Epilog dialogu s kouzelnou hůlkou

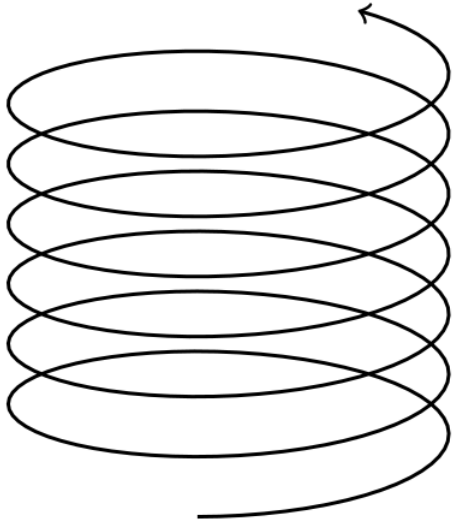
Máme za sebou čtyři kapitoly popisující základy kouzelného jazyka: jaká jsou podstatná jména a která slovesa se k nim hodí, jak pojmenovat konkrétní informace ukryté v podstatných jménech, rozhodování ponechané na kouzelné hůlce a jak kouzelnou hůlku naučit nová kouzelná slovesa.

Nedá se říci, že neprogramátor je teď čaroděj. Ještě chvíli potrvá, než dokáže v kouzelném jazyce plynule vyjádřit své myšlenky, než přijde na to, jaké myšlenky to vlastně mají být, než se naučí čarodějnické triky ověřené staršími generacemi, jak zaznamenat a zkontrolovat kouzelné věty a jak spolupracovat s ostatními čaroději.

Neprogramátor by ale měl už vědět, cože to to programování-čarování vlastně je - že je to vytváření kouzelných vět, které vykoná kouzelná hůlka - a měl by umět přečíst a rozumět jednoduchým větám kouzelného jazyka.

... že programování je čarování, že programátoři zvládnou udělat věci, které normální lidé nezvládnou...

(Aaron, The Internet's Own Boy)



Neprogramátor

Těší mě, Neprogramátor

Myslím si, že Aaronova představa programování jako čarování je vlastně dost přesná. Čaroděj vytvoří kouzelnou větu v neznámém jazyce, neznámého významu, která se neznámo jak vykoná, aby nakonec dostal, co chtěl.

Ale přece každý jazyk je neznámý, dokud se nenaučíme slova a skladbu věty. A žádná věta nedává smysl, dokud se v jazyce nenaučíme vyjádřit své myšlenky. Zbývá pak už jediný rozdíl mezi čarodějem a neprogramátorem - kouzelná hůlka, která kouzelnou větu vykoná. A ani čaroděj většinou netuší, jak to ta hůlka provede.

Kdybych se někdy znovu učil programovat, chtěl bych začít s How to Design Programs (HTDP). Kouzelný jazyk, který HTDP používá, se jmenuje Racket a kouzelná hůlka je vývojové prostředí DrRacket.

Programovací jazyk Racket, a Scheme vůbec, má tu výhodu, že je syntakticky jednoduchý. To znamená, že jde jednoduše vysvětlit, jak vytvořit kouzelnou větu - začátek a konec je pevně daný, stejně jako kde je to *co* se má stát a kde je to *s čím* se to má stát.

To ale neubírá na síle programovacího jazyka! Jestliže Neprogramátor je úvod k HTDP, pak zkratka je Scheme Primer a základ R5RS. A pak je tu samozřejmě SICP.

Takže… cože to to programování vlastně je? Jakože čarovat? Můžu se to naučit? A jak?

Dialog s kouzelnou hůlkou představuje interaktivní způsob programování, *Dopis pro kouzelnou hůlku* vysvětluje práci programátorů a *Nekonečně kouzelné jazykolamy* pojednávají o tom, jak a proč toho programátoři tolik zvládnou. *Kouzelné čtení* a *Čarodějnický žargon* jsou pro doplnění.

Učíme se ve spirále a Neprogramátor aspiruje být první částí spirály k programování.

Slovníček kouzelného jazyka

Kouzelné sloveso define může čaroděj použít k tomu, aby kouzelnou hůlku naučil nové kouzelné sloveso. To udělá tak, že napíše, jak by nová kouzelná věta (používající nové kouzelné sloveso) měla vypadat a co by se mělo stát s podstatnými jmény použitými v nové kouzelné větě:

```
(define (nové-kouzelné-sloveso podstatná-jména oddělená mezerami)
  (co-vykonat-s podstatná-jména oddělená mezerami))
```

Kde kouzelné sloveso co-vykonat-s musí kouzelná hůlka znát předem.

Je čas na dialog s kouzelnou hůlkou

```
(define (obsah šířka výška)
  (* šířka výška))

(define (obvod šířka výška)
  (* 2 (+ šířka výška)))

(define (obdélník šířka výška)
  (let ((orientován (cond
                      ((> šířka výška) "na šířku")
                      ((> výška šířka) "na výšku")
                      (else "čtverec")))
        (text-obsah (number->string (obsah šířka výška)))
        (text-obvod (number->string (obvod šířka výška))))
    (string-append "Obdélník "
                   (number->string šířka)
                   " x "
                   (number->string výška)
                   " je " orientován "."
                   " Jeho obsah je " text-obsah
                   " a obvod " text-obvod ".")))

(obdélník 10 3)
(obdélník 3 10)
(obdélník 4 4)
```


Kouzelný jazyk, kouzelná hůlka

Jazyk *čeština* má deset druhů slov: podstatná jména (třeba "kniha"), přídavná jména (třeba "kouzelná"), zájmena, číslovky (třeba 0 a 3.14), slovesa (třeba sečti a spoj), a pak ještě příslovce, předložky, spojky, částice a citoslovce. Každá věta začíná velkým písmenem, pak jsou slova oddělená mezerami, a nakonec tečka ., vykřičník !, nebo otazník ?.

Sečti 0 a 3.14!

Spoj "kouzelná" a "kniha".

V jazyce *čeština* je stavba věty složitá, takže příklady vět zní kostrbatě a nejspíš by se řeklo spíše:

Prosím, sečti pro mě čísla nula a 3.14.

Co vznikne, když ke slovu "kouzelná" přidáš slovo "kniha"?

Kouzelný jazyk *Racket* má jen dva druhy slov: podstatná jména a slovesa. Podstatná jména ukrývají nějakou informaci. Třeba "kniha", "kouzelná", 0 a 3.14 jsou všechno podstatná jména kouzelného jazyka. Slovesa říkají, co se má s podstatnými jmény stát. Třeba + a string-append jsou slovesa kouzelného jazyka. Každá kouzelná věta začíná závorkou (a hned po ní následuje sloveso. Potom jsou podstatná jména oddělená mezerami a úplně nakonec zase závorka).

(+ 0 3.14)

(**string-append** "kouzelná" "kniha")

Aby čaroděj mohl v kouzelném jazyce vyjádřit, co chce, kromě slov a skladby věty (odborně syntax) potřebuje znát i význam slov (odborně sémantika), protože i když slova zná a vytvoří kouzelnou větu podle správných pravidel, výsledek nemusí dávat smysl. Věta

Prosím, sečti pro mě nulu a "kniha".

smysl nedává, stejně tak ani kouzelná věta

(+ 0 "kniha")

A nakonec je tu kouzelná hůlka, která umí *vykonat* kouzelné věty zapsané v kouzelném jazyce *Racket*. Kouzelná hůlka vykoná kouzelnou větu

```
(+ 0 3.14)
```

tak, že se z ní stane

```
3.14
```

a z kouzelné věty

```
(string-append "kouzelná" "kniha")
```

```
"kouzelnákniha"
```

Když kouzelná hůlka vykoná kouzelnou větu, výsledek je slovo kouzelného jazyka. Může to být podstatné jméno ukrývající informaci, kterou chtěl čaroděj získat. Nebo to může být sloveso, které čaroděj použije pro další čarování.

Protože výsledek vykonání kouzelné věty je slovo kouzelného jazyka, může čaroděj použít více jednodušších vět, aby vytvořil jednu dlouhou mocnou kouzelnou větu.

```
(+ (+ 0 3.14) (+ 0 3.14))
```

```
6.28
```

Jenže aby se vykonala kouzelná věta složená z menších, musí se nejprve vykonat ty menší. O to se už ale postará kouzelná hůlka.

Tentokrát ale po define následuje nová kouzelná věta, jak by ji chtěl čaroděj kouzelné hůlce napsat, a hned potom její význam, tedy co nová kouzelná věta znamená.

Takové kouzelné větě se říká *definice funkce obsah s parametry šířka a výška*. Podstatným jménům v nové kouzelné větě se říká *parametry*. Významu, který následuje, se říká *implementace funkce*.

Pak už kouzelná hůlka odpoví i na kouzelnou větu:

```
(obsah 10 3)
```

Takové kouzelné větě se říká *volání funkce obsah s argumenty 10 a 3*, nebo *aplikace funkce obsah na argumenty 10 a 3*. Argumenty jako aktuální hodnoty.

Stejným způsobem naučí čaroděj kouzelnou hůlku i kouzelné sloveso obvod:

```
(define (obvod šířka výška)
  (* 2 (+ šířka výška)))
```

A pak se kouzelné hůlky zeptá na:

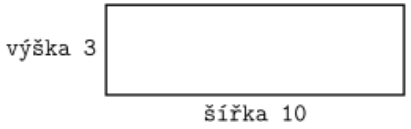
```
(obvod 10 3)
```

Nová slovesa: definice funkcí

Čaroděj vymýšlí kouzelné věty, které kouzelné hůlce píše do REPL. Kouzelná hůlka kouzelné věty vykoná a čaroději napíše zpátky výsledek. Často se ale stane, že čaroděj napíše - nebo chce napsat - kouzelnou větu, kterou kouzelná hůlka nechápe. Protože čaroděj použil kouzelná slova, která kouzelná hůlka nezná.

Jak na nová kouzelná podstatná jména víme z Pojmenování informací: definice proměnných; s novými kouzelnými slovesy je to podobně.

Obsah a obvod obdélníku



Obrázek 10: Obrázek obdélníku šířky 10 a výšky 3.

Obsah obdélníku se vypočítá šířka × výška a čaroděj tak píše kouzelné hůlce:

```
(* 10 3)
```

Raději by ale psal:

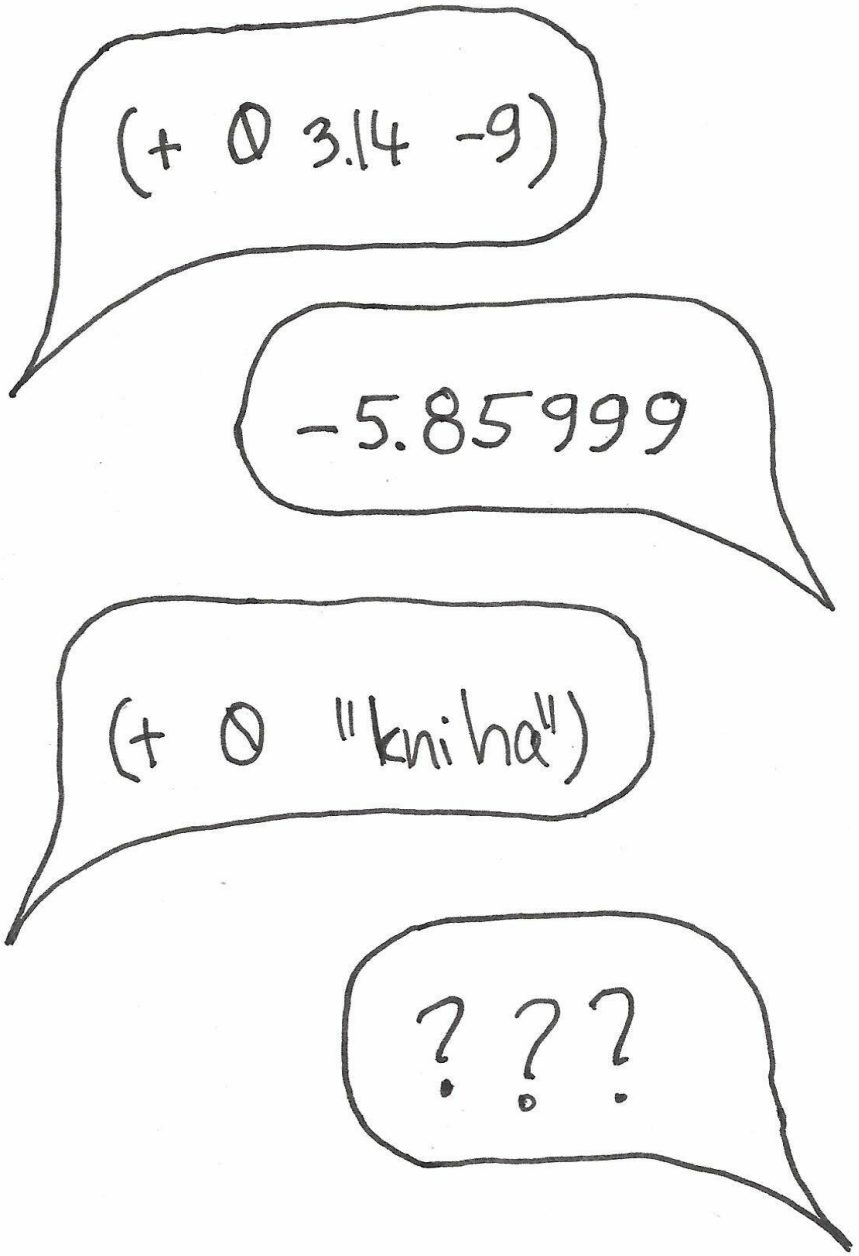
```
(obsah 10 3)
```

To aby bylo jasné, co po kouzelné hůlce chce.

Čaroděj tedy kouzelné hůlce vysvětlí, jak nová kouzelná věta vypadá - jakým slovesem začíná a jaká podstatná jména následují. Vysvětlí jí také význam nové kouzelné věty - co se s podstatnými jmény v nové kouzelné větě má stát.

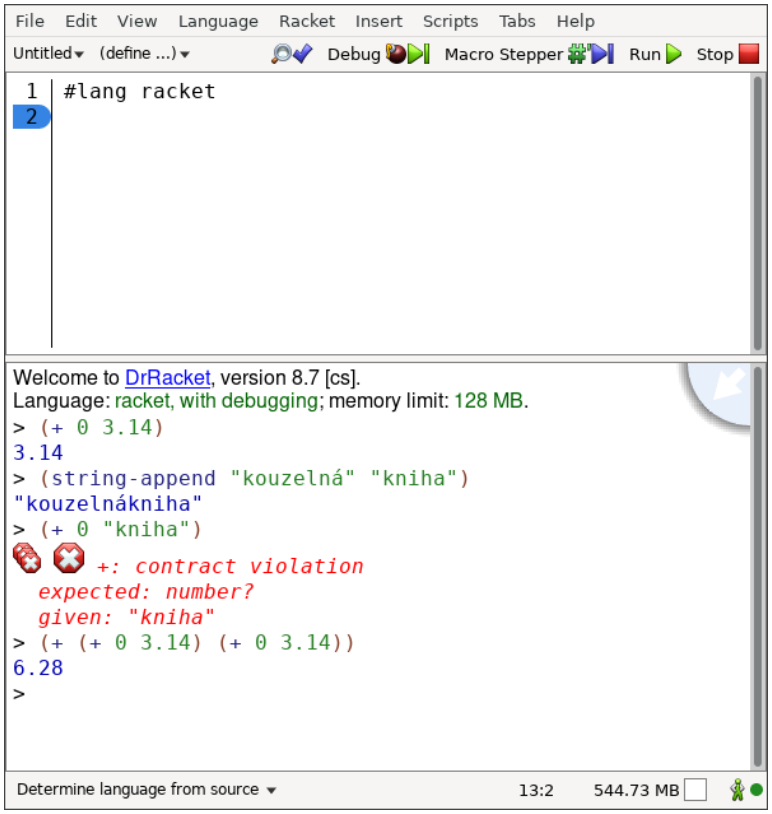
Stejně jako pro nová podstatná jména, použije čaroděj pro nová slovesa (a tedy pro nové kouzelné věty) kouzelnou větu začínající kouzelným slovesem define:

```
(define (obsah šířka výška)
  (* šířka výška))
```



Dialog s kouzelnou hůlkou

Kouzelná hůlka poslouchá, jestli po ní čaroděj něco nechce. Pokud ji čaroděj předá kouzelnou větu, kouzelná hůlka kouzelnou větu přečte a vykoná, vytiskne výsledek a pak čeká, až jí čaroděj předá nějakou další kouzelnou větu. Anglicky se *Přečti, Vykonej, Vytiskni (výsledek), Opakuj* řekne *Read, Evaluate, Print, Loop*, a tak se dialogu s kouzelnou hůlkou říká zkratka REPL.



Obrázek 1: Screenshot kouzelné hůlky (vývojového prostředí) DrRacket s REPL v dolní polovině. Kouzelné věty se do REPL zadávají postupně - i když se vykonání třetí kouzelné věty nepovedlo, vykonání čtvrté kouzelné věty zase ano.

REPL je ve vývojovém prostředí DrRacket prostor v dolní polovině a použijeme jej pro sdílení kouzelných vět s kouzelnou hůlkou.

- = všechna čísla jsou si rovna?
- < všechna čísla řazena vzestupně?
- > všechna čísla řazena sestupně?
- <= všechna čísla řazena neklesajíce?
- >= všechna čísla řazena nestoupajíce?

I když pro některá ano:

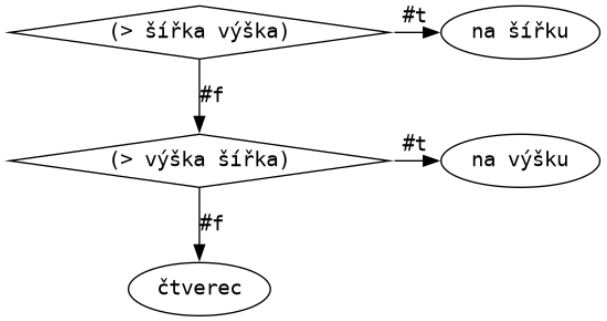
- **number?** je číslo?
- **integer?** je celé číslo?
- **zero?** je nula?
- **positive?** je kladné?
- **negative?** je záporné?
- **odd?** je liché?
- **even?** je sudé?

Kouzelná slovesa pro text, která se používají v predikátech, konvenci s otazníkem na konci dodržují:

- **string?** je text?
- **string=?** jsou texty stejné?
- **string-ci=?** jsou texty stejné bez ohledu na velikost písmen?
- **string<?**, **string>?**, **string<=?**, **string>=?** jsou texty lexikograficky řazené?
- **string-ci<?**, **string-ci>?**, **string-ci<=?**, **string-ci>=?** jsou texty lexikograficky řazené bez ohledu na velikost písmen?

A kouzelná slovesa pro pravdivostní hodnoty jsou tři, všechna bez otazníku:

- **and** konjunkce
- **or** disjunkce
- **not** negace



Obrázek 9: Vývojový diagram "na šířku či na výšku".

```
(let ((šířka 10)
      (výška 3))
  (cond
    ((> šířka výška) "na šířku")
    ((> výška šířka) "na výšku")
    (else "čtverec")))
```

Za kouzelným slovesem **cond** následují kouzelné skoro-věty. Skoro- proto, že na místě, kde je v kouzelné větě sloveso, je v kouzelné skoro-větě predikát. Za predikátem už následuje kouzelné podstatné jméno nebo kouzelná věta, kterou má kouzelná hůlka vykonat, když situace nastane (výsledek vykonání predikátu je #t). Každá z kouzelných skoro-vět následujících po **cond** tak kouzelné hůlce říká: "Jestli na moje první slovo nebo větu odpovíš #t, vykonej ty další! To je výsledek!"

Kouzelné slovo **else** je vlastně #t a v kouzelné skoro-větě tedy říká: "Vždy vykonej co je za mnou!" **else** se hodí, když čaroděj ví, co má kouzelná hůlka udělat, pokud všechny ostatní situace nenastaly (výsledky vykonání predikátů jsou #f).

Slovníček kouzelného jazyka

Kouzelným větám začínajícím kouzelnými slovesy **if** a **cond** se říká podmínkové výrazy - jejich vykonávání je podmíněné predikátem.

Konvence pro predikáty je, že končí otazníkem (?); pro kouzelná slovesa použitá v predikátech pro čísla to ale většinou neplatí:

Podstatná jména: primitivní datové typy

Podstatná jména kouzelného jazyka uchovávají nějakou informaci. Jsou ale různé typy informací - desetinná čísla, celá čísla, nebo text, či pravdivostní hodnota: pravda a nepravda. S každým typem podstatných jmen je potřeba zacházet trochu jinak, a tak se pro každý typ podstatného jména hodí trochu jiná slovesa.

Pojmenování informací: definice proměnných

Když se stejná informace, třeba číslo nebo text, používá na více místech, hodí se si ji zapamatovat. To se nejlépe udělá tak, že se informace pojmenuje a její jméno se pak použije všude tam, kde je informace třeba.

Vykonat či nevykonat: predikáty a podmínkové výrazy

Když kouzelná hůlka vykoná kouzelnou větu, výsledkem je slovo kouzelného jazyka. A když je to slovo podstatné jméno a informace v něm ukrytá buď pravda, nebo nepravda, říká se, že ona kouzelná věta je predikát.

Čarodějové často tvoří mocné kouzelné věty, kde použijí predikát a dvě další kouzelné věty. Podle toho, jestli je predikát pravda nebo nepravda, vykoná kouzelná hůlka první nebo druhou z těch dvou dalších kouzelných vět. Takovým mocným kouzelným větám se říká podmínkový výraz a kouzelná hůlka tak čarodějům pomáhá s rozhodováním.

Nová slovesa: definice funkcí

Čaroděj občas ví, jakou kouzelnou větu by chtěl s kouzelnou hůlkou sdílet - jaké by použil sloveso a co by se mělo stát s podstatnými jmény. Jenže kouzelná hůlka žádné takové sloveso nezná. Potom čaroději nezbývá, než kouzelnou hůlku nové kouzelné sloveso naučit. Musí ale použít jen slovesa a podstatná jména kouzelného jazyka, kterým kouzelná hůlka rozumí.

Podstatná jména: primitivní datové typy

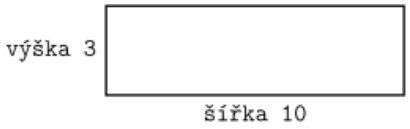
Začít jednoduchým příkladem je složité, protože jednoduché příklady nejsou žádné čáry. Sečíst první číslo s druhým je snadné, na to člověk nemusí být čaroděj. Jenže když se učíme nový jazyk, začínáme právě s jednoduchými slovy a jednoduchými větami. Kouzelný jazyk není výjimkou.

(+ 0 3.14) je věta kouzelného jazyka, která sečte 0 a 3.14. Podstatná jména této kouzelné věty jsou čísla a čísla se v kouzelném jazyce píší stejně. Slovesem této kouzelné věty je +, symbol, který používáme v matematice.

Slovesa pro čísla

Pro podstatná jména kouzelného jazyka, která uchovávají nějakou číselnou informaci, se kromě kouzelného slovesa + hodí ještě -, *, /, nebo **min**, **max**, a **abs**.

Neprogramátora občas trápí, jaký je obsah obdélníku, který je 10 cm široký a 3 cm vysoký. Obsah obdélníku se vypočítá šířka × výška, tedy 10 × 3 = 30 cm2.



Obrázek 2: Obrázek obdélníku šířky 10 a výšky 3.

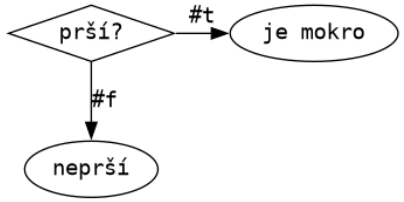
Čaroděj se o svoje trápení podělí s kouzelnou hůlkou:

```
(* 10 3)
```

Neprogramátor i čaroděj musí tedy vědět, co chce vypočítat a jak. Třeba i obvod takového obdélníku; ten je 2 × (šířka + výška). Neprogramátor pak zapíše do kalkulačky 2 × (10 + 3) a kalkulačka mu vrátí výsledek 26 cm. Čaroděj místo toho napíše kouzelné hůlce do REPL:

```
(* 2 (+ 10 3))
```

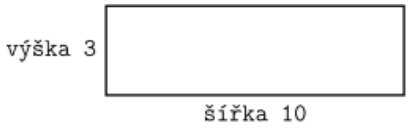

No a aby se čarodějové neztratili v množství situací, na které kouzelnou hůlku připravují, kreslí si obrázky, kterým říkají *vývojové diagramy* nebo *rozhodovací stromy*.



Obrázek 7: Vývojový diagram "když prší".

Co když je možností více?

Čarodějům se vždycky nakonec podaří všechny situace popsat predikáty, tedy kouzelnými slovy či kouzelnými větami, na které kouzelná hůlka odpoví vždy jen #t nebo #f.



Obrázek 8: Obrázek obdélníku šířky 10 a výšky 3.

Třeba obdélník může být nakreslený "na šířku", jako na obrázku. Když není nakreslený "na šířku", tak může být nakreslený "na výšku". A když není ani "na výšku", není to obdélník!

Kouzelné věty můžeme vnořovat a tvořit tak mocné kouzelné věty:

```
(let ((šířka 10)
      (výška 3))
  (if (> šířka výška)
      "na šířku"
      (if (> výška šířka)
          "na výšku"
          "čtverec")))
```

Pokud je možných situací opravdu hodně, není vnořování vždy přehledné. V takových případech čarodějům místo **if** poslouží **cond**:

Vykonat či nevykonat: predikáty a podmínkové výrazy

Čaroděj může kouzelnou hůlku připravit na různé situace. To udělá tak, že pro každou situaci, která může nastat, vymyslí kouzelné slovo. Nebo kouzelnou větu. A pak vymyslí kouzelné slovo či větu pro tu samou situaci, když nenastane. K tomu čaroděj použije kouzelné sloveso **if**:

```
(if prší?
    "je mokro"
    "neprší")
```

Krátká anglická poznámka: Z angličtiny se *if* přeloží jako "když".

Pojmenování informací: definice proměnných už tu bylo, takže můžeme napsat kouzelné hůlce do REPL:

```
(let ((prší? #t))
  (if prší?
      "je mokro"
      "neprší"))
```

a taky

```
(let ((prší? #f))
  (if prší?
      "je mokro"
      "neprší"))
```

Kouzelné větě začínající kouzelným slovesem **if** se říká *podmínkový výraz*. Výsledek vykonání podmínkového výrazu záleží na pravdivostní hodnotě prvního podstatného jména, kterému se říká *predikát*. (Výsledek vykonání predikátu je vždy #t nebo #f.) Dohoda je, že predikáty končí otazníkem (?), ale neplatí to vždy. A protože kouzelné věty lze vnořovat a vytvářet tak mocné kouzelné věty, může být namísto prvního podstatného jména nějaká kouzelná věta; v takovém případě končí otazníkem sloveso.

To je mocná kouzelná věta složená ze dvou. Výsledek vykonání té vnitřní, (+ 10 3), je podstatné jméno kouzelného jazyka 13. Kouzelná hůlka tak z kouzelné věty (* 2 (+ 10 3)) udělá nejdříve (* 2 13) a pak 26.

Slovesa pro text

Nejdůležitější slovesa kouzelného jazyka pro práci s textem jsou: **string-length**, **substring**, **string-append**, **number->string** a **string->number**. Hodně kouzelných sloves připomíná angličtinu.

Jak dlouhé je slovo "Neprogramátor"? Pokud se neprogramátor nepřehlédl, má 12 znaků. Čaroděj mezitím píše kouzelné hůlce do REPL:

```
(string-length "Neprogramátor")
```

A také celkem jednoduše nechá kouzelnou hůlku odpovědět celou větou, protože **string-append** slouží pro spojení textu:

```
(string-append "Délka slova Neprogramátor je "
               (number->string (string-length "Neprogramátor")))
```

Pro každý typ podstatného jména se hodí trochu jiná slovesa. Také se očekává, že slovesa kouzelného jazyka se budou používat se správným typem podstatných jmen. Byla by proto chyba použít pro sloveso **string-append** podstatné jméno, které je výsledkem vykonání kouzelné věty (**string-length** "Neprogramátor"), protože **string-append** se hodí na text, jenže výsledek vykonání (**string-length** "Neprogramátor") je číslo.

To ale neznamená, že slovesa kouzelného jazyka očekávají pořád stejný typ podstatných jmen, jako třeba v kouzelné větě o části textu:

```
(substring "Neprogramátor" 2 (string-length "Neprogramátor"))
```

Kouzelné sloveso **substring** očekává jako první podstatné jméno text. Jako druhé a třetí očekává ale číslo - pozici, od které má text zkrátit, a pozici kam až. Pozor, pozice začínají od nuly!

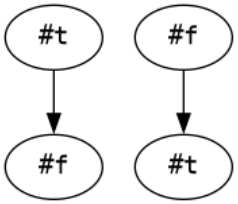
Pravdivostní hodnoty

Podstatná jména #t a #f jsou pravdivostní hodnoty. Pravda a nepravda. K těm se hodí slovesa **and**, **or** a **not**. Větami kouzelného jazyka složených z takových podstatných jmen a sloves se zabývá Logika.

Krátká anglická poznámka: Anglicky je pravda *true* a nepravda *false*, proto #t a #f. V angličtině znamená *and* spojku "a", *or* znamená "nebo" a *not* znamená "ne".

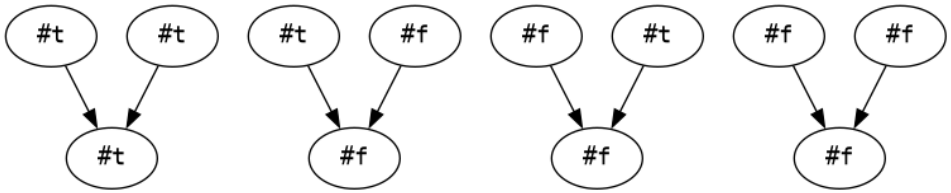
Krátká logická poznámka: Lež není opak pravdy. Opakem pravdy je nepravda, zatímco lež je úmyslné uvedení v omyl.

Kouzelné větě začínající slovesem **not** se říká *negace*. Taková kouzelná věta má jedno podstatné jméno. Pro (**not** #t) je výsledek vykonání #f, a naopak, na kouzelnou větu (**not** #f) odpoví kouzelná hůlka #t.



Obrázek 3: negace (not)

Kouzelná věta začínající slovesem **and** je *konjunkce* a věta začínající **or** je *disjunkce*. Obě mají kromě slovesa ještě dvě podstatná jména.



Obrázek 4: konjunkce (and)

Slovníček kouzelného jazyka

Slovesa **let** a **let*** slouží čaroději pro dočasné pojmenování informací v jedné větě.

Slovesem define čaroděj naučí kouzelnou hůlku novou informaci, kterou si kouzelná hůlka zapamatuje.

```
(let ((slovo "Neprogramátor"))
  (let ((část (substring slovo 2 (string-length slovo))))
    (string-append "Délka slova " část " je "
                   (number->string (string-length část)))))
```

Pomocí **let** lze pojmenovat podstatná jména, která na sobě nezávisí. Jenže podstatné jméno část závisí na podstatném jméně slovo. Místo **let** je tak potřeba použít **let***. **let*** je sloveso kouzelného jazyka, které je hodně podobné **let**, jen o trochu lépe umí pojmenovat více věcí najednou - následující slova, například část, mohou záviset na těch předchozích, v tomto případě na slovo:

```
(let* ((slovo "Neprogramátor")
       (část (substring slovo 2 (string-length slovo))))
  (string-append "Délka slova " část " je "
                 (number->string (string-length část))))
```

Nová slova, která si kouzelná hůlka zapamatuje

Když čaroděj řekne kouzelné hůlce:

```
(let ((šířka 10)
      (výška 3))
  (* šířka výška))
```

kouzelná hůlka odpoví 30. Když se hned na to čaroděj zeptá:

```
šířka
```

kouzelná hůlka neví. Čaroděj naučil kouzelnou hůlku nová slova jen na chvilku.

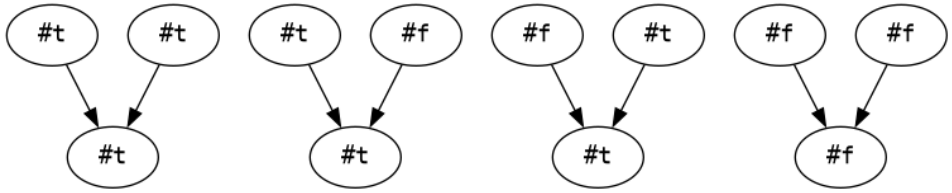
Aby si kouzelná hůlka nová slova zapamatovala, vysvětlí je čaroděj v kouzelných větách začínajících slovesem define:

```
(define šířka 10)
(define výška 3)
```

Kouzelná hůlka pak už ví a na šířka odpoví 10, na výška odpoví 3 a na (* šířka výška) odpoví 30. A na (* 2 (+ šířka výška)) samozřejmě 26.

Kouzelná hůlka odpoví #t na kouzelnou větu (**and** #t #t). Na všechny ostatní kouzelné věty začínající kouzelným slovesem **and**, tedy (**and** #t #f), (**and** #f #t) a (**and** #f #f), odpoví kouzelná hůlka #f.

Obdobně je to s kouzelným slovesem **or** - kouzelná hůlka odpoví #f jen na kouzelnou větu (**or** #f #f). Na všechny ostatní, tedy (**or** #f #t), (**or** #t #f) a (**or** #t #t), odpoví kouzelná hůlka #t.



Obrázek 5: disjunkce (or)

Pravdivostní hodnoty jsou důležité při rozhodování; použijeme je v kapitole *Vykonat či nevykonat: predikáty a podmínkové výrazy*.

Slovníček kouzelného jazyka

Podstatná jména:

- čísla jako třeba 0 nebo 3.14
- text jako třeba "Neprogramátor"
- pravdivostní hodnoty #t a #f

Slovesa pro čísla:

- + sečti čísla
- - odečti čísla
- * vynásob čísla
- / vyděl čísla
- **min** najdi nejmenší číslo
- **max** najdi největší číslo
- **abs** absolutní hodnota čísla

Slovesa pro text:

- **string-length** počet znaků textu
- **substring** část textu od do
- **string-append** spojení textů
- **number->string** číslo na text
- **string->number** text na číslo

Slovesa pro pravdivostní hodnoty:

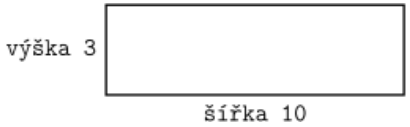
- **and** konjunkce
- **or** disjunkce
- **not** negace

Pojmenování informací: definice proměnných

Stejně jako se čaroděj může naučit nová slova kouzelného jazyka, může se nová slova naučit i kouzelná hůlka. Tedy je to spíše čaroděj, kdo kouzelnou hůlku nová slova naučí. Když použije ty správné kouzelné věty.

Nová slova v jedné kouzelné větě

Kouzelná věta začínající slovesem **let** může přispět k přehlednosti při obdélníkových počtech i hledání délky slova.



Obrázek 6: Obrázek obdélníku šířky 10 a výšky 3.

Obsah obdélníku se vypočítá šířka × výška. Kouzelné sloveso **let** použijeme v kouzelné větě, ve které pojmenujeme číselnou informaci ukrytou v kouzelných podstatných jménech 10 a 3:

```
(let ((šířka 10)
      (výška 3))
  (* šířka výška))
```

Obvod je 2 × (šířka + výška) a kouzelné hůlce píšeme do REPL:

```
(let ((šířka 10)
      (výška 3))
  (* 2 (+ šířka výška)))
```

Čaroděj může pojmenovat i podstatná jména typu text:

```
(let ((slovo "Neprogramátor"))
  (string-append "Délka slova " slovo " je "
                 (number->string (string-length slovo))))
```

Navíc čaroději nic nebrání použít výsledek vykonání kouzelné věty začínající kouzelným slovesem **let** jako část jiné kouzelné věty: