

Návrh a implementace jazyka pro code-golf challenge

Filip Kliber

Matematicko-fyzikální fakulta, Univerzita Karlova, 2016

Code-golf

Druh programátorské soutěže, ve které je cílem vyřešit relativně snadnou programátorskou úlohu tak, že zdrojový kód řešení je nejkratší, co do počtu bytů.

Příklad. Pro zadání „Rozhodnout, zda je číslo prvočíslo nebo ne“, mohou řešení v různých jazycích vypadat takto:

Java	C++
<pre>class P{public static void main (String[]a){int i=2,n=Short. valueOf(a[0]);for(;i<n;)n=n%i++ <1?0:n;System.out.print(n>1);}}</pre>	<pre>#include<ios> int main(int,char**a){int n=atoi(a[1]),i=2;for(;i<n;)c*=n %i++;puts(c&&n>1?"1":"0");}</pre>
R	Pyth
<pre>n=scan();cat(sum(!n%%1:n)==2)</pre>	<pre>}QPQ</pre>

Cíle práce

Navrhnout programovací jazyk, který umožňuje zápis stručného řešení, ale zachovává čitelnost zdrojového kódu.
Implementovat překladač tohoto jazyka a běhové prostředí ve formě standardní knihovny.

Ukázky

Program, který na standardní výstup vypíše obsah standardního vstupu, pokud vstup neodpovídá přímo textu `cat`. Pak se na výstup vypíše řetězec `cat goes "Meow"`:

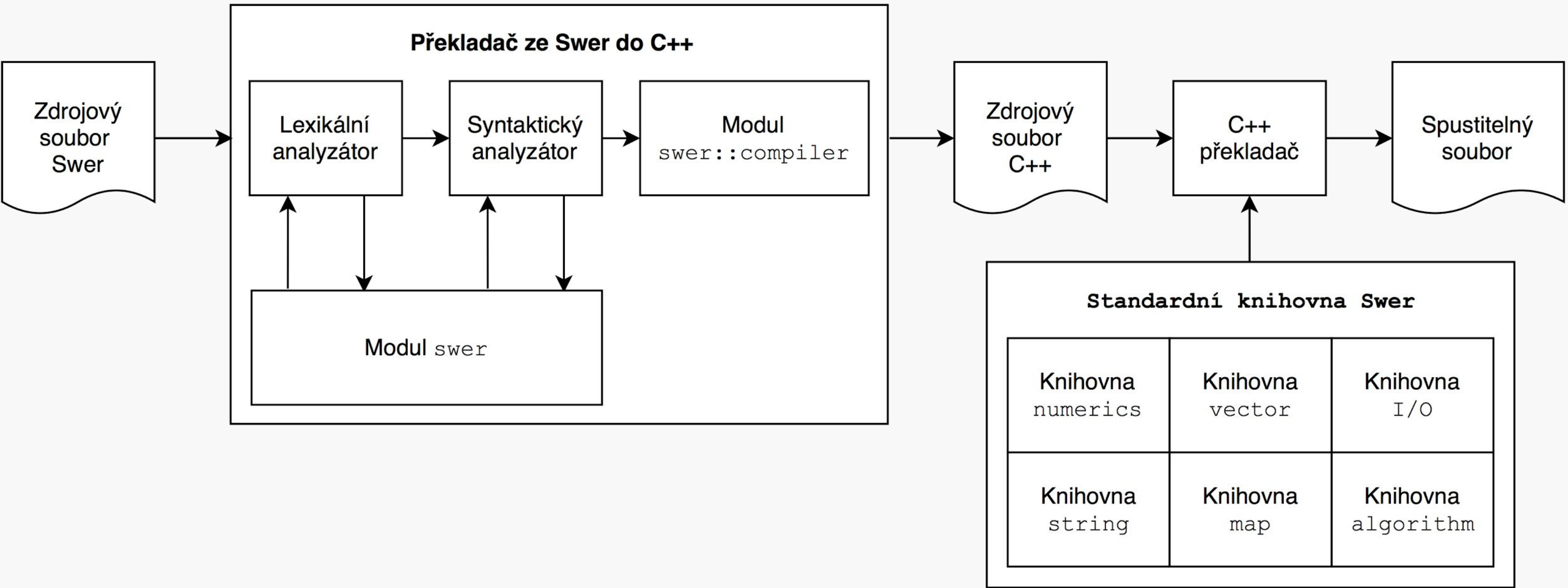
```
$s = re(); // deklarace proměnné s, která obsahuje celý vstup
s == "cat" // test na rovnost
? p(s, " goes \"Meow\"") // výstup pro s == "cat"
: p(s); // výstup pro s != "cat"
```

Pro dané `t` z intervalu $(-10^9, 13)$, vypsát hodnotu `gamma` funkce `v` bodě `t`:

```
t=>{ $v=1.; // deklarace lambda výrazu. Deklarace proměnné v
@ (1..10^9) // cyklus od 1 do 10^9 s implicitní řídicí proměnnou i
v *= (1+1./i)^t / (1 + t/i); // krok cyklu
-> v / t; // příkaz return
};
```

Swer

Swer je název implementovaného jazyka pro code-golf challenge.



Architektura řešení – překladač swer

Většina identifikátorů v jazyce Swer má dvě verze, výřečnou a zkrácenou (zpravidla na 1-3 znaky dle užitečnosti).

- Datové typy: `string` (`"text"`), `integer` (`5`), `double` (`7.3`), `pole` (`[7, 8, 9]`).
- Příkazy: `if-else`, `switch`, `for`, `for-each`, `lambda` výrazy.
- Operátory: Většina běžně používaných z C-like jazyků, navíc pak `faktorial` (`!`), `umocnění` (`^`), `bitový xor` (`^^`).
- Standardní typy
 - `string`: jako `std::string` z C++, navíc pak `.c(.size())`, `.r(.reverse())`
 - `pole`: jako `std::deque` z C++, použití pomocí operátorů: `+` pro přidání prvku a `konkatenaci`, `-` pro odebírání.
 - `mapa`: jako `std::map`, použití pomocí operátorů
 - `I/O`: funkce `r_()`, kde `_` je první písmeno chtěného typu. Dále `r1()` pro načtení řádku, `re()` pro načtení celého vstupu. Funkce `p()` zajišťuje výstup.
 - Algoritmy: Z funkcionálního programování: `map`, `filter`, `take_while`, `fold`
 - Řetězce a pole se při indexování chovají jako cyklické kontejnery.