Netradiční programovací jazyky

Matěj Klíma

FJFI ČVUT

20. listopadu 2012

Obsah prezentace

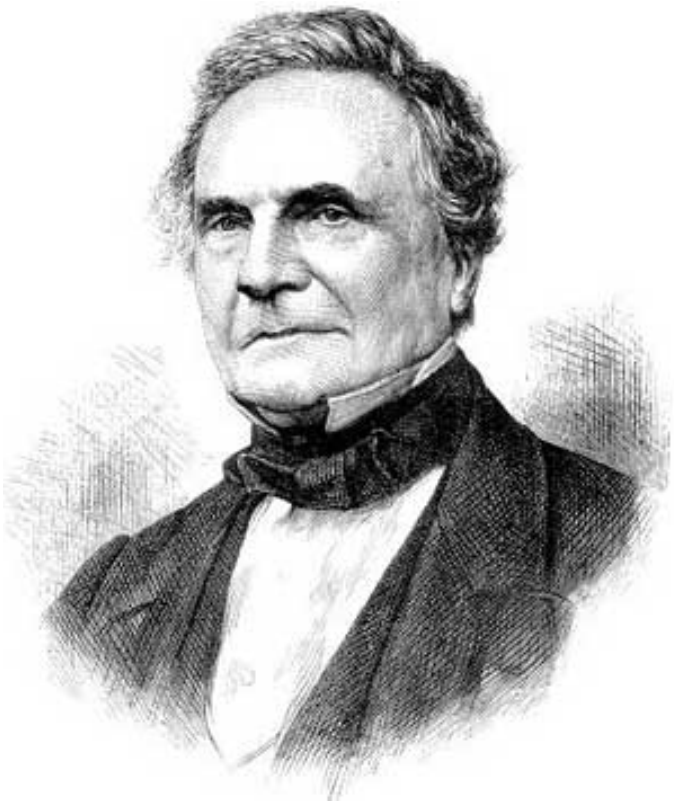
- Historický úvod - První programovací jazyky
- Logické programování - PROLOG
- Esoterické programovací jazyky

Mechanické výpočetní stroje

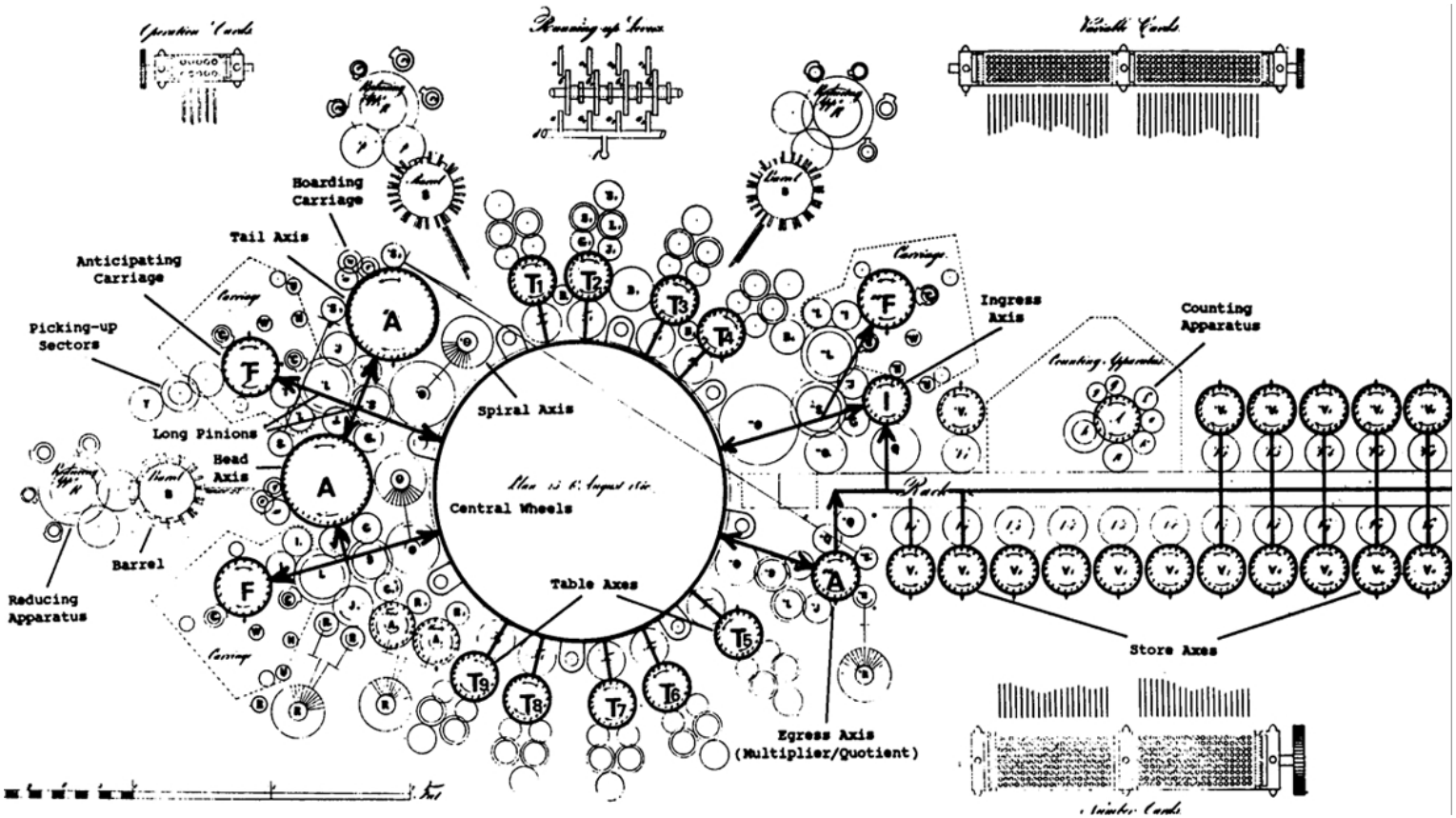
- 1623 - William Schickard - první mechanický kalkulátor
- Gottfried Leibniz - jeho stroj uměl i násobení a dělení
- 18. století - další rozvoj, počátek průmyslové výroby
- 1801 - Joseph Marie Jacquard - tkalcovský stav programovatelný pomocí děrných štítků

Charles Babbage (1791 – 1871)

- Navrhoval komplikované stroje pro výpočet tabulkových hodnot funkcí
- Analytical Engine - návrh mechanického počítače
- První Turingovsky úplný stroj, podobná struktura jako pozdější elektronické počítače
- Ada Lovelace - algoritmus na výpočet Bernoulliho čísel pro tento stroj



Charles Babbage (1791 – 1871)

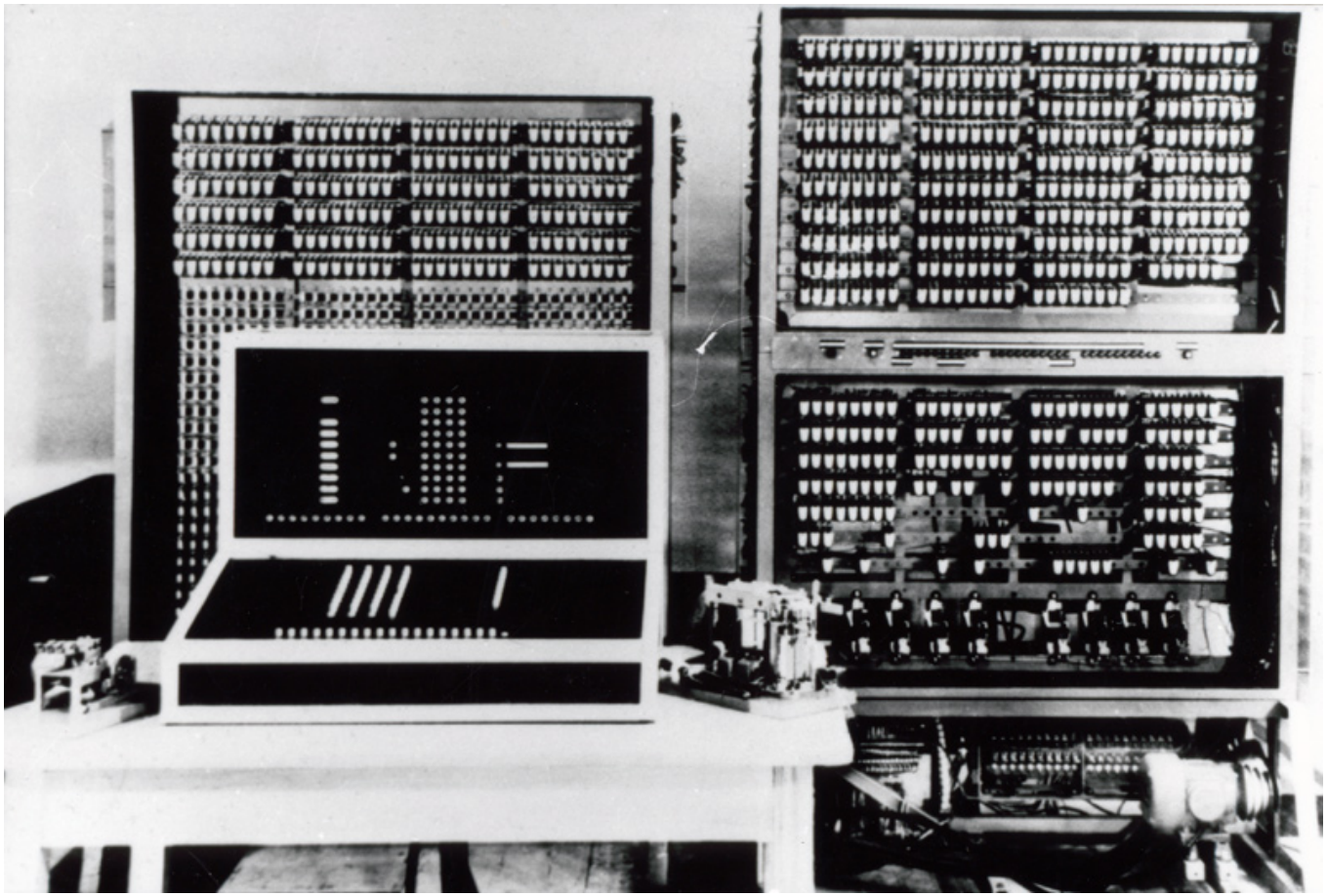


Obrázek: Jedno ze schémat Analytického stroje

Konrad Zuse (1910 – 1995)

- První funkční elektromechanický počítač (řada Z1-Z4), používá relé
- Z3 (1941), programovatelný počítač, 8bit instrukce, 22bit registr, 64x22bit paměť, takt 5,33 Hz
- Plankalkül (”Výpočetní plán”) - první koncept vysokoúrovňového programovacího jazyka

Konrad Zuse (1910 – 1995)



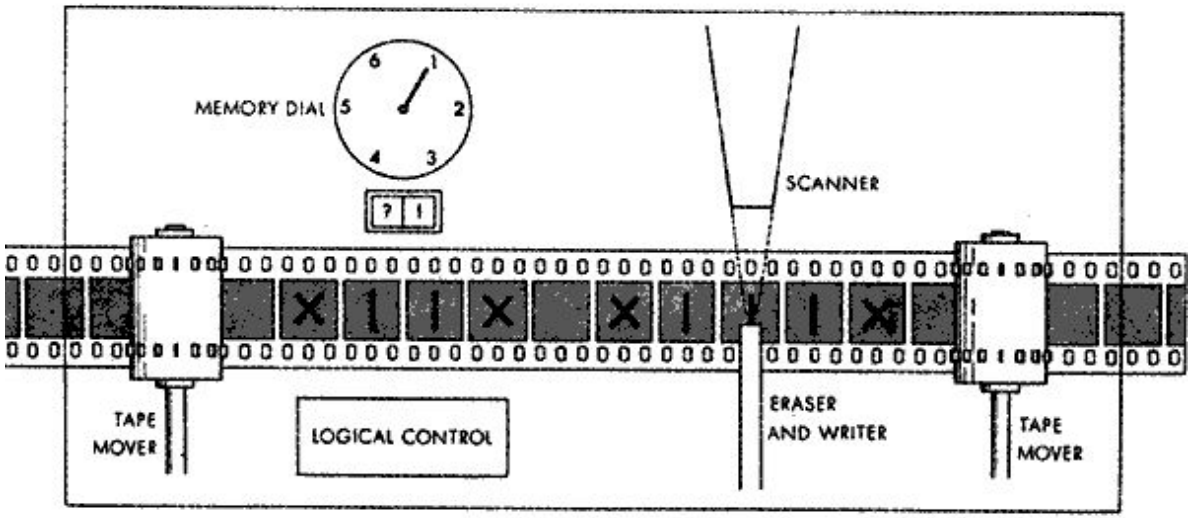
Obrázek: Počítač Z3 (1941)

Alan Turing (1912 – 1954)

- 1936 - Koncept Turingova stroje, teoretický model obecného počítače
- Programovací jazyk je Turingovsky úplný, pokud dokáže simulovat Turingův stroj
- Během druhé světové války pomáhal luštit německé šifry v Bletchley parku
- Turingův test - návrh na testování inteligentního stroje

Turingův stroj

- Skládá se z pásky, pohyblivé hlavy a stavového registru
- Šest základních instrukcí - čtení, zápis, pohyb vlevo a vpravo, změna stavu a zastavení stroje
- Pomocí těchto instrukcí lze vytvářet cykly i podmíněné skoky
- Turingův stroj je schopen provádět stejné výpočty jako dnešní počítače (při dostatku času a paměti)



Další vývoj

- 1946 - První elektronický počítač ENIAC v USA
- 50. léta - vznik mnoha programovacích jazyků, některé používané dodnes (FORTRAN, LISP, COBOL, ALGOL...)
- 70. léta - C, Pascal, Basic, Bourne shell, MATLAB, TeX...
- 80. léta - C++ (a objektové jazyky obecně), Perl, PostScript
- 90. léta - Python, Java, PHP, Visual Basic

Logické programování

- Jedno ze 4 hlavních programovacích paradigmat
- Programování pomocí prostředků matematické logiky
- Často je deklarativní - popisujeme vstup a cíl výpočtu, nemáme zde překladač, ale řešič/dokazovač
- Algoritmus se dělí na vlastní logický program a řízení řešení, které je součástí jazyka
- Existují i překladače, které z logického programu vytvoří procedurální program (často kvůli efektivitě)

PROLOG (1972)

- Nejznámější logický programovací jazyk
- Využití zejména v oblastech umělé inteligence a zpracování přirozeného jazyka
- Snaha o abstraktní vyjádření vztahů a co největší potlačení imperativní složky
- Využívá predikátovou logiku prvního řádu, konkrétně Hornovy klauzule:

$$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \implies u$$

- Relativně jednoduchá syntaxe, navržen pro lingvisty

Ukázka kódu:

Klauzule:

```
muz(pavel).
rodic(jana,pavel).
dite(X,Y) :- rodic(Y,X).
syn(X,Y) :- dite(X,Y), muz(X).
```

- Atomy - řetězce znaků začínající malými písmeny (konstanty)
- Proměnné - začínají velkým písmenem, platí pouze pro jednu klauzuli
- Fakta - jednoduché deklarativní klauzule
- Pravidla - složitější klauzule s implikacemi

Ukázka kódu:

Dotazy:

```
?- muz(jana).
no.
?- dite(pavel,jana).
yes.
?- syn(X,jana).
X = pavel;
no.
```

- Dotaz pomocí proměnné, která se vyskytuje pouze jednou vrátí hodnotu této proměnné.

Esoterické programovací jazyky

- Jazyky, které nejsou určeny k praktickému použití
- Slouží k demonstraci nových možností programování, případně experimentům se syntaxí i principy programování.
- Většina z nich však slouží pro zábavu.
- "Turing-tarpit"- turingovsky úplný jazyk s minimálním množstvím příkazů/operátorů
- Některé jazyky experimentují s jiným rozložením kódu než v textovém souboru, např. v prostoru...
- Nedeterministické jazyky - probabilistické definice funkcí pro nepředvídatelný běh programu

INTERCAL (1972)

- Byl vytvořen jako parodie tehdejších programovacích jazyků.
- "Compiler Language With No Pronounceable Acronym"
- Vychází z předpokladu, že čím nesrozumitelnější je kód, tím váženější je jeho autor.
- Používá direktivy jako "IGNORE", "FORGET", případně "PLEASE".
- Přesto je turingovsky úplný.
- `http://www.catb.org/~esr/intercal/`

Obrázek: Jeden z autorů - Jimbo Lyon

Program Hello World:

```
DO ,1 <- #13
PLEASE DO ,1 SUB #1 <- #238
DO ,1 SUB #2 <- #108
DO ,1 SUB #3 <- #112
DO ,1 SUB #4 <- #0
DO ,1 SUB #5 <- #64
DO ,1 SUB #6 <- #194
DO ,1 SUB #7 <- #48
PLEASE DO ,1 SUB #8 <- #22
DO ,1 SUB #9 <- #248
DO ,1 SUB #10 <- #168
DO ,1 SUB #11 <- #24
DO ,1 SUB #12 <- #16
DO ,1 SUB #13 <- #162
PLEASE READ OUT ,1
PLEASE GIVE UP
```

Brainfuck (1993)

- Extrémně minimalistický programovací jazyk, autor Urban Müller.
- Příkazy $<$, $>$ slouží pro posun datového ukazatele.
- Příkazy $+$,- slouží pro zvýšení/snížení aktivní hodnoty.
- Příkazy . a , slouží k vypsání/načtení jednoho znaku ze std. vstupu.
- Příkazy [,] přesunou instrukční ukazatel doprava/doleva za druhou hranatou závorku, pokud je aktivní hodnota rovna/různá od nuly.
- Některé překladače mají méně než 200 bytů.
- Komentáře lze psát jako normální text, veškeré ostatní znaky jsou ignorovány.
- `http://esolangs.org/wiki/Brainfuck/`

Příklady

Hello World:

```
++++++++++[>+++++++>++++++++++>+++>+<<<<
-]>++.>+.+++++++..+++.>++.<<++++++++++++
+++.>.+++.------.--------.>+.>.
```

Sečtení dvou číslic a vypsání ASCII znaku s výslednou hodnotou:

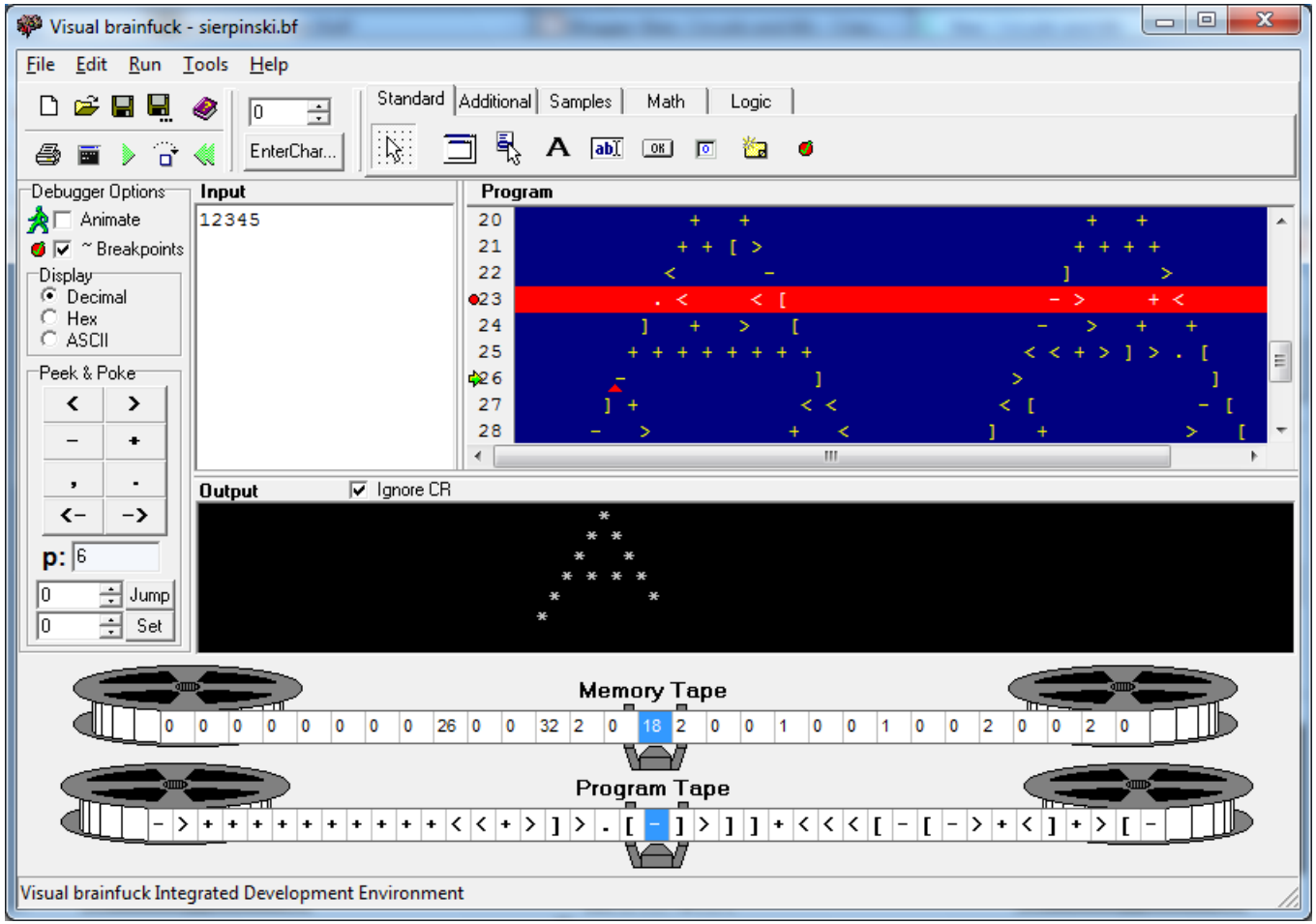
```
,>++++++[<-------->-],[<+>-],<.>.
```

Cyklus vypisující hodnotu na vstupu, ukončený mezerníkem:

```
+[,.----------------------------]
```

Visual Brainfuck

http://sites.google.com/site/visualbf/



Befunge (1993)

- Autor Chris Pressey, původně snaha o co nejobtížněji kompilovatelný jazyk.
- Narozdíl od konvenčních jazyků jsou zde příkazy uspořádány v 2-rozměrné síti.
- Má poměrně hodně příkazů, mimo jiné umožňuje napsat sebe modifikující kód.
- Základní příkazy jsou šipky $<$, $>$, $\wedge$, v pro kontrolu běhu programu.
- Číslo 0-9 způsobí načtení dané hodnoty do zásobníku, uvozovky pro řetězec.
- Cykly vytvoříme jako smyčky v programu, podmínky jako větvení cesty...
- `http://catseye.tc/`

Příklady:

Hello World:

```
>              v
v  ,,,,,"Hello≪
>48*,          v
v,,,,,,"World!≪
>25*,@
```

Jiná možnost:

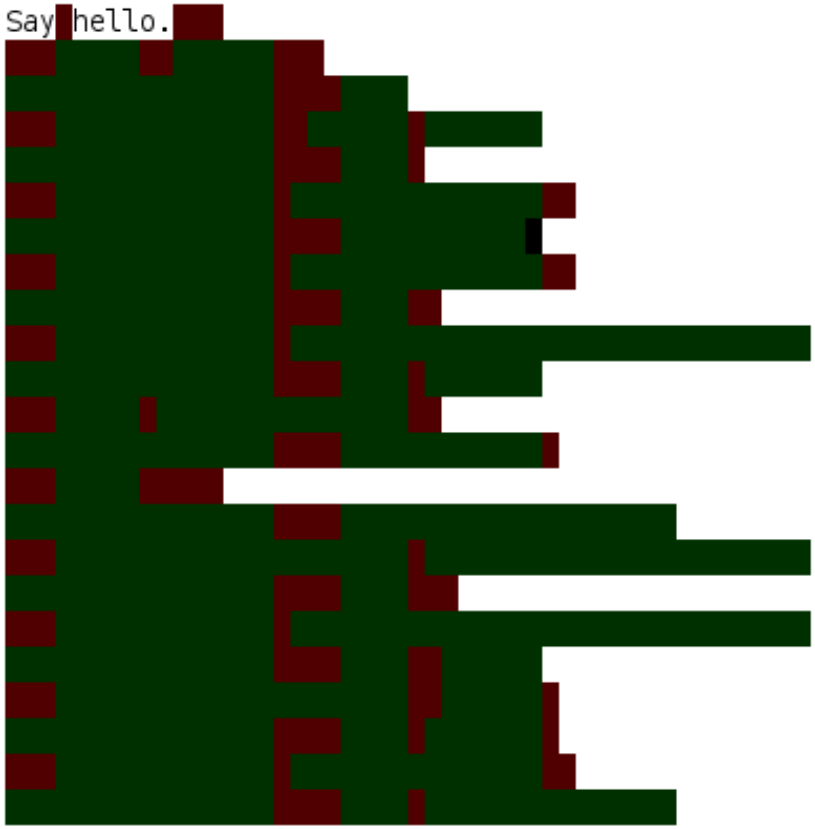
```
>25*"!dlrow ,olleH":v
                   v:,_@
                   >  ^
```

Whitespace (2003)

- Autoři Edwin Brady a Chris Morris, uvolněn 1.4.2003.
- Většina jazyků tzv. whitespace znaky ignoruje, zde je tomu naopak, ignorovány jsou všechny ostatní.
- Je to imperativní jazyk, běží na virtuálním stroji s haldou a zásobníkem.
- Příkazy jsou složeny pouze z mezerníků, tabulátorů a odřádkování, data taktéž (mezerník představuje 0, tabulátor 1).
- Přes zdánlivou nesmyslnost lze tento koncept využít v steganografii.
- `http://compsoc.dur.ac.uk/whitespace/`

Hello World ve Whitespace

Prvky syntaxe jsou zvýrazněny - tabulátory zeleně, mezery červeně...



7,8-32 Anfang

LOLCODE (2007)

- Jazyk inspirován populárním internetovým meme "Lolcats".
- Syntaxe není jednotná, existují různé interpretry nebo překladače.
- Jeden z dialektů je Turingovsky úplný, což bylo dokázáno napsáním interpreteru Brainfucku.
- Praktické využití veskrze žádné, ale může přitáhnout nové zájemce o programování.
- `http://lolcode.net/`

Příklady:

Hello World:

```
HAI
CAN HAS STDIO?
VISIBLE "HAI WORLD!"
KTHXBYE
```

Otevření a vypsání souboru:

```
 HAI
 CAN HAS STDIO?
 PLZ OPEN FILE "LOLCATS.TXT"?
     AWSUM THX
         VISIBLE FILE
     O NOES
         INVISIBLE "ERRROR!"
 KTHXBYE
```

Další zdroje

- `http://history-computer.com`
- `http://en.wikibooks.org/wiki/Prolog`
- `http://www.logic.at/prolog/faq/faq.html`
- `http://esolangs.org`
- Wikipedia (i česká)

Děkuji za pozornost.