Zlepšenie matematických schopností študenta pomocou logických počítačových hier*

Filip Chromek
Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xchromek@stuba.sk

11. október 2022

Abstrakt

Iba málo študentov v dnešnej dobe má rado matematiku. Na druhej strane má veľa mladých ľudí rado hranie hier. Či už hier na počítači alebo na mobilnom telefóne.

Počítačové hry zväčša využívajú prirodzenú zvedavosť a súťaživosť ľudskej mysle, aby presvedčili človeka, aby sa im venoval. Hry taktiež využívajú zvýšené vylučovanie endorfínov a dopamínov počas hrania, čo vedie k ešte väčšiemu záujmu hrať.

Tento fakt je vhodné využiť pre niečo dobré. Pri hraní hier, ktorým obsahom sú informácie z reálneho sveta, si daný hráč dokáže podvedome tieto informácie zapamätať. V hrách musí tieto informácie využívať pre dosiahnutie progresu, musí sa nad nimi zamýšľať a vďaka tomu sa mu „uložia“ v pamäti, aj keď sa ich pôvodne učiť nechcel. Takýto štýl učenia je veľmi účinný najmä pri vedomostiach z oblasti matematiky, keďže sa ich väčšinou nedá „naučiť naspamäť“, ale je potrebné sa nad nimi zamyslieť.

1 Úvod

Motivujte čitateľa a vysvetlite, o čom píšete. Úvod sa väčšinou nedelí na časti.

Uveďte explicitne štruktúru článku. Tu je nejaký príklad. Základný problém, ktorý bol naznačený v úvode, je podrobnejšie vysvetlený v časti 2. Dôležité súvislosti sú uvedené v častiach 4 a 5. Záverečné poznámky prináša časť 6.

2 Nejaká časť

Z obr. 1 je všetko jasné.

*Semestrálny projekt v predmete Metódy inžinierskej práce, ak. rok 2022/23, vedenie: Ing. Vladimír Mlynarovič, PhD.

3 Iná časť

Základným problémom je teda... Najprv sa pozrieme na nejaké vysvetlenie (časť 3.1), a potom na ešte nejaké (časť 3.1).[1]

Môže sa zdať, že problém vlastne nejestvuje [1], ale bolo dokázané, že to tak nie je [2,3]. Napriek tomu, aj dnes na webe narazíme na všelijaké pochybné názory [4]. Dôležité veci možno *zdôrazniť kurzívou*.

3.1 Nejaké vysvetlenie

Niekedy treba uviesť zoznam:

- jedna vec
- druhá vec
  - x
  - y

Ten istý zoznam, len číslovaný:

1. jedna vec
2. druhá vec
   (a) x
   (b) y

3.2 Ešte nejaké vysvetlenie

Veľmi dôležitá poznámka. Niekedy je potrebné nadpisom označiť odsek. Text pokračuje hneď za nadpisom.

4 Dôležitá časť

5 Ešte dôležitejšia časť

6 Záver

[1] Niekedy môžete potrebovať aj poznámku pod čiarou.

Úloha 4.1: Napíšte program, ktorý zo štandardného vstupu načíta dve celé čísla a vypíše všetky prvočísla z daného intervalu. V programe použite funkciu **zisti_prvocislo(int cislo)**, ktorej parametrom je zisťované číslo a návratovou hodnotou je číslo 1 ak zisťované číslo je prvočíslo, inak -1. Program načíta zo vstupu 2 čísla oddelené jednou medzerou a ukončené znakom konca riadku (hranice intervalu). Výstupom programu budú prvočísla z daného intervalu, každé v samostatnom riadku. Ošetrite hranice intervalu. V prípade, ak v zadanom intervale neexistuje prvočíslo program vypíše správu Prvocislo neexistuje. Každý riadok je ukončený znakom konca riadku.

Ukážka vstupu:
1 4
Výstup pre ukážkový vstup:
2
3

Úloha 4.2: Napíšte program, ktorý v prvom riadku načíta celé číslo n predstavujúce počet vstupov. Potom postupne zo vstupu prečíta riadok obsahujúci dvojicu reálnych čísel: hodinovú mzdu a počet odpracovaných hodín za týždeň. Dvojica reálnych čísel je na vstupe oddelená vždy jednou medzerou a ukončená znakom konca riadku. Pre každú dvojicu čísel program zavolá funkciu **double tyzdenna_mzda(double h_mzda, double hod)**, ktorá vráti mzdu za týždeň. Pre každú z n dvojíc vypíše program jeden riadok ukončený znakom konca riadku a obsahujúci správu Hod. mzda: m Euro/hod, hodin: h, spolu: s Euro, kde m je hodinová mzda, h je počet odpracovaných hodín a s je celková mzda za týždeň.

Ukážka vstupu: 3
Ukážka vstupu: 20 38
Ukážka výstupu: Hod. mzda: 20.00 Euro/hod, hodin: 38.00, spolu: 760.00 Euro
Ukážka vstupu: 22 48
Ukážka vstupu: Hod. mzda: 22.00 Euro/hod, hodin: 48.00, spolu: 1144.00 Euro
Ukážka vstupu: 24 68
Ukážka vstupu: Hod. mzda: 24.00 Euro/hod, hodin: 68.00, spolu: 2064.00 Euro
Celkova mzda: 3968.00 Euro

Úloha 4.3: Napíšte program, na zisťovanie reverzného čísla. Program načíta zo vstupu číslo x ukončené znakom konca riadku. V programe použite funkciu **long reverzne_cislo(long x)** ktorá vráti reverzné číslo k číslu x. Výstupom programu bude vrátené reverzné číslo.
Program rozšírte tak, že bude načítať všetky čísla zo vstupu. Počet čísel na vstupe nie je známy pred spustením programu. Využite návratovú hodnotu funkcie scanf. Ku každému načítanému číslu vytvorí reverzné číslo a navyše zistí, či načítané číslo je palindróm a vypíše správu: Cislo X je palindrom, alebo Cislo X nie je palindrom, kde X je zisťované číslo. Správa je nasledovaná znakom konca riadku.

Ukážka vstupu: 12345
Výstup pre ukážkový vstup: 54321
Cislo 12345 nie je palindrom
Ukážka vstupu: 12321
Výstup pre ukážkový vstup: 12321
Cislo 12321 je palindrom

Úloha 4.4: Napíšte program, ktorý rekurzívnou funkciou vypočíta súčet nepárnych kladných celých čísel menších ako N. Na vstupe je dané kladné celé číslo N, na výstup program vypíše jedno číslo: súčet všetkých nepárnych kladných celých čísel menších ako N.

Ukážka vstupu:
10
Výstup pre ukážkový vstup:
25

Úloha 4.5: Napíšte program, ktorý rekurzívnou funkciou vypočíta ciferný súčet čísla. Na vstupe je dané kladné celé číslo N, na výstup program vypíše jedno číslo: ciferný súčet čísla N.

Ukážka vstupu:
56
Výstup pre ukážkový vstup:
11

Úloha 4.6: Napíšte program, ktorý zo štandardného vstupu načíta reálne číslo x. Do súboru nasobky.txt zapíše 1, 2, ..., 10-násobky čísla x. Súbor má obsahovať 10 riadkov s nasledujúcim formátovaním: v i-tom riadku vypíšte i * x = ix, kde i je číslo riadku na 2 miesta, x je načítané číslo vypísané na 2 desatinné miesta a ix je i-ty násobok čísla x tiež vypísaný na 2 desatinné miesta. Každý riadok je ukončený znakom konca riadku.

Ukážka vstupu:
2.5
Súbor nasobky.txt pre ukážkový vstup:
1 * 2.50 = 2.50
2 * 2.50 = 5.00
3 * 2.50 = 7.50
4 * 2.50 = 10.00
5 * 2.50 = 12.50
6 * 2.50 = 15.00
7 * 2.50 = 17.50
8 * 2.50 = 20.00
9 * 2.50 = 22.50
10 * 2.50 = 25.00

Úloha 4.7: Napíšte program, ktorý z textového súboru cisla.txt postupne načíta reálne čísla, vypíše ich na obrazovku a vypočíta ich sučet. V prípade neotvorenia txt súboru program vypíše vetu “Neexistuje subor: cisla.txt.” a program skončí.

Ukážka súboru cisla.txt:
1.25
0.26
1.36
4.52
Ukážkový výstup:
1.25
0.26
1.36
4.52
Sucet cisel je: 7,39.

Úloha 4.8: Napíšte program, ktorý zo štandardného vstupu (klávesnice) načíta znak. Ďalej číta znaky zo súboru znak.txt. Ak program prečítal zo štandardného vstupu (klávesnice) 's', vypisuje načítané znaky do súboru novy.txt. Ak načítal ľubovoľný iný znak, vypisuje načítané znaky na štandardný výstup (obrazovku). Súbor novy.txt alebo štandardný výstup bude teda obsahovať presnú kópiu obsahu súboru znak.txt.

Ukážka vstupu:
s
Ukážka súboru znak.txt:
abrakadabra
bubu
Súbor novy.txt pre ukážkový vstup:
abrakadabra
bubu

Úloha 4.9: Napíšte program, ktorý bude čítať znaky zo súboru text.txt pokiaľ nenačíta znak '*'. Ak načíta znak 'x' alebo 'X' vypíše Precital som X, ak znak 'y' alebo 'Y' vypíše Precital som Y, ak načíta znaky '#', '$' alebo '&' vypíše Precital som riadiaci znak a ak načíta znak '*' vypíše Koniec a skončí čítanie súboru. Po prečítaní súboru vypíše správu Pocet precitanych medzier: nasledovanú medzerou a počtom prečítaných medzier. Každá správa je nasledovaná koncom riadku.

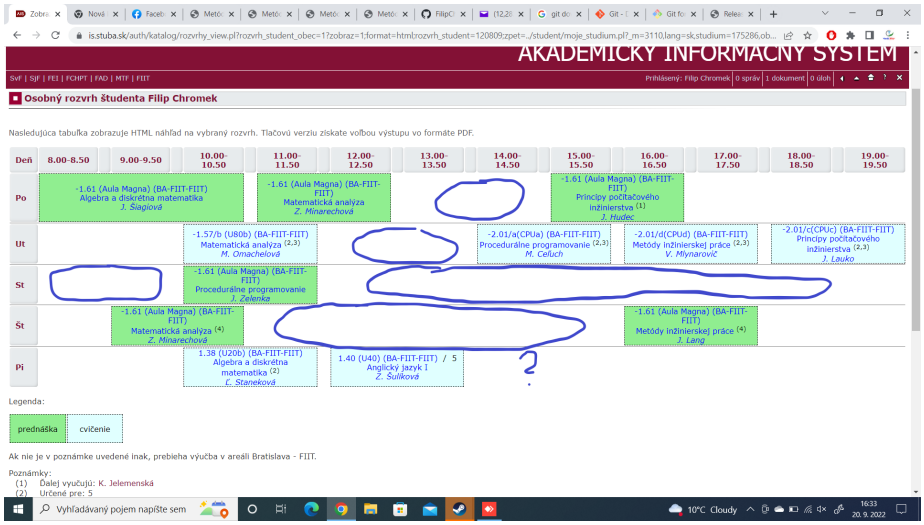
Ukážka súboru text.txt:
$ abc 5 xyz #
& Q *# abf
Ukážkový výstup pre súbor text.txt:
Precital som riadiaci znak
Precital som X
Precital som Y
Precital som riadiaci znak
Precital som riadiaci znak
Koniec
Pocet precitanych medzier: 6

Úloha 4.10: Napíšte program, ktorý určí, či majú dva súbory prvy.txt a druhy.txt rovnaký obsah. Program nečíta žiadne dáta zo štandardného vstupu. Ak majú súbory rovnaký obsah, program vypíše Subory su identicke Ak súbory rovnaký obsah nemajú, vypíše program Pocet roznych znakov: nasledovaný medzerou, počtom rôznych znakov v súboroch a ukončený koncom riadku. i-ty znak v jednom súbore považujte za rôzny od i-teho znaku v druhom súbore, ak oba znaky existujú (t.j. ani jeden súbor nemá menej ako i znakov) a príslušné znaky sa nerovnajú. Ak majú súbory nerovnakú dĺžku, na výstup program vypíše ešte jeden riadok obsahujúci správu Jeden zo suborov je dlhsi o x znakov Pričom x je počet znakov o ktoré je jeden zo súborov dlhší. Správa je nasledovaná koncom riadku.

Ukážka súboru prvy.txt:
ahoj
Ukážka súboru druhy.txt:
ahujx
*
Výstup pre ukážkové súbory:
Pocet roznych znakov: 1
Jeden zo suborov je dlhsi o 3 znako

Aj text môže byť prezentovaný ako obrázok. Stane sa z neho označný plávajúci objekt. Po vytvorení diagramu zrušte znak % pred príkazom `\includegraphics` označte tento riadok ako komentár (tiež pomocou znaku %).

Obr. 1: Rozhodujúci argument.



Obr. 2: A boat.

Literatúra

[1] James O. Coplien. *Multi-Paradigm Design for C++*. Addison-Wesley, 1999.

[2] Krzysztof Czarnecki, Simon Helsen, and Ulrich Eisenecker. Staged configuration through specialization and multi-level configuration of feature models. *Software Process: Improvement and Practice*, 10:143–169, April/June 2005.

[3] Krzysztof Czarnecki and Chang Hwan Peter Kim. Cardinality-based feature modeling and constraints: A progress report. In *International Workshop on Software Factories, OOPSLA 2005*, San Diego, USA, October 2005.

[4] Carnegie Mellon University Software Engineering Institute. A framework for software product line practice—version 5.0. `http://www.sei.cmu.edu/productlines/frame_report/`.