Fractions 2.0

$$\frac{\frac{1}{2}+\frac{3}{2}}{\frac{123}{456}\frac{456}{123}} = \underline{\underline{2}}$$

written by Michal Hrušecký

15. září 2004

Obsah

Část I

Uživatelský manuál

Kapitola 1

Slovo úvodem

Program Fractions je určen pro počítání se zlomky. Umožňuje jednoduše zadávat matematické výrazy a počítat výsledky. Umožňuje zapsat několik výrazů do souboru a tento soubor poté nechat spočítat. Je podporován i výstup ve formátu vhodném pro následné zpracování sázecím programem LaTeX. Takto získáte přehledně zpracované výsledky.

Program Fractions je určen především pro nekomereční využití. Z toho plynou i nízké nároky na hardware a software. Naopak již zmiňovaný vstup i výstup do souboru umožňuje využití i při složitějších výpočtech.

Program je navíc vybaven velmi intuitivním a userfriendly ovládáním. Běží v `DOS`ovém okně, což u mnohých zajisté vzbudí nostalgické vzpomínky. A ostatním připomene, jak bylo v dávných dobách ovládání jednoduché a přívětivé.

Doufám, že tento program splní Vaše očekávání a dopomůže Vám k lepším výsledkům

Michal Hrušecký
autor programu

Kapitola 2

Instalace

Zkontrolujte prosím nejdříve, zda váš počítač splňuje minimální požadavky.

2.1 Požadavky

Zde je seznam požadavků. Uváděny jsou minimální požadavky i optimální konfigurace pro běh programu.

	minimální	optimální
Operační systém	MS Windows	`MS-DOS`
Další software	unzip	unzip, LaTeX
Volné místo na pevném disku	500Kbytů	1Mbyte

Tabulka 2.1: minimální požadavky

2.2 Instalace (MS Windows)

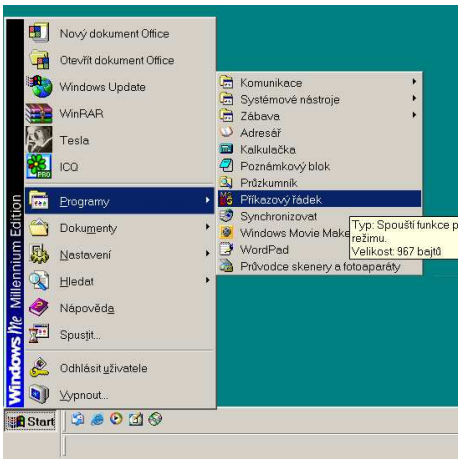
2.2.1 Vyvolání příkazové řádky

Vyvolejte nabídku `START`[1]. Dále vyvolejte podnabídku programy[2]. Pokud se v této nabídce nachází `Prompt MS-DOS` nebo `Příkazový řádek`, klikněte na něj. Pokud ne, vyvolejte podnabídku příslušenství. Pokud se nenachází ani tam, kontaktujte administrátora.

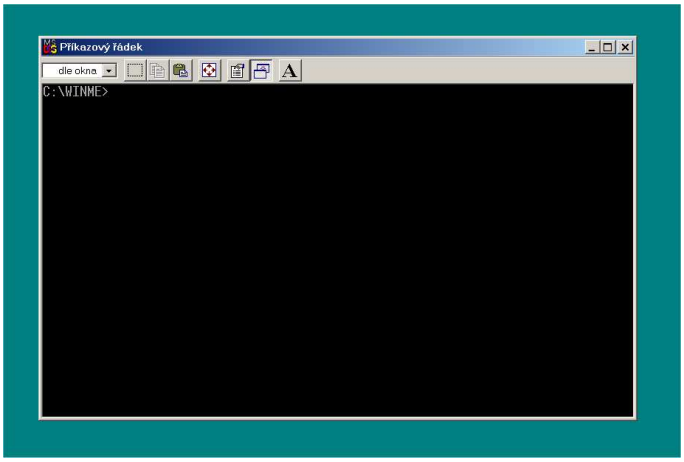
Po vyvolání příkazové řádky by se na obrazovce mělo objevit černé okno příkazového řádku `MS-DOS`. V tomto okně pokračujte podle popisu instalace pro uživatele systému `MS-DOS`.

[1] Tento úkon se obvykle provádí stiskem levého tlačítka myši nad obrázkem v levém dolním rohu obrazovky nebo stiskem speciálního tlačítka na klávesnici

[2] najedete myší nad nápis Programy, který se objevil po vyvolání nabídky Start



Obrázek 2.1: nabídka START



Obrázek 2.2: příkazový řádek `MS-DOS`

2.3 Instalace (MS-DOS)

Vytvořte adresář, v němž chcete mít program nainstalovaný a rozbalte do něj obsah souboru frac.zip. Například, chci-li mít program nainstalovaný v adresáři `C:\bin\frac` a mám-li zip v adresáři `D:\Software` provedu to následujícím způsobem:

```
C:\> cd bin
C:\BIN> mkdir frac
C:\BIN> cd frac
C:\BIN\FRAC> copy d:\Software\frac.zip
     1 zkopírovaných souborů
C:\BIN\FRAC> unzip frac.zip
Archive: frac.zip
     inflating: frac.exe
     inflating: readme.pdf
C:\BIN\FRAC> del frac.zip
```

Obrázek 2.3: Postup instalace

Po úspěšném nainstalování by měl být program připraven k použití. Můžeme tedy přejít k vlastnímu spouštění.

2.4 Spouštění programu (MS Windows)

Vyvoláme příkazový řádek `MS-DOS`[3] a postupujeme stejně jako uživatelé operačního systému `MS-DOS`.

2.5 Spouštění programu (MS-DOS)

Program spustíme zadáním `C:\>BIN\FRAC> frac.exe`. Po spuštění se na obrazovce objeví úvodní menu.

[3] postup pro vyvolání příkazového řádku je popsán v části 2.2.1

Obrázek 2.4: Úvodní menu

Kapitola 3

Ovládání

3.1 Volby v menu

3.1.1 Zobrazovat mezivýsledky

Je-li volba zapnutá `[X]`, program bude při počítání zobrazovat mezivýsledek. To jest vzorec, ve kterém jsou již spočítány všechny násobení, dělení a kde je spočítán výsledek všech závorek.

3.1.2 Vstup ze souboru

Je-li volba zapnutá `[X]`, program se po zadání příkazu k počítání nebude dotazovat na výraz, který chcete počítat, ale načte si výraz ze souboru `input.txt`. Do tohoto souboru můžete zapsat výrazy, jež chcete spočítat. Jednotlivé výrazy musí být každý na samostatné řádce a musí splňovat podmínky pro zadávání výrazů(viz. 3.2)

3.1.3 Výstup do souboru

Je-li volba zapnutá `[X]`, program se bude po zadání příkazu k počítání zapisovat výsledky i do souboru `output.txt`.

3.1.4 Export do TEXu

Obsah souboru `output.txt` převede na vstupní soubor pro sázecí software LaTeX. Výsledek se uloží do souboru `Vysledky.tex` a obsah souboru `output.txt` se smaže. Vypíše se informační hláška:

```
Soubor byl zapsan jako Vysledky.tex
```

a vyčká se stisku klávesy. Po stisku libovolné klávesy je program opět připraven přijímat vaše příkazy.

3.1.5 Počítat

Umožní počítat. Není-li nastaven vstup ze souboru, program se zeptá na výraz, jenž chcete spočítat - `Zadejte vyraz`. Zadejte tedy výraz[1] a potvrďte `enter`em. Po vypsání výsledku budete dotázáni na další zadání. Nechcete-li už dále pokračovat, zadejte `#exit` a potvrďte klávesou `Enter`. Program se vrátí do úvodní obrazovky.

3.1.6 Konec

Ukončí program.

3.2 Pravidla pro zadávání výrazů

3.2.1 Operátory

Ve výrazu se nesmí vyskytovat dva operátory hned po sobě. Chcete-li násobit či dělit záporným číslem, nebo chcete-li použít ve jmenovateli zlomku záporné číslo, musí bát uzavřeno do závorek. Program podporuje tyto operandy:

+	sčítání
−	odčítání
*	násobení
/	dělení

Tabulka 3.1: podporované operátory

3.2.2 Zlomky

Zlomky se zadávají v tomto tvaru: `čitatel_jmenovatel`. Čitatel i jmenovatel může být nahrazen uzávorkovaným výrazem. Např. `((1+4)*4)_(5-4)`.

3.2.3 Neurčité výrazy

Bude-li ve jmenovateli nula, nahradí se jedničkou a výsledek se označí jako zaokrouhlený (viz. 3.2.5)

3.2.4 Proměnné

Ve výrazech lze používat i proměnné. Po spuštění programu je k dispozici proměnná `ans`, do níž se ukládá hodnota posledního výrazu. Použití proměnné ve výrazu je možné zadáním znaku $ a bezprostředně za ním musí

[1] přečtěte si nejdříve pravidla pro zadávání výrazů v části 3.2

následovat název proměnné(např `$ans+1`). Názvy proměnných mohou obsahovat jen písmena. Výchozí hodnota všech proměnných je nula. Zadáte-li do výrazu proměnnou, jejíž hodnota zatím nebyla definována, dosadí se místo ní nula. O práci s proměnnými se můžete více dozvědět v části 4.1.4.

3.2.5 Zaokrouhlování

Bude-li čitatel nebo jmenovatel větší než 32000, bude výsledek (i mezi výsledek) zaokrouhlen. Tuto skutečnost indikuje nápis `cca`, jenž se zobrazí před výsledkem. V takovémto případě výsledek není přesný a může se od skutečného výsledku značně lišit. Proto je lepší pokusit se přeformulovat zadaný výraz tak, aby se všechny mezivýsledky vešly do daných mezí.

3.2.6 Funkce

Do zadávaných výrazů lze používat také uživatelem definované funkce. Jejich použití je velmi snadné. Stačí obdobně jako při zadávání proměnných napsat znak & a bezprostředně za něj název funkce. Má-li funkce nějaký parametru zadá se do závorek, jež bezprostředně následují(např. `&mocni(2)`). Pokud žádný parametr funkce nemá, nebo jej nechcete zadávat, závorky nepište. Je důležité si uvědomit, že funkce mají přístup ke všem proměnným a mohou je měnit. Tak mohou poskytovat více výsledků, nebo požadovat více parametrů. Více se dozvíte v kapitole 4.

Kapitola 4

Programování vlastních funkcí

Do programu lze doprogramovat vlastní funkce a pak je dále používat. Funkce se ukládají do souboru, jehož jméno je zároveň i názvem funkce. Nová funkce se dá vložit tak, že pomocí textového editoru vytvoříme v pracovním adresáři programu soubor, jehož jméno je názvem funkce. Z toho plynou i jistá omezení a to ta, že v systému `MS-DOS` může být maximální délka názvu souboru pouhých 8 znaků. To je také maximální délka názvu funkce. Soubor se ukládá bez jakékoliv přípony.

4.1 Příkazy

Řádky obsahující příkazy začínají vždy znakem #. Některé z příkazů jako svůj parametr požadují nějakou hodnotu. Tou může být zlomek(případně obyčejné číslo), proměnná(např. `$ans`), nebo i celý výraz. Pokud to ale má být výraz, musí být z obou stran uzavřen do závorek.

4.1.1 `exit`

Tento příkaz slouží pro ukončení vykonávání současné funkce(případně pro ukončení zadávání výrazů). Musí stát na samostatném řádku a má přednost před všemi ostatními.

4.1.2 `if`

Tento příkaz slouží k větvení. Přesná syntaxe je:

`if` *hodnota operátor hodnota příkaz.*

Operátorem musí být jeden ze znaků <, >, =. Pokud nerovnost(případně rovnost) platí, provede se následující příkaz. To může být `goto, set` nebo

`get` se svými parametry(např. `# if $ans > 10 goto end`). Pokud bude příkaz zadán špatně, vyhodnotí se, jako kdyby nerovnost(rovnost neplatila).

4.1.3 `goto`

Tento příkaz umožní skok na návěští. Syntaxe je "`goto` *návěští*", kde návěští je označená řádka programu. Návěští je jednoslovný název a zadává se takto: "`#` *navesti*`:`", kde mezi názvem návěští a dvojtečkou nesmí být mezera.

4.1.4 Práce s proměnnými

Pomocí příkazů lze také nastavit nebo přečíst hodnotu proměnné. Hodnotu lze nastavit pomocí příkazu "`set` *jméno hodnota*"(např `#`set result $ans). Tento příkaz lze použít i po příkazu `if`. Dále lze použít příkaz "`get` *název*" pro zjištění hodnoty proměnné(vypíše se na obrazovku). Pokud vaše funkce má podporovat vstupní parametr, předává se v proměnné `input`. Výsledek práce funkce by se měl zapsat do proměnné `result`. Tato proměnná se po skončění běhu funkce použije jako výsledek.

Část II

Programatorská dokumentace

Kapitola 1

Globální proměnné

Na začátku bych rád objasnil význam některých globálních proměnných a konstant určujících chování některých funkcí a procedur.

1.1 MAXCISLO

Tato konstanta má vliv na zaokrouhlování. Určuje, která čísla jsou už mimo rozsah a budou zaokrouhlena. Měla by být nastavena maximálně na $\lfloor\sqrt{\texttt{MAX}}\rfloor$, kde `MAX` je nejvyšší možné uložené číslo.

1.2 nula, jedna, deset

Používají se jako konstanty typu `cislo` obsahující patřičné prvky. Tyto konstanty se často hodí při výpočtech a tak se pro jednoduchost inicializují hned na začátku.

1.3 pozice

Je-li nastavená proměnná `rozpoznavam`, funkce pro rozpoznavani cisel ze `string`u začínají rozpoznávat až od znaku, který je určen touto proměnnou

1.4 rounded

Globální booleanové proměnná určující, zda během výpočtu došlo k zaokrouhlování. K zaokrouhlení může dojít při načítání (viz. 2.3.3) nebo při zaokrouhlování mezivýsledku(viz 2.2). Dojde-li k zaokrouhlování, připíše se před výsledek slovo `cca`.

1.5 `rozpoznavam`

Určuje, zda-li se zrovna vyhodnocuje výraz a má-li se brát v úvahu proměnná `pozice`.

1.6 `ans`

Tato proměnná je prvním členem spojového seznamu proměnných. Musí být alokovaná vždy, ukládají se do ní výsledky při počítání. Od ní dál pokračuje seznam dalších proměnných.

Kapitola 2

Postupy

V této kapitole se budu věnovat řešením jednotlivých dílčích problémů.

2.1 Ukládání zlomků

Zlomky jsou reprezentovány jako proměnná typu `record`. Tento typ obsahuje složku `cit` pro čitatele a `jmen` pro jmenovatele. Obě tyto složky jsou typu `cislo`. Tento typ momentálně označuje `longint`. Pro práci s čitatelem a jmenovatelem se používají zvláštní fukce. Toto řešení jsem zvolil proto, aby nebyl velký problém v budoucnu program rozšírit o možnost používání delších čísel ve jmenovateli i čitateli. Mělo by stačit nahradit pár funkcí a procedur, které operují přímu s těmito čísly. Jejich seznam je uveden v tabulce.

init	secti	prirad
nasob	rovna	vetsi
odecti	mymod	vypis

Tabulka 2.1: Seznam funkcí pracujících přímo s proměnnými typu `cislo`

2.2 Krácení

Pro krácení se používá Eukleidův algoritmus. Na začátku se ovšem ošetří některé speciální případy (neurčité výrazy a případ, kdy se čitatel rovná jmenovateli). Je-li čitatel nebo jmenovatel větší, než konstanta `MAXCISLO`, zmenší se oba o tolik řádů, aby výsledný zlomek splňoval zadané podmínky.

2.3 Načítání

2.3.1 Vstupy

Každou zpracovávanou řádku dostane nejdříve funkce `zpracuj`, která si sama otevře vstupní nebo výstupní soubory(je-li potřeba) a pokud řádek nezačíná znakem # tak nechá řádku zpracovat funkcí `vyraz`. V opačném případě zjistí, zda jde o podporovaný příkaz a pokud ano, provede ho.

2.3.2 Výrazy

Výrazy vyhodnocuje procedura `vyraz`. Ze zadaného `string`u začne od zarážky `pozice` vyhodnocovat veškeré násobení a dělení. Hodnoty získává pomocí funkce `fgetst`, která podle počátečního znaku buď načte zlomek, vyhodnotí výraz(pomocí funkce `vyraz`), dosadí hodnotu proměnné nebo vyhodnotí volání uživatelské funkce zavoláním funkce `zpracuj`, které se zakáže cokoliv vypisovat. Do pomocného `string`u se tak uloží už pouze zlomky a operace sčítání a odčítání. Tento string se případně vypíše jako mezivýsledek. Poté spočítá veškeré sčítání a odčítání v pomocné proměnné a tímto získá konečný výsledek.

2.3.3 Zlomky

U zlomku se načítá nejdříve čitatel a pak jmenovatel (je-li čitatel i jmenovatel obyčejné číslo). Je-li jmenovatel celý výraz, přiřadí se zlomku jmenovatel jedna a zkončí se na pozici znaménka zlomku. Pokud se čitatel nebo jmenovatel nevejde do zadaného rozsahu, zjistí se podle proměnné přetečení, okolik se nevešel a oba se zmeší o tolik řádů, o kolik se nevešel větší z nich, a zaznamená se zaokrouhlování. Toto se neprovede, je-li na místě čitatele nebo jmenovatele celý výraz.

2.3.4 Čísla

Pro rozpoznávání čísel se používá Hornerovo schéma.

2.4 Ostatní

Pro ostatní funkce se používají všeobecně známé postupy pro počítání se zlomky. Všechny operace se zlomky se převádí na posloupnost několika operací s proměnnými typu `cislo`, pro něž jsou základní operace již definovány, což by mohlo v budoucnu ulehčit práci při přepracování programu tak, aby byl schopen pracovat s přesnějšími čísly.