpečnostné problémy.

Poznámky k úlohe:
Úloha nemá algoritmický charakter. Podľa CP má mať väčšina úloh algoritmický charakter. Úloha zasahuje do okruhov Komunikácia a spolupráca, Reprezentácie a nástroje, Informačná spoločnosť. Opäť by sa dalo polemizovať, či úloha spĺňa CP, alebo je nad rámec CP. Z popisu okruhov Softvér a hardvér, Komunikácia a spolupráca, Informačná spoločnosť nie je presne zrejmá hĺbka požadovaných vedomostí.

Zdroje:

[1] Cieľové požiadavky na vedomosti a zručnosti maturantov z informatiky, Bratislava, 2016, http://www.statpedu.sk/files/articles/nove_dokumenty/cielove-poziadavky-pre-mat-skusky/informatika.pdf, dostupné jún 2018 (www.statpedu.sk)

[2] Cieľové požiadavky na vedomosti a zručnosti maturantov z informatiky, Bratislava, 2012, http://www.statpedu.sk/files/articles/dokumenty/cielove-poziadavky-na-maturitne-skusky/cp_informatika_2013_2014.pdf, dostupné jún 2018 (www.statpedu.sk)

[3] Cieľové požiadavky na vedomosti a zručnosti maturantov z informatiky, Bratislava, 2010, http://www.statpedu.sk/files/articles/dokumenty/cielove-poziadavky-na-maturitne-skusky/informatika_cp.pdf, dostupné jún 2018 (www.statpedu.sk)

[4] Kučera, P., Výbošťok, J.: Maturujeme v Pythone (zbierka riešených úloh k maturite z informatiky). Bratislava: Peter Kučera, 2018, ISBN 978-80-972987-2-2 (www.programujemevpythone.sk)