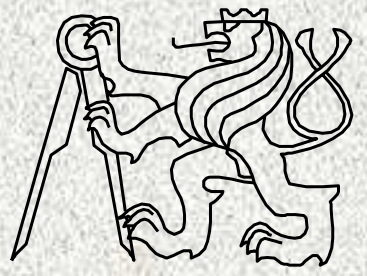
Programování 1 - úvod

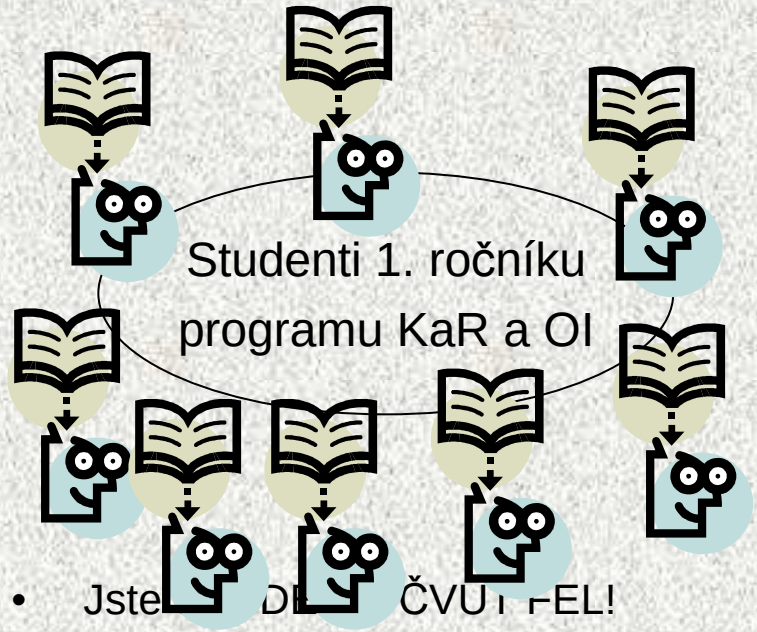


A0B36PR1-Programování 1

Fakulta elektrotechnická

České vysoké učení technické

Vítejte, představme se … ☺



- Jste [illegible] ČVUT FEL!
- Jste členy akademické obce!
- Děkan, senát, vědecká rada
- Zaměstnanci (tituly …), doktorandi
- Katedry

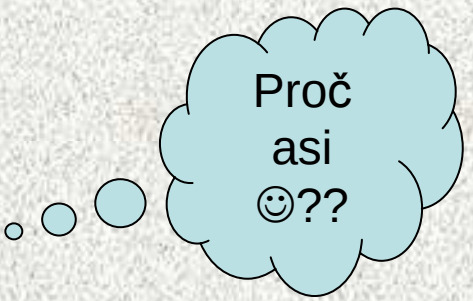
Ing. Miroslav Balík, PhD.
Doc. Ing. Ivan Jelínek, CSc.

Cvičící:….

Kdo je kdo v Programování I

1. Vy – studenti
   - Různé zkušenosti s různými programovacími jazyky
2. Přednášející:
   - Ing. Miroslav Balík, PhD.,
   - Doc.Ing. Jelínek Ivan CSc.
3. Nejdůležitější učitelé: cvičící!!

- Ing. Balík Martin
- Ing. Bloch Martin, CSc.
- Ing. Buk Zdeněk
- Ing. Chludil Jiří
- Ing. Kuchař Jaroslav
- Ing. Malinský Radek



- Garant: Doc.Ing. Jelínek Ivan CSc.

Cíl předmětu

- Programování 1

- prerekvizita Programováni 2
- prerekvizita Algoritmizace

- *reprezentace dat v počítači*
- *reprezentace čísel*
- základní struktury jazyka Java
- cykly
- jednoduché programy v Javě
- ladění programů
- procedurální vs. objektový přístup
- objekty, třídy
- soubory a proudy, kolekce,
- …

Organizace a hodnocení předmětu

- A0B36PR1 Programování 1
- Rozsah: 2p+2c
- Zakončení Z,ZK
- Kredity 6
  - po prvním semestru je nutné získat a
  - výsledná známka ovlivní možnost tv
- https://webdev.felk.cvut.cz/courses/A0B36PR1/

(*)Způsoby zakončení předmětu:
- zápočet
- klasifikovaný zápočet
- zápočet / zkouška

Hodnocení a zkouška

Zdroje bodů pro hodnocení	Body
aktivita a DÚ na cvičeních	20 b (min. 10 b)
semestrální práce	20 b (min. 10 b)
test u počítače na cvičeních	30 b (min. 15 b)
písemný zkouškový test	25 b (min. 10 b)
Ústní zkouška	25 b (-10b pokud k ústní)

Body ze cvičení, maximálně 70. 65 a více bodů →výborně bez zkoušky

Klasifikace *na základě bodového hodnocení)*

klasifikace	*počet bodů*	číselně	slovně	
A	90 - 100	1	výborně	
B	80 - 89	1,5	velmi dobře	
C	70 - 79	2	dobře	
D	60 – 69	2,5	uspokojivě	
E	50 - 59	3	dostatečně	
F	< 50	4	nedostatečně	

Možnost nechat si zapsat známku nebo jít k ústní zkoušce – odečte se 10 bodů

Zakončení předmětu: zápočet, **zkouška** *(na základě bodového hodnocení)*

Doporučená literatura

Základní zdroje:

- Poznámky z přednášek a cvičení
- Slidy z přednášek https://webdev.felk.cvut.cz/courses/A0B36PR1
- Základní příručky jazyka Java:
  - Herout, P.: Učebnice jazyka Java, Kopp, 2007
  - Keogh, J.:Java bez předchozích znalostí, Computer Press, 2005
  - Virius, M.: Java pro zelenáče, Neocortex, 2001
  - Zakhour, S: Java 6, výukový kurz, CPress, Brno, 2007

Další zdroje (publikace v češtině**)**:

- Troníček, Z: Programovaci jazyk Java, 2007
- Eckel, B.: Myslíme v jazyku Java, Grada, 2000, I + II
- Chapman, S., J.: Začínáme programovat v jazyce JAVA, Computer Press, 2001
- Pitner,T.: Java, začínáme programovat, Grada, 2002
- Hawlitzek, JAVA2, příručka programátora, Grada, 2000
- Shildt, H.: Java 2, Příručka programátora, Softpress, 2001
- Herout, P.: JAVA, grafické uživatelské prostředí a čeština, Kopp, 2001

Základní pojmy

- *Definujeme základní pojmy, shrneme důležité vlastnosti a vztahy (u řady z nich se předpokládá znalost!!)*
  - *Informatika*
  - *Software*
  - *Hardware*
  - *Operační systém*
  - *Algoritmus*
  - *Programovací jazyk*
    - *Překladač*
    - *Kompilátor*
  - *Reprezentace dat v počítači*

Informatika

Computer science is no more about computers than astronomy is about telescopes.

Informatika se nezabývá počítači o nic více než astronomie dalekohledy.

- Edsger Dijkstra, informatik

- zabývá se zpracováním informací nejen na počítačích
- studuje výpočetní a informační procesy z hlediska hardware i software – „*technická* informatika“
- je součástí teorie informací, věda spojující aplikovanou matematiku a elektrotechniku za účelem kvantitativního vyjádření informace

Software – programové vybavení

- V informatice sada všech počítačových programů v počítači
- Software zahrnuje
  - operační systém (zajišťuje běh programů)
  - aplikační software (pracuje s ním uživatel)
  - další (knihovny, middleware, BIOS, firmware apod.)

Hardware

Zahrnuje všechny fyzické součásti počítače

- čistě elektronická zařízení (procesor, paměť, display)
- elektromechanické díly (klávesnice, tiskárna, diskety, disky, jednotky CD-ROM, páskové jednotky, reproduktory) pro vstup, výstup a ukládání dat.

- Počítač se skládá z procesoru, operační paměti a vstupně-výstupních zařízení.

Operační systémy - dělení

- OS mainframů (velké počítače pro kritické aplikace) - VMS, OS/360,..
- OS osobních počítačů řady PC - HP-UX, Solaris, BSD, MS DOS, MS Windows,…
- OS osobních počítačů řady Apple - Systém 1..Systém 9, Mac OS X,…
- OS kapesních počítačů, PDA, komunikátorů a smartphonů - Symbian, PalmOS, Android,…
- OS pro vestavné (embeded) systémy
- Smart Card OS
- Realtimové OS

Algoritmus

- Algoritmus
  - postup při řešení určité třídy úloh, který je tvořen seznamem jednoznačně definovaných příkazů a zaručuje, že pro každou přípustnou kombinaci vstupních dat se po provedení konečného počtu kroků dospěje k požadovaným výsledkům
- Vlastnosti algoritmu:
  - *hromadnost*
    měnitelná vstupní data
  - *determinovanost*
    každý krok je jednoznačně definován
  - *konečnost a resultativnost*
    pro přípustná vstupní data se po provedení konečného počtu kroků dojde k požadovaným výsledkům
- Algoritmus – syntetický model postupu řešení obecných úloh
- Prostředky pro zápis algoritmu
  - přirozený jazyk, vývojové diagramy, struktogramy, pseudojazyk, programovací jazyk

Algoritmus, příklad

- Úloha:

  Najděte největšího společného dělitele čísel 6 a 15

- Řešení:

  Popišme postup tak, aby byl použitelný pro dvě libovolná přirozená čísla,

  nejen pro 6 a 15:

  - označme zadaná čísla *x* a *y* a menší z nich *d*
  - není-li *d* společným dělitelem *x* a *y*, pak zmenšíme *d* o 1, test opakujeme a skončíme, až *d* bude společným dělitelem *x* a *y*

- Poznámka:

  Význam symbolů *x*, *y* a *d* použitých v algoritmu:

  - jsou to **proměnné** (paměťová místa), ve kterých je uložena nějaká hodnota, která se může v průběhu výpočtu měnit

Algoritmus – společný dělitel

Úloha: najděte největšího společného dělitele čísel 6 a 15
Průběh řešení:

krok	**x**	**y**	**d**	**poznámka**
	6	15	?	zadání vstupních dat
1	6	15	6	
2	6	**15**	**6**	**d není dělitelem y**, proveď krok zmenšení d
3	6	15	5	
2	**6**	15	**5**	**d není dělitelem x,** proveď krok zmenšení d
3	6	15	4	
2	**6**	**15**	**4**	**d není dělitelem x ani y,** proveď krok zmenšení d
3	6	15	3	
2	**6**	**15**	**3**	**d je dělitelem x i y,** proveď krok 4
4	6	15	3	**výsledek je hodnota 3**

Algoritmus - příklad

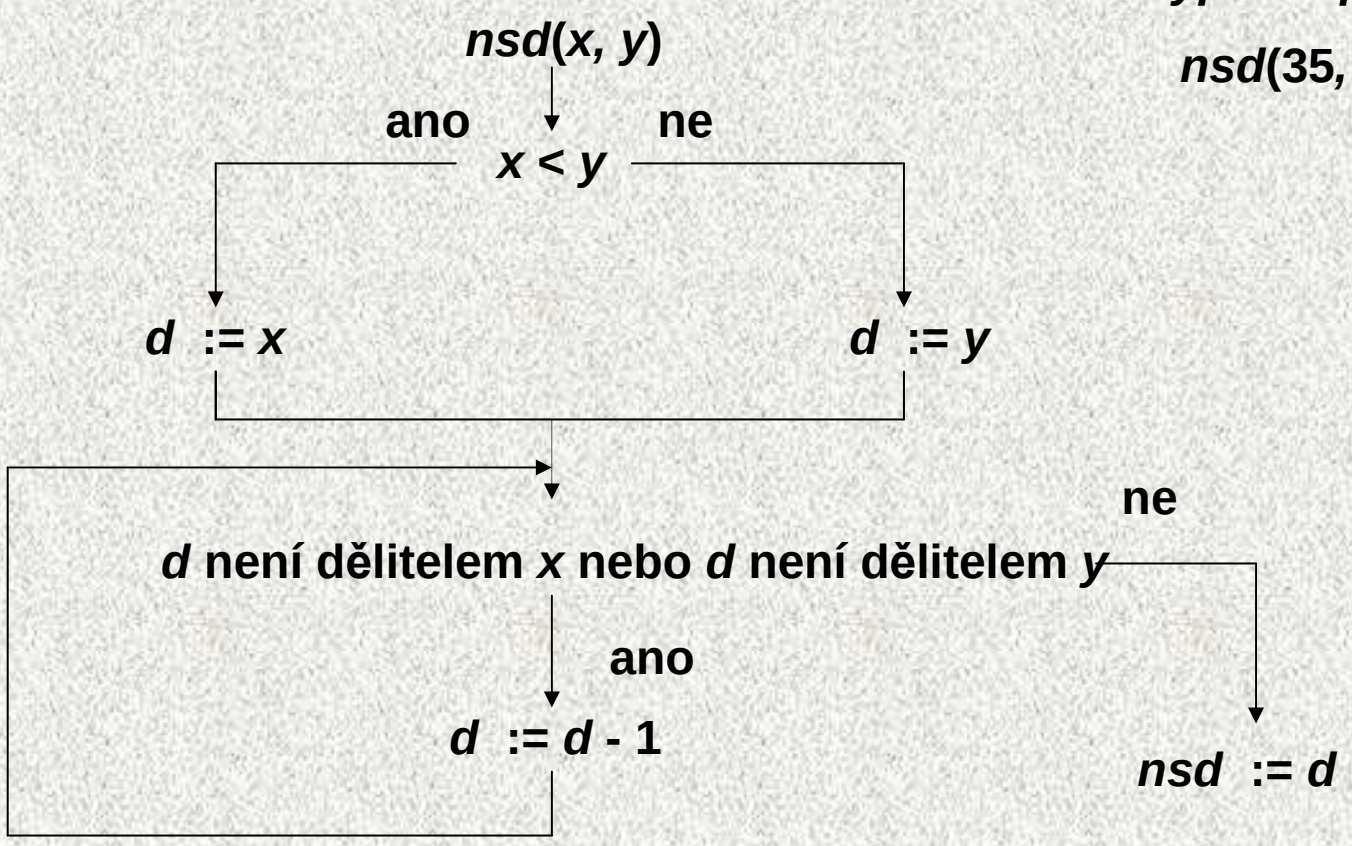
- Úloha:
  najděte největšího společného dělitele

- Obecný popis*:*
  Vstup: přirozená čísla *x* a *y*
  Výstup: *nsd*(*x,y*)
  Postup:
  1. Je-li *x*<*y*, pak *d* má hodnotu *x*, jinak *d* má hodnotu *y*
  2. Opakuj krok 3,pokud *d* není dělitelem *x* nebo *d* není dělitelem *y*
     3. Zmenši *d* o 1
  4. Výsledkem je hodnota *d*

- Sestavili jsme algoritmus pro výpočet největšího společného dělitele dvou přirozených čísel

Algoritmy

- Vývojový diagram



Programy a programovací jazyky

- Program je předpis (zápis algoritmu) pro provedení určitých akcí počítačem zapsaný v programovacím jazyku
- Programovací jazyky
  - strojově orientované
    - strojový jazyk = jazyk fyzického procesoru
    - asembler (jazyk symbolických adres)
  - vyšší jazyky
    - **imperativní** (příkazové, procedurální)
    - neimperativní (např. funkcionální)
- Hlavní rysy imperativních jazyků (např. C, C++, **Java,** Pascal, Basic, ...)
  - zpracovávané údaje mají formu datových objektů různých typů, které jsou v programu reprezentovány pomocí proměnných resp. konstant
  - program obsahuje deklarace a příkazy
  - deklarace definují význam jmen (identifikátorů)
  - příkazy předepisují akce s datovými objekty nebo způsob řízení výpočtu

Algoritmy a progra

- Zápis algoritmu v pseudojazyku

```
nsd(x,y):
  if x<y then d:=x else d:=y;
  while d „není dělitelem“ x or d „není dělitelem“ y do
    d:=d-1;
  nsd:=d;
```

- Zápis algoritmu v programovacím jazyku

```
int nsd(int x, int y){
  int d;
  if (x<y) d=x; else d=y;
  while (x%d!=0 | |  y%d!=0) d--;
  return d;
}
```

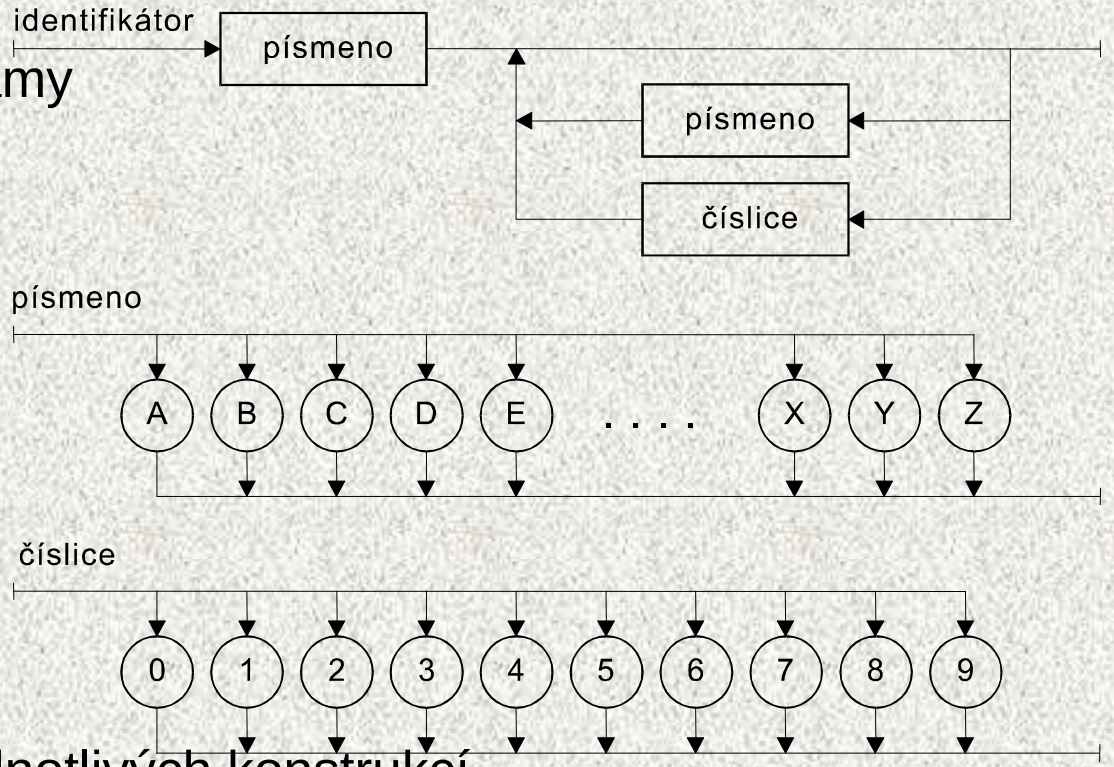
int .. celé číslo, proměnné x,y,d a výsledek budou typu **int**

d--; příkaz, který sníží hodnotu uloženou v proměnné d o jedničku

x%d … zbytek po dělení čísla x číslem d,
!= … není rovno,
|| … nebo

Vlastnosti programovacích jazyků

- Syntaxe
  - souhrn pravidel udávajících přípustné tvary dílčích konstrukcí a celého programu
  - syntaktické diagramy



- Sémantika
  - udává význam jednotlivých konstrukcí

Rozšířená BNF

- Rozšířená Backus-Naurova forma – EBNF
- Příklad: identifikátor

  identifikátor = písmeno {písmeno | číslice}

  písmeno = 'A' | 'B' | 'C' | 'D' | ... | 'X' | 'Y' | 'Z'

  číslice = '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '9'
- Neterminály:

  identifikátor, písmeno, číslice
- Terminály:

  ‘A’, ‘B’, ...
- Význam metasymbolů:

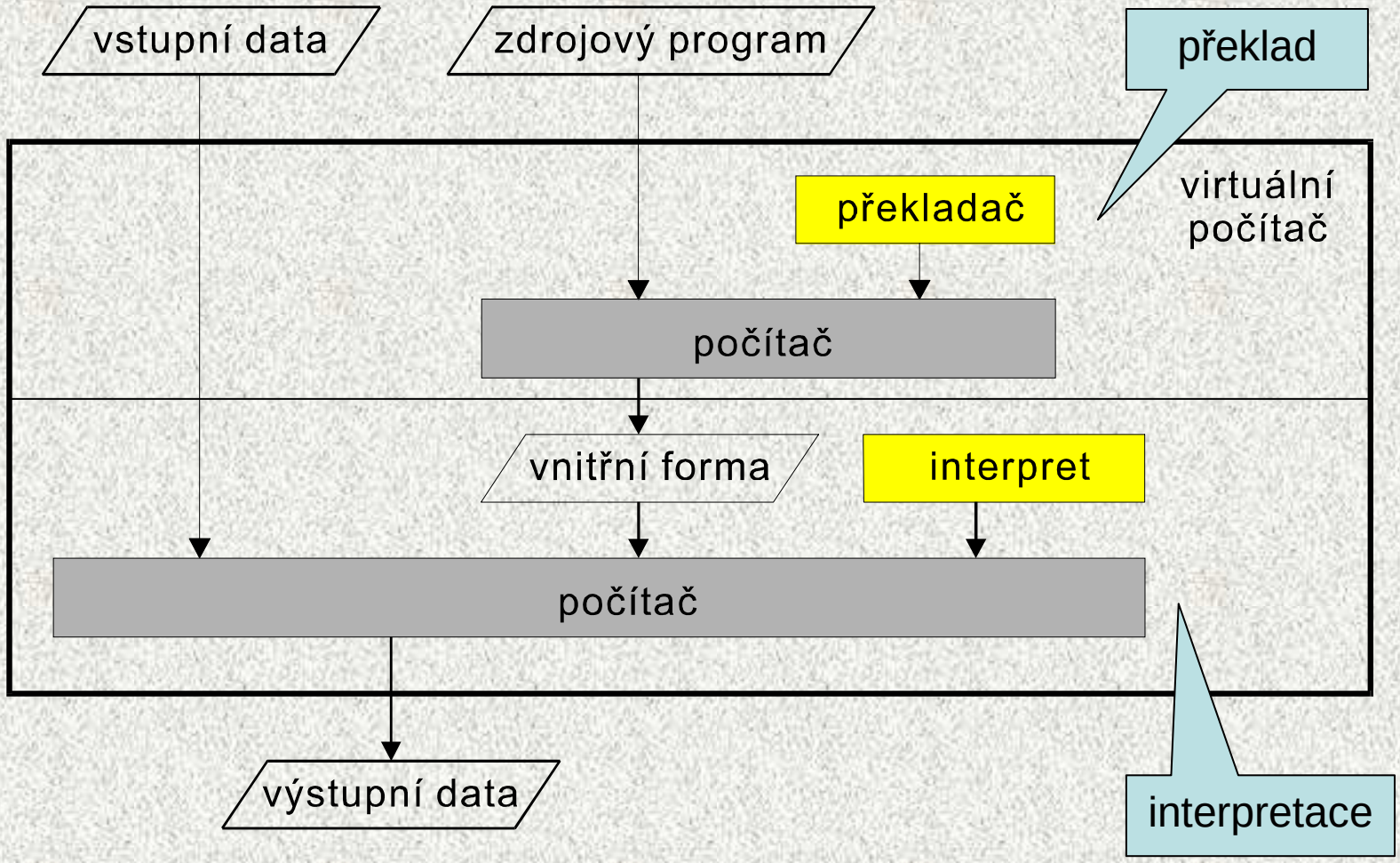
  {*x*} žádný nebo několik výskytů *x*

  x | *y* *x* nebo *y*

  [*x*] žádný nebo jeden výskyt *x*

Implementace programovacích jazyků

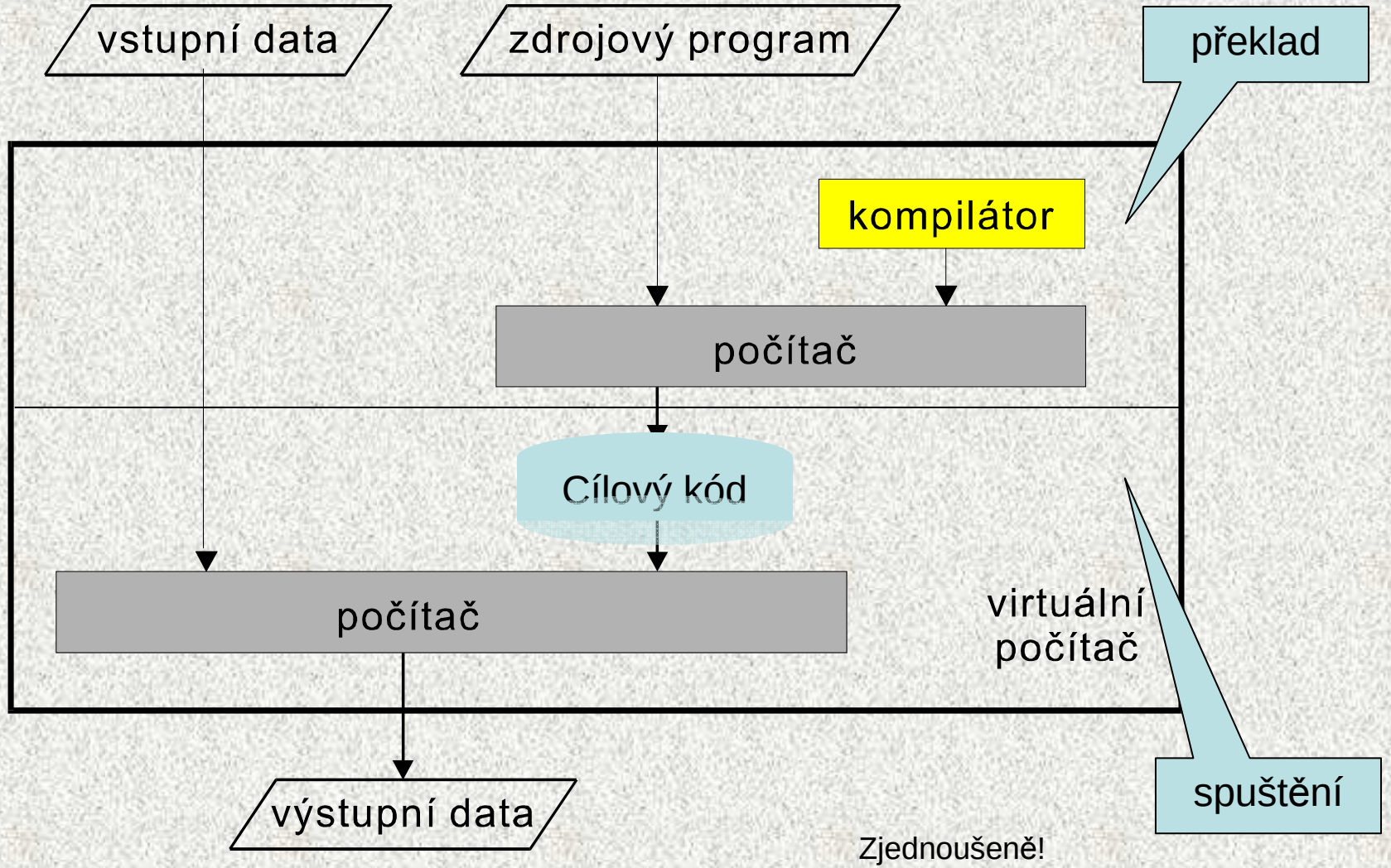
- Interpretační metoda:



Zjednoušeně!

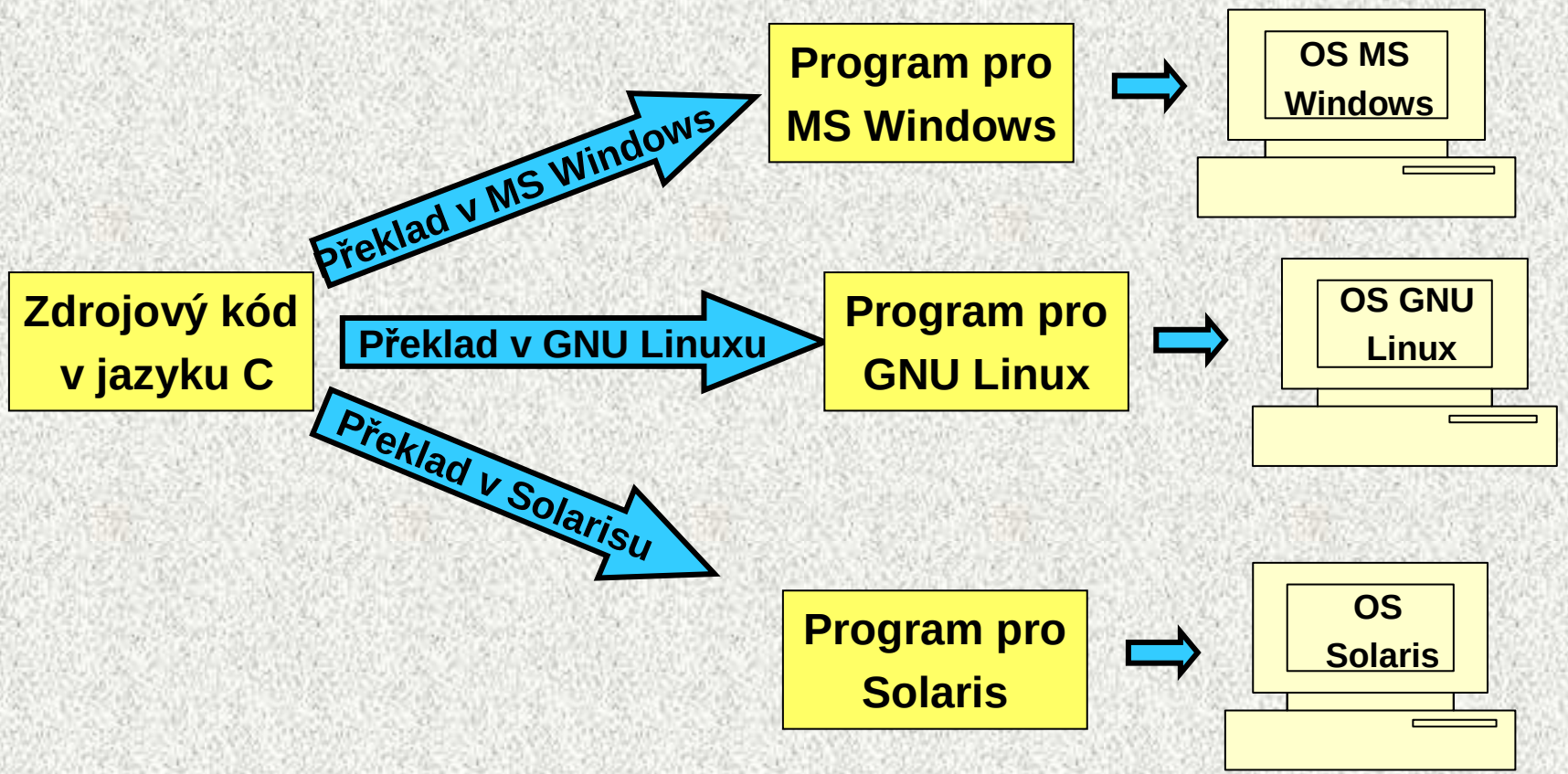
Implementace programovacích jazyků

- Kompilační metoda:

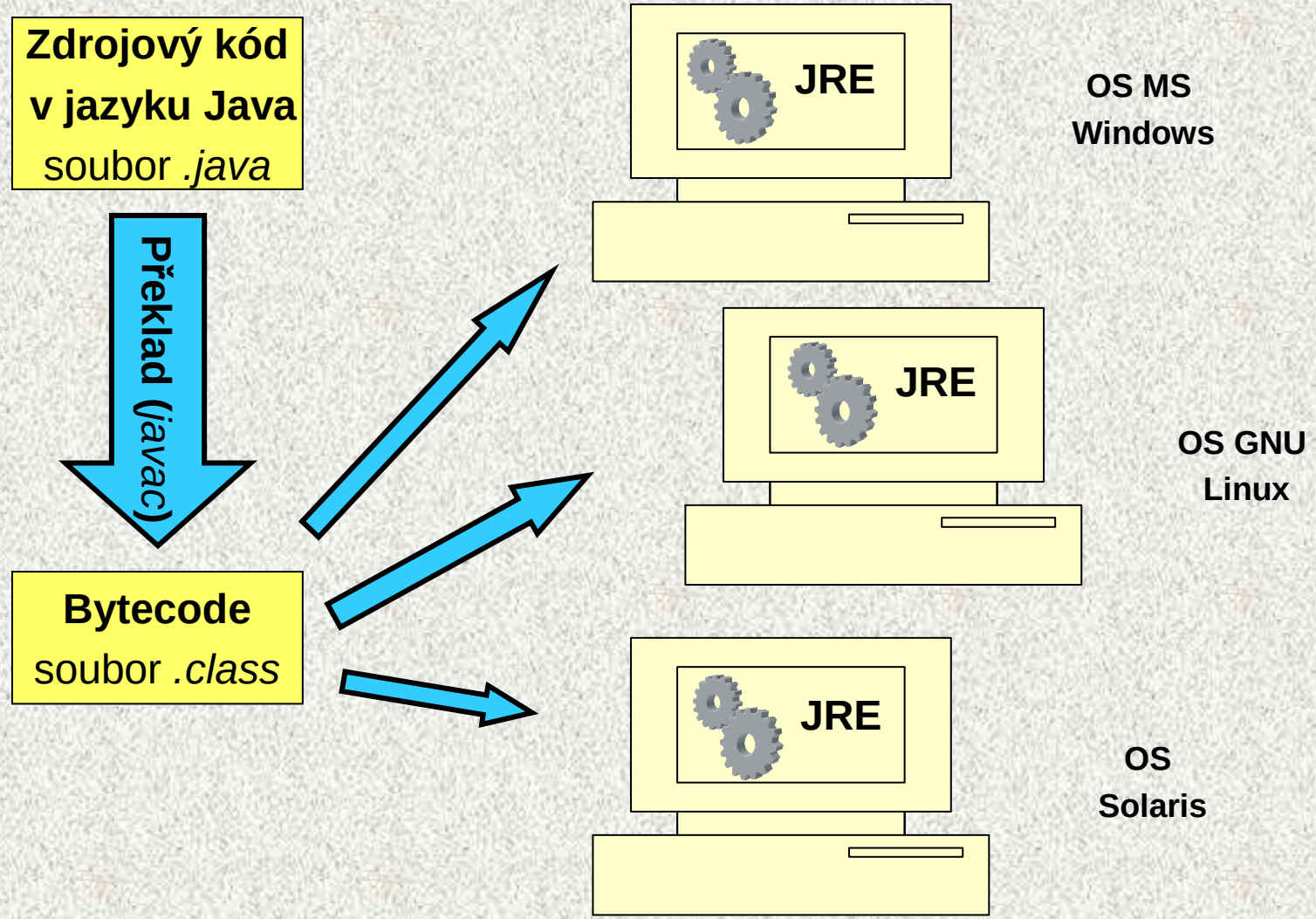


Zjednoušeně!

Kompilační metoda - jazyk C, C++



Interpretační metoda - jazyk Java



Reprezentace dat v počítači

```
00000000: CA FE BA BE 00 00 00 31|00 19 0A 00 05 00 15 07 | Ęţş╛■■■1■■■■■■■■
00000010: 00 16 0A 00 02 00 15 07|00 17 07 00 18 01 00 06 | ■■■■■■■■■■■■■■■■
00000020: 3C 69 6E 69 74 3E 01 00|03 28 29 56 01 00 04 43 | <init>■■■()V■■■C
00000030: 6F 64 65 01 00 0F 4C 69|6E 65 4E 75 6D 62 65 72 | ode■■■LineNumber
00000040: 54 61 62 6C 65 01 00 12|4C 6F 63 61 6C 56 61 72 | Table■■■LocalVar
00000050: 69 61 62 6C 65 54 61 62|6C 65 01 00 04 74 68 69 | iableTable■■■thi
00000060: 73 01 00 1A 4C 64 61 74|61 62 61 7A 65 72 69 64 | s■■■Ldatabazerid
00000070: 69 63 75 32 2F 53 74 61|72 74 47 55 49 3B 01 00 | icu2/StartGUI;■■
00000080: 04 6D 61 69 6E 01 00 16|28 5B 4C 6A 61 76 61 2F | ■main■■■([Ljava/
00000090: 6C 61 6E 67 2F 53 74 72|69 6E 67 3B 29 56 01 00 | lang/String;)V■■
000000A0: 04 61 72 67 73 01 00 13|5B 4C 6A 61 76 61 2F 6C | ■args■■■[Ljava/l
000000B0: 61 6E 67 2F 53 74 72 69|6E 67 3B 01 00 03 67 75 | ang/String;■■■gu
000000C0: 69 01 00 15 4C 64 61 74|61 62 61 7A 65 72 69 64 | i■■■Ldatabazerid
```

Reprezentace dat

- bit (z anglického **bi**nary digi**t** - dvojková číslice; angl. bit = drobek, kousek) je základní a současně nejmenší jednotkou informace
- byte, někdy též bajt - 8 bitů

  Násobky:
  - 1kB, kilobyte, 1000 bytů
  - 1KB = 1KiB, kibibyte(někdy nesprávně kilobyte), 2^{10}=1024 bytů
  - obdobně mebibyte, gibibyte, tebibyte – bi od binární

Reprezentace dat v počítači

- V počítači není u každé datové položky určeno, jaký typ dat uchovává

Příklad:

`0x41` binárně 01000001 může být číslo 65, nebo znak A

Reprezentace celých čísel

- číselné soustavy - polyadické

  Desítková soustava

  $138{,}24 = \mathbf{1}*10^{\mathbf{2}}+\mathbf{3}*10^{\mathbf{1}}+\mathbf{8}*10^{\mathbf{0}}+\mathbf{2}*10^{\mathbf{-1}}+\mathbf{4}*10^{\mathbf{-2}}$

  $=1*100+3*10+8+2*0{,}1+4*0{,}01$

  počítače používají dvojkovou soustavu

  $11010{,}01 = \mathbf{1}*2^{\mathbf{4}}+\mathbf{1}*2^{\mathbf{3}}+\mathbf{0}*2^{\mathbf{2}}+\mathbf{1}*2^{\mathbf{1}}+\mathbf{0}*2^{\mathbf{0}}+\mathbf{0}*2^{\mathbf{-1}}+\mathbf{1}*2^{\mathbf{-2}} =$

  $1*16+1*8+0*4+1*2+0+0*1/2+1*1/4= 26{,}25$

Celá čísla v Javě – typ int

int je reprezentováno 32 bity

nejmenší číslo *1000...000* = -2^{31} = -2147483648

největší číslo *0111...111* = 2^{31} -1 = -2147483647

Poznámka:

Pozor na důsledky počítání (v doplňkovém kódu).:

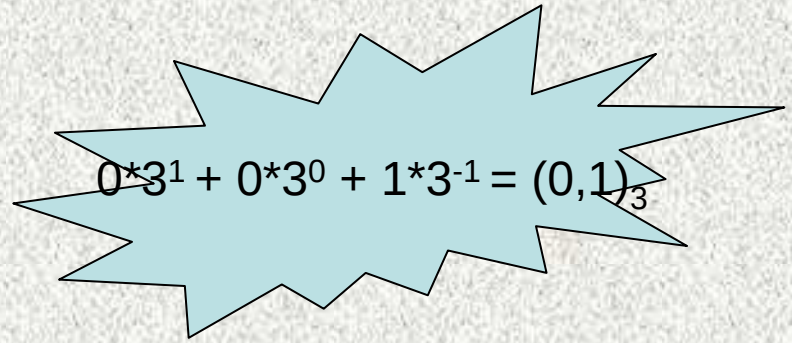
2 000 000 000 + 1 000 000 000 = -1 294 967 296

Math.abs(-2147483648) = -2147483648 !! záporné číslo

Nepřesnost v zobrazení čísel

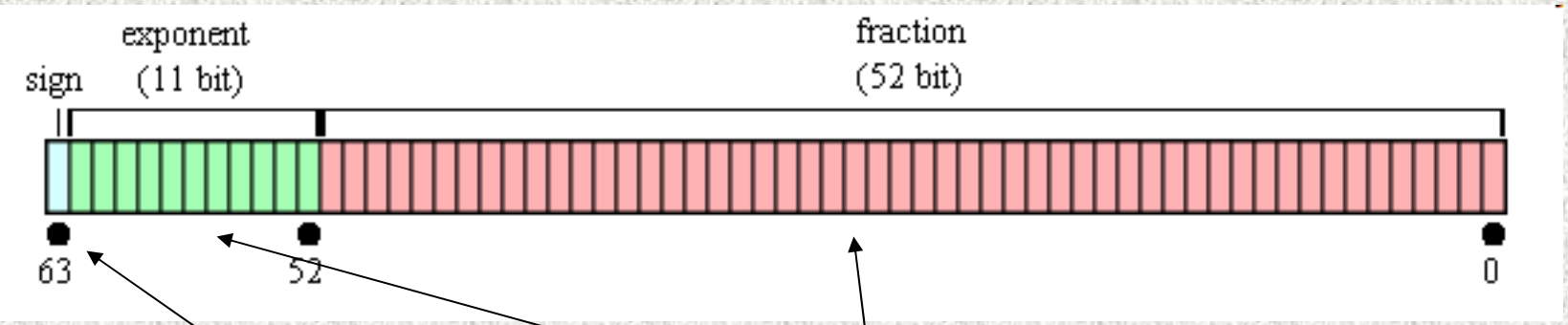
- pokusme se v dekadické soustavě zapsat číslo 1/3 (jedna třetina)
  - 0,3
  - přesněji 0,33
  - lépe 0,333
  - ...
  - 0,33333333333333333333333333333333

- jelikož máme omezený paměťový prostor, nikdy neuložíme číslo 1/3 přesně (Jak by číslo 1/3 vypadalo v trojkové soustavě?)



Necelá čísla v Javě – typ double

double je reprezentován 64 bity, norma IEEE 754



znaménkový bit (s), exponent, mantisa

nejmenší čísla v normalizovaném tvaru (v abs. hodnotě)

$\pm 2^{-1022} \approx \pm 2.2250738585072020 \times 10^{-308}$

největší čísla

$\pm(1-(1/2)^{53})2^{1024} \approx \pm 1.7976931348623157 \times 10^{308}$

Pozor!! 1E100+50000000000000L = 1.0E100

Nepřesnost v zobrazení čísel

Nepřesnosti způsobují

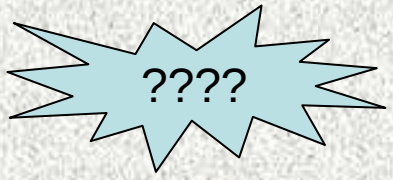
1. Iracionální čísla (e , π, $\sqrt{2}$,….
2. Čísla, jež mají v dané soustavě periodický rozvoj (1/3 ve dvojkové či dekadické soustavě, 1/10 ve dvojkové)
3. Čísla, která mají příliš dlouhý zápis:

   double

   - 1bit znaménko – dvě možnosti, +,-
   - 11 bitů exponent – 2048 možností
   - 52 bitů – 4 503 599 627 370 496 možností, tedy asi 4,5 biliardy

   - není možné v typu double přesně uložit čísla se zápisem delším než 52 bitů!

   - čím větší exponent tím větší „mezery“ mezi sousedními aproximacemi!!!



Model reprezentace reálných čísel

Reálná čísla se zobrazují jako aproximace daných rozsahem paměťového místa
Reálná čísla se zobrazují ve tvaru:

X= mantisa*základ $^{\text{exponent}}$ = m*z $^{\text{exponent}}$

Mantisa musí být normalizována:

0,1<=m<1, důvod: jednoznačnost zobrazení

Model:

- délka mantisy 3 pozice + znaménko
- délka exponentu 2 pozice + znaménko

Příklad

X= 77.5 = +0.775*z^{+02}

- zakódujeme znaménko +=0, - = 1
- z je 10, 16, 2,..

+|775 +|02

1|775 1|02

Model reprezentace reálných čísel

Maximální zobrazitelné kladné číslo

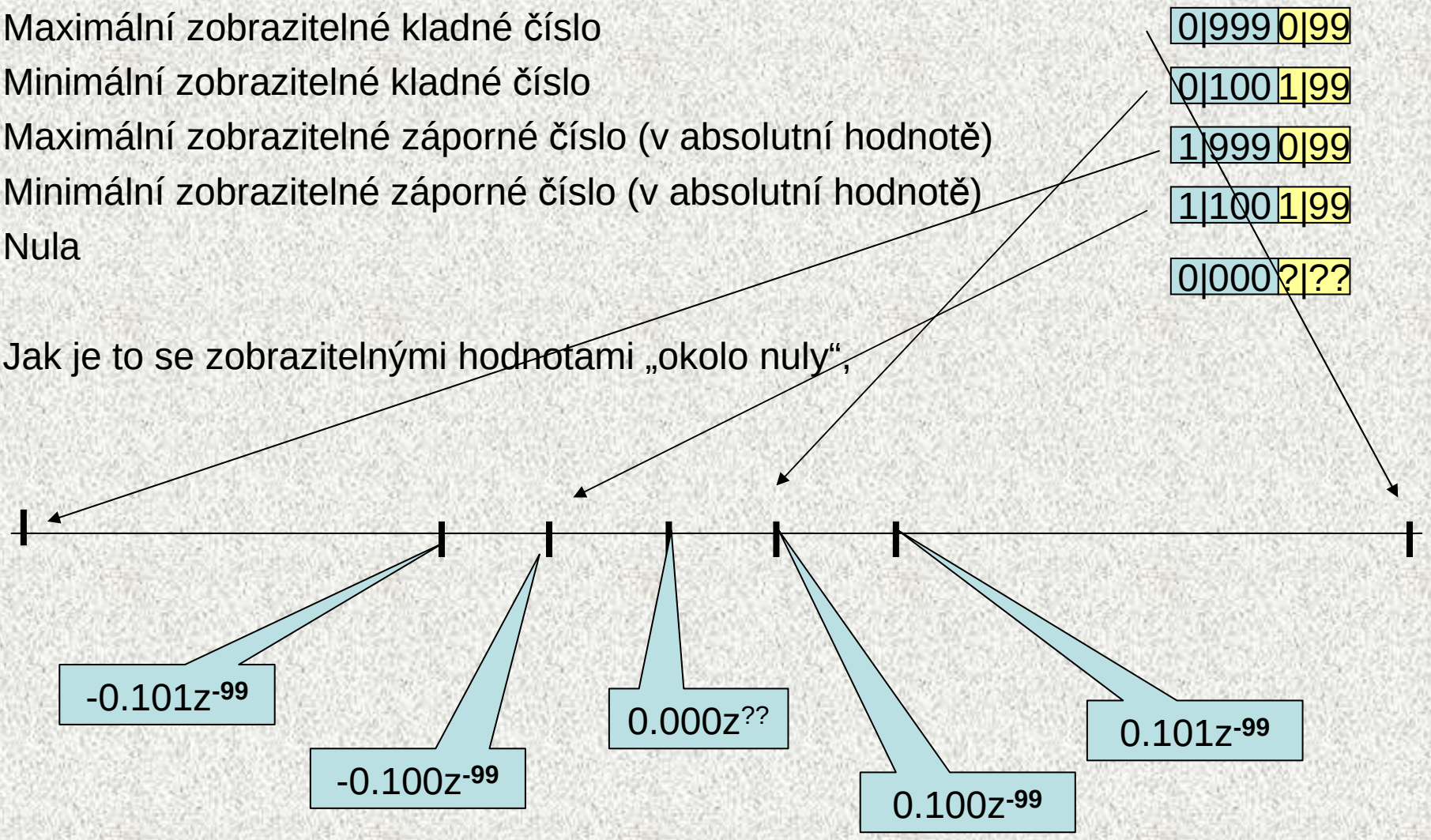
Minimální zobrazitelné kladné číslo

Maximální zobrazitelné záporné číslo (v absolutní hodnotě)

Minimální zobrazitelné záporné číslo (v absolutní hodnotě)

Nula

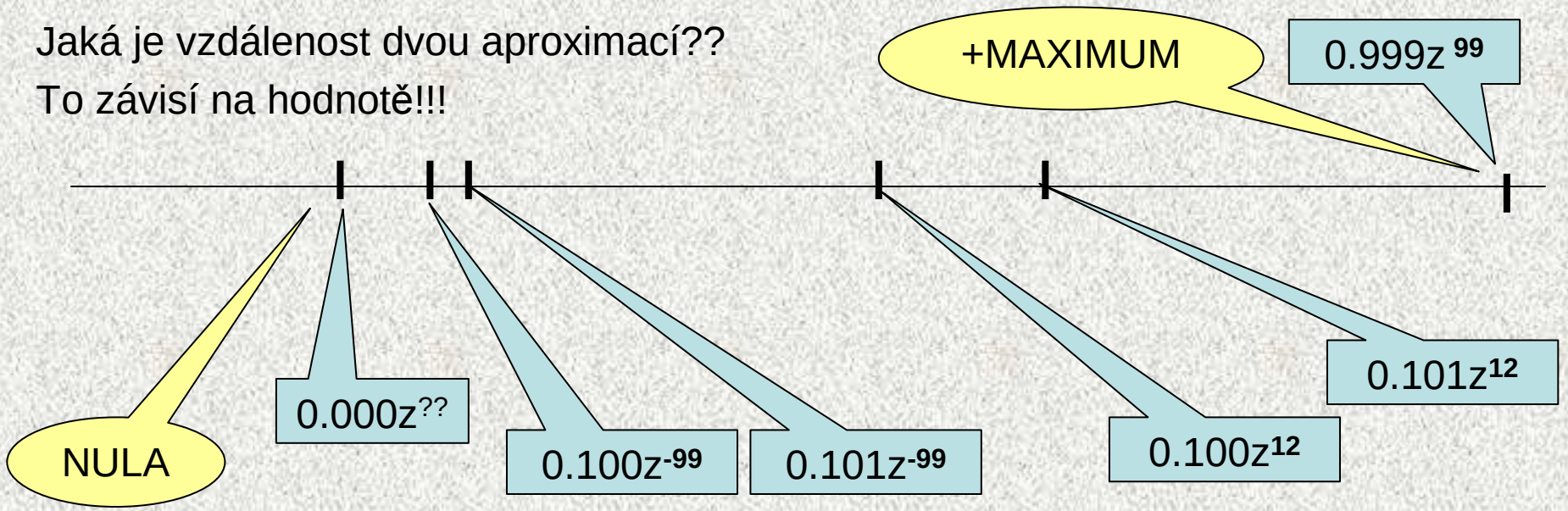
Jak je to se zobrazitelnými hodnotami „okolo nuly“,



Model reprezentace reálných čísel

Jaká je vzdálenost dvou aproximací??

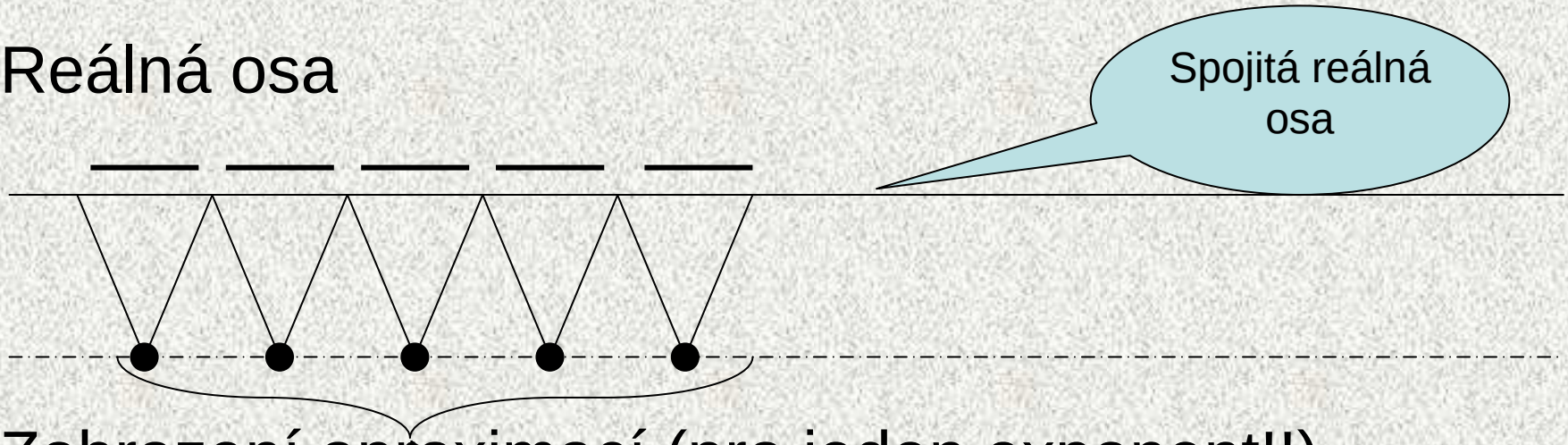
To závisí na hodnotě!!!



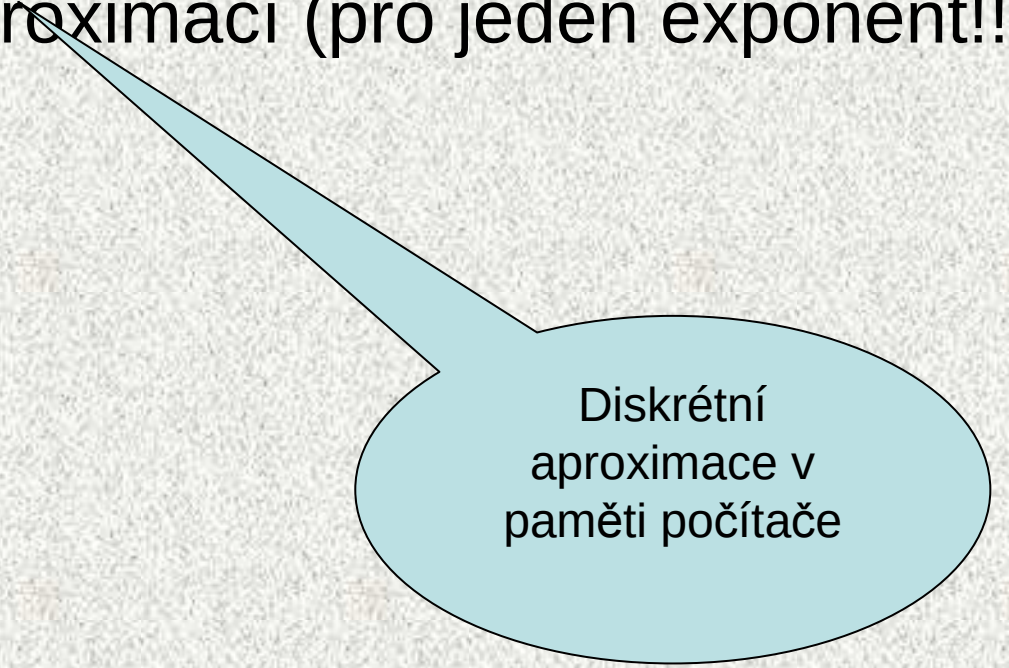
Aproximace reálných čísel: nejsou na číselné ose rovnoměrně rozložené!!

Reprezentace reálných čísel

Reálná osa



Zobrazení aproximací (pro jeden exponent!!)



Elektronická paměť

Součástka, zařízení nebo materiál, který umožní uložit obsah informace (zápis do paměti), uchovat ji po požadovanou dobu a znovu ji získat pro další použití (čtení paměti)

- Vnitřní paměť (primární)
- Vnější paměť

Vnitřní paměť

- Registry procesoru - několik (až desítky) registrů, ukládání operandů a výsledků aritmetických a logických operací, nejrychlejší paměť připojená k procesoru (stejně rychlá, jako procesor)
- Cache - pro urychlení komunikace s pamětí, rychlá statická paměť,u novějších procesorů velikost stovky kB az MB,více úrovní, přičemž číslo určuje vzdálenost od procesoru
  - L1 – typicky přímo na procesoru
  - L2 – například na destičce s procesorem (tzv. boxované procesory)
  - L3 – na základní desce
- Operační paměť RAM - pomalejší než procesor(stovky MHz), velikost stovky MB až GB, typicky dynamická paměť

Doplněk

Vnější paměť

- Sekundární paměti
  - Pevný disk, je na nich systém souborů (struktura adresářů), obsahuje obvykle statickou nebo dynamickou cache pro urychlení čtení/zápisu
- Terciární paměti
  - zařízení k zálohování dat, CD a DVD, Optické disky

Reprezentace znaků

- Různé OS různá kódování
  - sedmibitové ASCII (American Standard Code for Information Interchange), osmibitová Win1250, ISO 8859-2, kód s proměnnou délkou znaku - UTF8(UCS Transformation Format), ...
- Java – UCS2
  - Universal Character Set - univerzální znaková sada definována normou ISO 10646
  - programy v Javě provádí konverzi znaků mezi vnitřní formou a OS, většinou automaticky podle informací OS (Locale)

Doplněk

Reprezentace záporných celých čísel

- doplňkový kód – D(x)
  - předpokládejme reprezentaci čísel pomocí 8 bitů, lze reprezentovat $2^8 = 256$ čísel = r (rozsah)

$$D(x) = \begin{cases} x, \text{ pro } x<r/2 \\ r+x, \text{ pro } x>= r/2 \end{cases}$$

desítkově	**doplňkový kód**
0 – 127	*00000000 – 01111111*
128	nelze zobrazit na 8 bitů v d.k.
-128	D(-128) = -128+256 = 128 = *10000000*
-1	D(-1) = -1+256 = 257 = *11111111*

Doplněk

Reprezentace záporných celých čísel

- **doplňkový kód – D(x)**
  - **první bit je znaménkový**
  - **pro počítání se zápornými čísly není potřeba převádět čísla z doplňkového kódu**

převod záporného čísla na kladné:

10111011 ..... bitový obraz(8 bitů), 187= -69+256

01000100 ..... bitový doplněk

+1 ..... přičtení jedničky

01000101 ..... bitový obraz(8 bitů) 64+4+1 = 69

Reprezentace záporných celých čísel

- převod kladného na záporné:

01000101 ..... bitový obraz(8 bitů), 69

10111010 ..... bitový doplněk

+1 ..... přičtení jedničky

10111010 ..... bitový obraz(8 bitů)

přetečení: 127 + 1: *01111111 + 00000001 =10000000* → -128

Doplněk