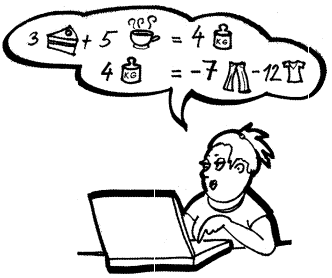
INFORMATIKA

Soustavy lineárních rovnic a počítače

ANTONÍN JANČAŘÍK

Pedagogická fakulta UK, Praha



Úvod

Řešení soustav lineárních rovnic patří mezi úlohy, s nimiž se seznamují žáci již na základních školách. Následně na střední a především vysoké škole je toto téma zasazeno do širšího rámce lineární algebry. Postup, který žáci používali pro řešení jednoduchých soustav: Gaussova, resp. Gauss-Jordanova eliminační metoda, je zobecněn a použit i pro řešení dalších typů úloh: výpočet determinantu, inverzní matice, vlastních vektorů či báze lineárního obalu. Všechny tyto typy úloh patří historicky mezi klasické partie matematiky.

Pokud jde pouze o dosažení správného výsledku, lze soustavu lineárních rovnic zadat a následně vyřešit i za pomoci vhodného výpočetního software, nejlépe Computer Algebra systémů (CAS). Všechny CAS systémy umí úlohy tohoto typu řešit, a to jak numericky, tak i algebraicky (tedy formálně, včetně parametrů), zpravidla za pomoci příkazu *Solve* (více viz např. [1]). CAS programy mají, vzhledem k použití při výuce, i několik nevýhod. Zaprvé robustnost, která přesahuje potřeby žáků většiny nejenom základních, ale i středních škol. Zadruhé to, že tyto programy prezentují výsledné řešení úlohy a v naprosté většině případů (např. Maple 13, Mathematica 7) neumožňují zobrazit postup řešení. Tyto programy lze tedy při výuce použít pouze pro kontrolu výsledku, ale nikoli pro demonstraci postupu řešení (srovnej [2]).

Cílem tohoto článku je představit několik počítačových nástrojů, pomocí nichž lze pro úlohy s použitím Gauss-Jordanovy eliminační metody nejenom nalézt výsledek řešení, ale také zaznamenat a v některých případech i řídit průběh řešení. V článku budou představeny dva on-line dostupné nástroje pro řešení soustav lineárních rovnic a doplněk do programu Excel, který nabízí provádění Gauss-Jordanova eliminační metody krok za krokem.

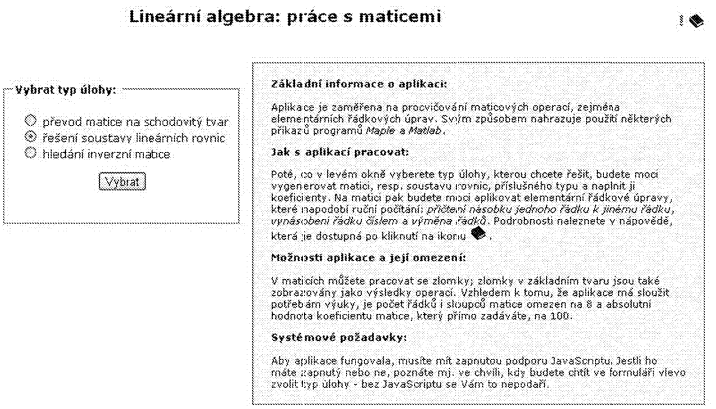
Nástroje na Webu

Obě ukázky nástrojů dostupných na Internetu pocházejí z dílny českých autorů. Jedná se o nástroj zveřejněný na portálu Mathematical Assistant on Web a aplikaci využívanou v rámci e-learningové podpory kurzu lineární algebry na PedF UK v Praze. V obou případech se jedná o nástroje, které jsou uživatelům nabízeny bezplatně. Ani jeden z představovaných nástrojů neřeší soustavy lineárních rovnic automaticky. Oba nástroje však poskytují uživateli, který je s principem Gaussovy eliminace seznámen, prostředí pro pohodlné řešení soustav rovnic bez numerických chyb, s možností vést průběh celého řešení.

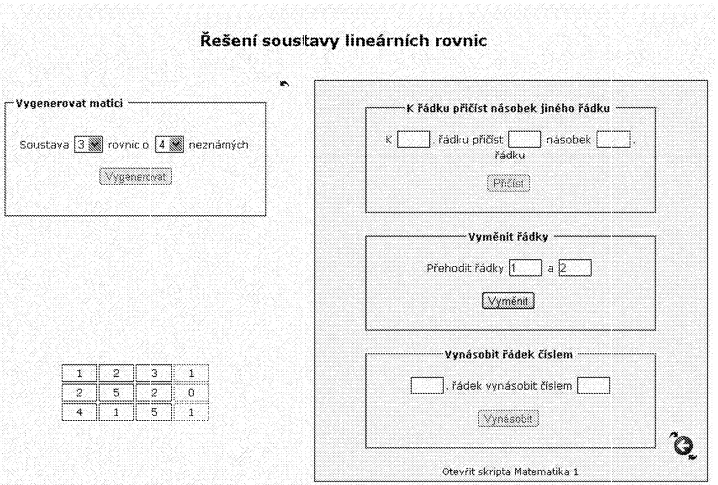
Mathematical Assistant on Web (MAW)

Projekt MAW – Matematické výpočty online (http://user.mendelu.cz/~marik/maw/) autorů Roberta Mařík a Miroslavy Tihlaříkové patří mezi nejkvalitnější projekty, které lze na síti Internet nalézt. Portál, za jehož webovým rozhraním se ukrývá výkonný CAS Maxima, nabízí on-line výpočty, včetně postupu řešení, pro širokou škálu úloh z diferenciálního a integrálního počtu. V současné době jsou na tomto portále nabízeny aplikace Michala Nováka pro práci s maticemi, a to včetně nástroje pro řešení soustav lineárních rovnic.

Tato aplikace, vytvořená v jazyce PHP a JAVA script, umožňuje pracovat s maticemi způsobem obdobným „normálnímu“ řešení. Řešitel manuálně zadává jednotlivé operace, které jsou následně prováděny na zadané matici. Program umožňuje všechny tři základní úkony – vynásobení řádku konstantou, prohození dvou řádků i přičtení k-násobku jednoho řádku k druhému. Vzhledem ke zvolené technologii je při výpočtu nutné nepřetržité připojení k internetu, protože všechny operace jsou prováděny na serveru.



Obr. 1 Lineární algebra na MAW



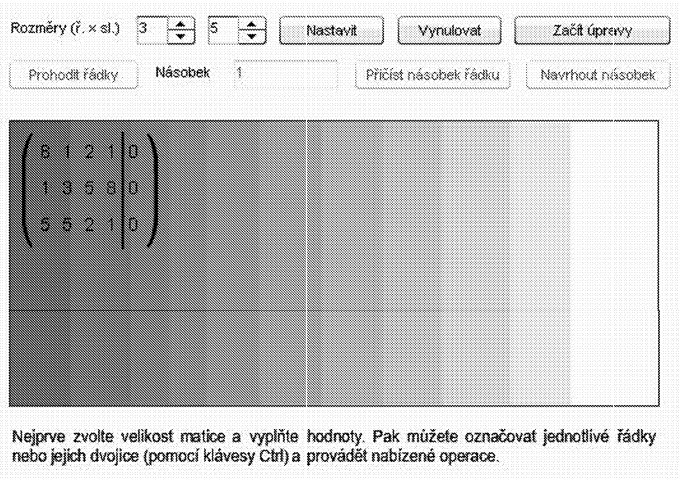
Obr. 2 Řešení soustav lineárních rovnic na MAW

Řešení z kurzu Lineární algebra na PedF UK

Jiné řešení bylo zvoleno autorem v rámci internetového kurzu Lineární algebra (http://moodle.pedf.cuni.cz/, heslo pro hosta Algebra). Autorem aplikace je Martin Adamec. Jako programovací jazyk byl zvolen Adobe

Flash. Tato aplikace, stejně jako PHP aplikace z předchozí ukázky, nabízí základní řádkové úpravy matice. Na rozdíl od předchozí ukázky však poskytuje uživatelům některé výhody:

1. Výsledky výpočtů jsou zobrazovány ve formátu, který je obvyklý v učebnicích lineární algebry (maticový zápis).
2. Při výpočtu aplikace nabízí uživateli koeficienty, vhodné pro další krok v Gaussově eliminaci.
3. Použitá technologie (Adobe Flash) umožňuje stažení aplikace a provádění výpočtů bez potřeby nepřetržitého připojení k Internetu.

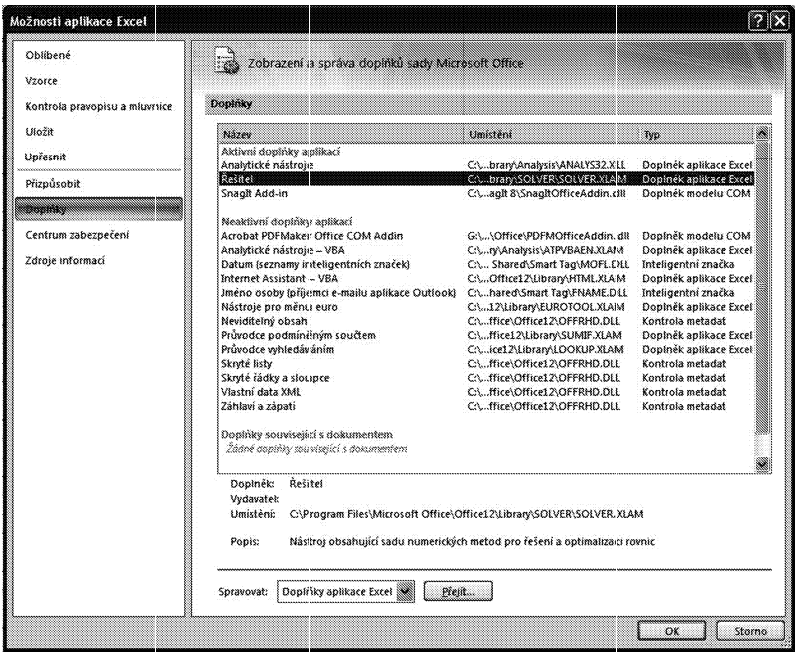


Obr. 3 Řešení soustav lineárních rovnic

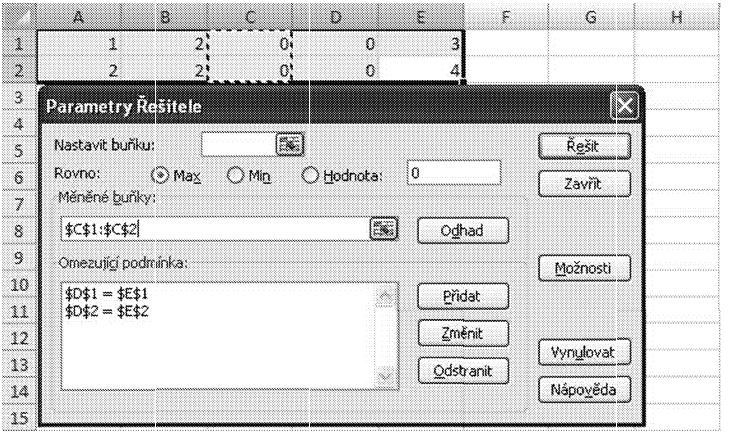
Výrazným vylepšením je především automatický návrh vhodných koeficientů pro řádkové úpravy, díky této funkci je možné soustavy řešit „poloautomaticky“, ale také porovnávat vlastní návrhy se zabudovanou funkcí, a tak se lépe seznámit s algoritmem Gaussovy eliminace.

MS Excel

Pro řešení soustav lineárních rovnic však lze použít i tak obyčejný nástroj, jako je tabulkový procesor – spreadsheet. Tabulkové procesory obvykle nenabízejí možnost zobrazit postup řešení. Existují ale způsoby, jak celý postup řešení soustavy lineárních rovnic získat i za pomoci tabulkových procesorů.



Obr. 4 Doplňky v programu MS Excel



Obr. 5 Použití nástroje Řešitel

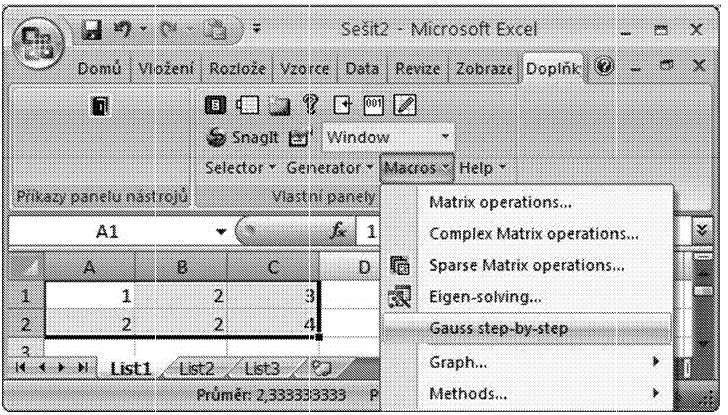
Jako ukázkové prostředí použijeme asi nejrozšířenější tabulkový procesor – MS Excel, a to v jeho poslední verzi 2007 (více o využití programu MS Excel pro řešení náročnějších matematických úloh viz [4]). Pro získání řešení soustavy lineárních rovnic lze použít doplněk Řešitel (více viz [5]). Tímto postupem však nezískáme postup řešení.

Doplněk Matrix and Linear Algebra Functions for Excel

Díky integraci VBA (Visual Basic for Applications) do programu MS Excel lze nyní v této aplikaci naprogramovat nové funkce. Velice často se však setkáváme s tím, že požadované „nové“ funkce jsou již vytvořené. Mnohé důležité funkce jsou již v programu MS Excel předdefinované, nebo je lze doinstalovat jako součást volně dostupných „Add-In“ komponent. Seznam dostupných Add-In komponent lze nalézt například na serveru Mathtools.net [6]. V dalším textu budeme kromě vestavěných funkcí MS Excelu využívat i funkce z komponenty MATRIX 2.3 – Matrix and Linear Algebra Functions for Excel (více viz [7])

Pomocí funkcí z toho volně šiřitelného doplňku programu MS Excel můžeme řešit celé spektrum úloh z lineární algebry v reálném i komplexním oboru.

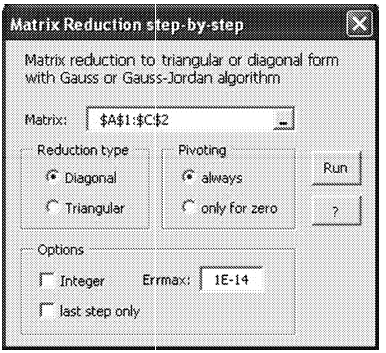
Řešení soustavy pomocí toho doplňku je velice jednoduché a intuitivní. V prvním kroku zadáme koeficienty soustavy rovnic do jednotlivých buněk tabulky.



Obr. 6 Doplněk Matrix 2.3

Následně z nabídky Doplňky vybereme ze submenu Macros doplňku MATRIX 2.3 nabídku Gauss step-by-step.

V tomto doplňku nejprve nastavíme základní parametry pro řešení soustavy a následně spustíme makro klepnutím na tlačítko *Run*. Makro pro-



Obr. 7 Nastavení parametrů makra Gauss step-by-step

9	2	2	4	-0,5
10	1	2	3	1
11				
12	Det(A2) = -1 Det(A)			
13				
14	2	2	4	1
15	0	1	1	-2
16				
17	Det(A3) = -1 Det(A)			
18				
19	2	0	2	
20	0	1	1	
21				1
22	Det(A4) = -1 Det(A)			
23				
24	2	0	2	1/2
25	0	1	1	1
26				
27	Det(A5) = -1/2 Det(A)			
28				
29	1	0	1	
30	0	1	1	

Obr. 8 Řešení soustavy

vede kompletní Gaussovu eliminaci krok za krokem, přičemž v každém kroku na konec řádku uvádí, které řádky matice byly použity a jakým způsobem. Např. v uvedeném případě v prvním kroku přičítá makro $-1/2$ – násobek prvního řádku k druhému řádku a v druhém kroku je od prvního řádku odečten dvojnásobek druhého řádku. Informace o determinantech jsou v tomto případě poněkud matoucí a nesouvisí s řešením soustavy lineárních rovnic. Determinanty poskytují důležitou informaci v případě, kdy používáme makro na výpočet determinantu čtvercové matice. Řádek s determinanty nás informuje, jak souvisí determinant nově získané matice s determinantem matice původní, tedy zda provedená úprava má či nemá vliv na determinant matice.

Závěr

Softwarové nástroje, představené v článku, nám nabízejí mnoho možností, jak nalézat řešení soustavy lineárních rovnic. Počínaje numerickým řešením pomocí doplňku Řešitel v programu MS Excel, přes algebraická řešení v CAS systémech a částečně manuální řešení v on-line nástrojích, až po plně automatizované řešení v programu MS Excel pomocí doplňku Matrix 2.3. Představení těchto nástrojů ve výuce představuje dobré propojení matematiky a výpočetní techniky. Gaussova eliminace je zároveň i algoritmem, který může být použit ve výuce programování, kdy si žáci mohou naprogramovat vlastní makra a doplňky do tabulkového procesoru, který ve své práci používají.

Literatura

[1] *Schneiderová, R.*: Řešení soustav lineárních rovnic s využitím programu Maple (bakalářská práce), 2008, on-line: http://mant.upol.cz/soubory/OdevzdanePrace/B08/b08-42-rs.pdf

[2] *Kutzler, B.*: CAS jako pedagogické prostředky ve vyučování a učení se matematice. Učitel matematiky. 2004, roč. 12, č. 50 a 51.

[3] *Mařík R. – Tihlaříková, M.*: Systém pro online výpočty v prostředí WWW. Sborník 5. ročníku konference o elektronické podpoře výuky SCO 2008, 2008.

[4] *Trávníček, S. – Matyáš, V.*: Derivace v Excelu, MFI, 19/2, 2009.

[5] *Pražák, P.– Pražáková, B.*: Základní použití Řešitele v Excelu, MFI, 18/7, 2009.

[6] Mathtools. net: Link Exchange for the Technical Computing Community, Dostupné online: http://www.mathtools.net/Excel/Mathematics/

[7] *Foxes Team*: Tutorial of Numerical Analysis with Matrix.xla – Vol. 1 and 2, PDF, Dostupné online: http://digilander.libero.it/foxes/Documents.htm#MatrixTutorial

Autorkou úvodní ilustrace je Mgr. Jaroslava Čermáková

Úlohy nového typu na IOI

PAVEL TÖPFER

Matematicko-fyzikální fakulta UK, Praha

(*Dokončení*)

Garáž

Ve velké garáži je N parkovacích míst, která jsou očíslována od 1 do N včetně. Garáž se otvírá každé ráno prázdná a je během dne provozována v následujícím režimu. Kdykoliv do garáže přijede auto, hlídač zkontroluje, zda je některé parkovací místo volné. Pokud není, auto musí počkat u vjezdu do garáže, dokud se nějaké místo neuvolní. Je-li v garáži více volných míst, auto zaparkuje na volném místě s nejmenším číslem. Jestliže do garáže přijedou další auta v době, kdy nějaké auto čeká na uvolnění místa, všechna tato auta se řadí u vjezdu do fronty v tom pořadí, v jakém přijela. Když se pak uvolní v garáži nějaké parkovací místo, zaparkuje na něm první auto z fronty (tzn. to z čekajících aut, které přijelo nejdříve).

Cena za parkování (v dolarech) se počítá jako součin hmotnosti auta (v kilogramech) a koeficientu, jehož hodnota závisí na použitém parkovacím místě. Cena nezávisí na tom, jak dlouho auto zůstane v garáži.

Správce garáže ví, že dnes přijede do garáže celkem M aut, a zná také pořadí jejich příjezdů a odjezdů. Pomozte mu spočítat, kolik dolarů dnes celkově vybere za parkování.

Úloha

Napište program, který na základě znalosti cenových koeficientů jednotlivých parkovacích míst v garáži a údajů o přijíždějících autech (hmotnosti aut, pořadí jejich příjezdů a odjezdů) určí celkovou cenu v dolarech, kterou řidiči parkujících aut zaplatí.

Omezení

$1 \leq N \leq 100$	počet parkovacích míst
$1 \leq M \leq 2\,000$	počet aut
$1 \leq R_s \leq 100$	koeficient parkovacího místa s v dolarech za kilogram
$1 \leq W_k \leq 10\,000$	hmotnost auta k v kilogramech

Vstup

Váš program musí přečíst ze standardního vstupu následující údaje:

- První řádek obsahuje celá čísla N a M oddělená mezerou.
- Dalších N řádků udává koeficienty parkovacích míst. V pořadí s-tý z těchto řádků obsahuje jedno celé číslo R_s, koeficient parkovacího místa číslo s v dolarech za kilogram.
- Následujících M řádků určuje hmotnosti aut. Auta jsou očíslována od 1 do M včetně, a to bez ohledu na pořadí jejich příjezdu do garáže nebo odjezdu. V pořadí k-tý z těchto M řádků obsahuje jedno celé číslo W_k, hmotnost auta číslo k v kilogramech.
- Posledních $2M$ řádků popisuje chronologicky, jak probíhají příjezdy a odjezdy jednotlivých aut. Kladné celé číslo i znamená, že auto číslo i přijíždí do garáže. Záporné celé číslo $-i$ znamená, že auto číslo i odjíždí z garáže. Žádné auto nebude odjíždět z garáže dříve, než přijelo, a číslo každého auta od 1 do M včetně se objeví v této posloupnosti právě dvakrát – jednou při příjezdu a podruhé při odjezdu. Žádné auto neodjede z garáže před zaparkováním (tzn. žádné auto neopustí předčasně frontu, v níž čeká na parkování).

Výstup

Váš program musí zapsat na standardní výstup jeden řádek obsahující jedno celé číslo – celkovou částku v dolarech, kterou dnes utrží správce garáže.

Hodnocení

V testovacích datech, za která lze získat hodnocení 40 bodů, bude mít každé přijíždějící auto k dispozici vždy alespoň jedno volné parkovací místo v garáži. V těchto případech tedy žádné auto nebude muset čekat na uvolnění místa.

Příklady

Vstup	**Výstup**
3 4 2 3 5 200	5300

100 300 800 3 2 −3 1 4 −4 −2 −1	

Auto číslo 3 zaparkuje na místě 1 a zaplatí 300*2 = 600 dolarů.
Auto číslo 2 zaparkuje na místě 2 a zaplatí 100*3 = 300 dolarů.
Auto číslo 1 zaparkuje na místě 1 (které se uvolnilo po odjezdu auta číslo 3) a zaplatí 200*2 = 400 dolarů.
Auto číslo 4 zaparkuje na místě 3 (poslední zbývající místo) a zaplatí 800*5 = 4 000 dolarů.

Vstup	**Výstup**
2 4 5 2 100 500 1000 2000 3 1 2 4 −1 −3 −2 −4	16200

Auto číslo 3 zaparkuje na místě 1 a zaplatí 1 000*5 = 5 000 dolarů.
Auto číslo 1 zaparkuje na místě 2 a zaplatí 100*2 = 200 dolarů.
Auto číslo 2 přijede a musí čekat u vjezdu.
Auto číslo 4 přijede a musí čekat u vjezdu za autem číslo 2.
Když auto číslo 1 opustí své parkovací místo, auto číslo 2 na něm zaparkuje a zaplatí 500*2 = 1 000 dolarů.
Když auto číslo 3 opustí své parkovací místo, auto číslo 4 na něm zaparkuje a zaplatí 2 000*5 = 10 000 dolarů.

⋆⋆⋆

Jednoduché simulační řešení nevyžaduje použití žádné datové struktury kromě dvou jednorozměrných polí pevné velikosti. Jedno z nich je indexováno čísly aut a pro každé auto je v něm uložena hmotnost tohoto auta (vstupní údaj), informace o jeho aktuálním stavu (zda parkuje, čeká ve frontě na zaparkování, nebo zda momentálně není v garáži), na kterém parkovacím místě parkuje (pokud je zrovna zaparkováno) a čas jeho příjezdu do garáže (pokud čeká ve frontě). Druhé pole je indexováno čísly parkovacích míst a obsahuje informaci, jaký cenový koeficient platí pro příslušné parkovací místo (vstupní údaj) a zda je toto parkovací místo momentálně volné, nebo zda je obsazeno nějakým zaparkovaným autem.

Na začátku výpočtu načteme do uvedených polí vstupní údaje (hmotnosti aut, koeficienty parkovacích míst) a nastavíme další počáteční hodnoty obou polí tak, aby odpovídaly výchozímu stavu, že v garáži není žádné auto. Nyní budeme postupně zpracovávat jednotlivé události ze vstupu. Při příjezdu auta projdeme parkovací místa v pořadí podle jejich čísel a hledáme první volné místo. Pokud ho najdeme, přijíždějící auto na něj zaparkujeme a k výsledku přičteme cenu za jeho parkování. V opačném případě musíme nechat auto stát ve frontě a poznamenáme si čas příjezdu. Při odjezdu auta z garáže projdeme seznam všech aut a hledáme auto čekající ve frontě s nejnižším časem příjezdu. Když takové auto najdeme, zaparkujeme ho na právě uvolněné místo (poznamenáme si, že toto auto již parkuje a kde, přičteme k výsledku cenu za jeho parkování). Jestliže žádné auto nečeká ve frontě, parkovací místo uvolněné odjíždějícím autem zůstane prázdné (poznamenáme si, že toto parkovací místo je volné).

Popsané řešení má časovou složitost $O(M^2 + MN)$. Takovéto řešení stačilo v soutěži k zisku plného počtu bodů, je ovšem možné vylepšit ho a urychlit tak výpočet programu. Jestliže pro uložení fronty čekajících aut použijeme samostatné pole, v němž budou zařazena čísla čekajících

aut v pořadí podle svého příjezdu do garáže, najdeme při odjezdu auta z garáže nejdéle čekající auto v konstantním čase, bez nutnosti procházet vždy celé pole aut. Výpočet programu se tím urychlí na čas $O(MN)$.

Vyhledávání volného parkovacího místa s nejmenším číslem můžeme částečně urychlit tím, že si budeme v pomocné proměnné stále pamatovat aktuální nejnižší číslo volného parkovacího místa. Tento údaj lze poměrně snadno aktualizovat (viz ukázkový program). Podstatnějšího zrychlení dosáhneme, jestliže si budeme ukládat čísla volných parkovacích míst do binární haldy. Při příjezdu auta do garáže potom získáme číslo prvního volného parkovacího místa v logaritmickém čase. Výsledný program využívající frontu čekajících aut i binární haldu volných parkovacích míst bude mít časovou složitost pouze $O(M \log N)$.

Na závěr si ukážeme jeden možný způsob naprogramování úlohy o garáži. Pro ukázku jsme zvolili částečně optimalizované řešení s časovou složitostí $O(MN)$, které má krátký a přehledný kód. Zavedení samostatného pole pro uložení fronty čekajících aut umožnilo zjednodušit pole s evidencí stavu jednotlivých aut – pro každé auto bude nyní stačit evidovat si pouze jeho hmotnost a číslo parkovacího místa. Celkové paměťové nároky programu tak oproti původnímu neoptimalizovanému řešení nevzrostly. Plně optimalizované řešení s ukládáním volných parkovacích míst do binární haldy je již pouze technickou modifikací, kterou si můžete zkusit naprogramovat sami.

```
program Garaz;

const MaxN = 100;          {maximální počet parkovacích míst}
      MaxM = 2000;                      {maximální počet aut}

var N: integer;                       {počet parkovacích míst}
    M: integer;                                   {počet aut}
    Cena: array [1..MaxN] of integer;{cena parkovacího místa}
    Volno: array [1..MaxN] of boolean;{volná parkovací místa}
    Hmotnost: array [1..MaxM] of integer;     {hmotnost auta}
    Parkovani: array [1..MaxM] of integer; {kde auto parkuje}
    Fronta: array [1..MaxM-MaxN] of integer;  {čekající auta}
    Prijezd, Odjezd: integer;           {ukazatelé do Fronty}
    Prvni: integer;                       {první volné místo}
    Suma: longint;                   {výsledná celková částka}
```

```
    A: integer;                          {číslo aktuálního auta}
    i, j: integer;

begin
{Inicializace datových struktur:}
read(N, M);                             {počet míst a počet aut}
for i:=1 to N do
  begin
  read(Cena[i]);        {cenové koeficienty parkovacích míst}
  Volno[i] := true                   {všude v garáži je volno}
  end;
for j:=1 to M do
  read(Hmotnost[j]);                             {hmotnosti aut}
Prijezd := 0; Odjezd := 1; {fronta čekajících aut je prázdná}
Prvni := 1;                          {všechna místa jsou volná}
Suma := 0;                              {dosud nikdo neplatil}

{Simulace příjezdu a odjezdu aut:}
for j:=1 to 2*M do
  begin
  read(A);                                        {aktuální auto}
  if A > 0 then                                  {příjezd auta A}
    begin
    if Prvni > N then                {není žádné volné místo}
      begin
      inc(Prijezd);
      Fronta[Prijezd] := A      {auto A bude čekat ve frontě}
      end
    else                               {máme první volné místo}
      begin
      Parkovani[A] := Prvni;             {auto A tam zaparkuje}
      Volno[Prvni] := false;
      Suma := Suma + Hmotnost[A]*Cena[Prvni];  {...a zaplatí}
      while (Prvni <= N) and (not Volno[Prvni]) do
        inc(Prvni);                {najdeme další volné místo}
      end
    end
  else                 {odjezd auta A - uvolní se jeho místo}
```

```
    if Prijezd >= Odjezd then    {nějaké auto čeká ve frontě}
      begin
      Parkovani[Fronta[Odjezd]] := Parkovani[-A];
                                    {auto z fronty zaparkuje}
      Suma := Suma + Hmotnost[Fronta[Odjezd]]*Cena
              [Parkovani[-A]];                {...a zaplatí}
      inc(Odjezd)
      end
    else                               {ve frontě nikdo nečeká}
      begin
      Volno[Parkovani[-A]] := true;   {uvolněné místo zůstane
                                                    prázdné}
      if Parkovani[-A] < Prvni then
        Prvni := Parkovani[-A] {případně opravíme první volné
                                                      místo}
      end
  end;

writeln(Suma)
end.
```

ZPRÁVY

Dílny Heuréky 2010 – neformální konference učitelů fyziky s mezinárodní účastí

Již od roku 2002 se na přelomu září a října schází několik desítek učitelů na Jiráskově gymnáziu v Náchodě na konferenci *Dílny Heuréky*. V letošním roce se tato tradiční akce konala o víkendu 1. – 3. 10. za účasti více než devadesáti učitelů z praxe, posluchačů učitelství fyziky i pracovníků a doktorandů fakult vzdělávajících budoucí učitele fyziky. V tomto počtu je i osm účastníků ze zahraničí (přičemž slovenské účastníky nepočítáme mezi zahraniční).

Projekt Heuréka asi není třeba většině čtenářů představovat, zmíníme pouze odkaz na webové stránky: http://kdf.mff.cuni.cz/heureka. Pro ty, kteří se s ním nikdy nesetkali, snad jen velmi stručně – Heuréka vznikla již téměř před dvaceti lety „zespodu“, když se několik učitelů fyziky rozhodlo, že by chtěli učit tak, aby to žáky bavilo a něco si z výuky odnesli. V tuto chvíli má Heuréka přes sto aktivních členů – převážně učitelů ze škol, ale i pracovníků fakult vzdělávajících budoucí učitele fyziky a dalších zájemců. Členové se mohou každoročně účastnit několika typů seminářů – „školky“ pro nové