# Seminár z algoritmizácie a programovania 1

**Martin Bobák**
**Ústav informatiky**
**Slovenská akadémia vied**

# Obsah prednášky

1. **Reprezentácia informácie**
2. **Zhrnutie**

Spätná väzba:
https://forms.gle/iKbuLdF6xDtNSEDP8

# Reprezentácia informácie

# Reprezentácia informácie

Digitalizácia = reprezentácia informácie pomocou čísla

Zvyčajný postup:

- niečo (zvuk, obrázok, text…) sa konvertuje nejako (algoritmom) na čísla
- tieto čísla sú uložené, spracované, preposlané
- následne sú čísla pretransformované do pôvodného formátu

Hodnoty z reálneho sveta (zvuky, obrázky…)

- čo najpresnejšie ich zmeriame
- prevedieme ich číselné hodnoty

# Analógová a digitálna reprezentácia

Analógový: "ručičkový"

- spojité hodnoty
- žiadne (diskrétne) skoky
- svet, ktorý vnímame je prevažne analógový

Digitálny: "diskrétne hodnoty"

- konečné množstvo hodnôt
- zmena vedie k prechodu z jednej diskrétnej hodnoty na inú diskrétnu hodnotu
- hodnoty sú reprezentované ako čísla

# Prevodník

Zariadenie, ktoré konvertuje z jednej reprezentácie do inej

- mikrofón
- reproduktor, slúchadlá
- fotoaparát, kamera, skener
- tlačiareň, obrazovka
- klávesnica, myška, dotyková obrazovka
- …

Zvyčajne dôjde k strate počas konverzie

- kópia nie je tak dobrá ako originál

# Kódovanie zvuku

- meranie tónu a jeho intenzity dosť často a presne, aby bolo možné daný zvuk rekonštruovať
- človek vie rozpoznať zvuky ~ 20 Hz – 20 kHz
  - merať zvuk s dvojnásobnou presnosťou (frekvenciou) je dostatočné

CD

- 44,100 meraní za sekundu
- presnosť (kvantovanie t.j. rozdelenie úrovní signálu) 65 536 úrovní (=$2^{16}$ diskrétnych hodnôt)
- stereo zvuk = dve merania
- 44,100 x 2 x 16 bit/meranie
  = 1,411,200 b/sec = 176,400 B/sec ~ 10.6 MB/min (bez kompresie)

MP3

- kompresia zvuku (odstránenie nepočuteľných zvukov)
- ~ 1MB/min

# Kódovanie obrazu

- obrázok sa rozdelí na mriežku malých obdĺžnikov ("pixelov")
- každý z nich má priradené číslo reprezentujúce jeho farbu
- menšie obdĺžniky = plynulejší prechod medzi farbami, presnejšia reprezentácia

# Kódovanie písmen

- číslo reprezentuje jeden znak
  - rôzne abecedy
  - špeciálne symboly
  - viacero spôsobov kódovanie -> treba štandard

ASCII (American Standard Code for Information Interchange)

**ASCII Code Chart**

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | NUL | SOH | STX | ETX | EOT | ENQ | ACK | BEL | BS | HT | LF | VT | FF | CR | SO | SI |
| 1 | DLE | DC1 | DC2 | DC3 | DC4 | NAK | SYN | ETB | CAN | EM | SUB | ESC | FS | GS | RS | US |
| 2 | | ! | " | # | $ | % | & | ' | ( | ) | * | + | , | - | . | / |
| 3 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | : | ; | < | = | > | ? |
| 4 | @ | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
| 5 | P | Q | R | S | T | U | V | W | X | Y | Z | [ | \ | ] | ^ | _ |
| 6 | ` | a | b | c | d | e | f | g | h | i | j | k | l | m | n | o |
| 7 | p | q | r | s | t | u | v | w | x | y | z | { | \| | } | ~ | DEL |

# Zhrnutie

- počet hodnôt a počet znakov kódovania je previazaný
  - algoritmizovateľný prevod
- interpretácia závisí na kontexte
  - bez vedomosti o aké kódovanie sa jedná, nie je možné určiť pôvodnú informáciu
  - čo je 39502???

# Bit a bajt

**Bit:**

- najmenšia jednotka informácie
- nadobúda dve hodnoty (true/false, áno/nie, on/off, m/ž…)
- reprezentovaný ako 0 a 1
  - dostatočné na zistenie pôvodnej hodnoty
  - jedna cifra s dvomi možnými hodnotami
  - binárna cifra (ang. binary digit = bit)
- v informatike sa používa pretože je ľahké vyrobiť rýchle, spoľahlivé a malé zariadenia s dvomi stavmi
  - vysoké/nízke napätie, (ne)prechádza prúd (čipy)
  - elektrický náboj (ne) prítomný (RAM, flash)
  - magnetizácia (disky)
  - (ne) odráža sa laser (CD, DVD)
- každý typ informácie je uložený a spracovaný v počítači ako bity

"on/off model"
- hw – transistor
- sw – bit

**Bajt:**

- 8 bitov, ktoré sú spracované ako celok (tak ako číslo 67327 v desiatkovej sústave)
- pamäť = pole bytov

# Binárne čísla

- ako desiatkové, ale majú iba dve číslice: 0 a 1
- každé binárne číslo je reprezentované ako mocnina 2 (1, 2, 4, 8, 16…)
  - nie mocnina 10 (1, 10, 100, 1000…)
- binárne počítanie:
  - 0 1
    - jedna binárna cifra reprezentuje 2 hodnoty
  - 00 01 10 11
    - dve binárne cifry reprezentujú 4 hodnoty
- binárne číslo je suma mocnín 2
  - $11011 = 1 \times 16 + 1 \times 8 + 0 \times 4 + 1 \times 2 + 1 \times 1$
    $= 1 \times 2^4 + 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 = 27$

# Binárna aritmetika

- podobne ako desiatková sústava

Sčítanie:

0 + 0 = 0

1 + 0 = 1

0 + 1 = 1

1 + 1 = 10

- podobne odčítanie, násobenie delenie

# Bajty

- jeden bajt = 8 bitov
  - na moderných počítačoch základná jednotka spracovania a ukladania informácií
  - vieme kódovať $2^8$ = 256 rôznych hodnôt (čísla, písmená, špeciálne znaky… -> ASCII tabuľka)
- dva bajty = 16 bitov
  - $2^{16}$ = 65 536 hodnôt
  - veľké čísla, znaky "väčšej abecedy" (napr. Unicode)
- štyri bajty = 32 bitov
  - $2^{32}$ = 4 294 967 296 hodnôt
  - ešte väčšie čísla, desatinné čísla, adresovanie pamäte (RAM môže mať max 4 GB)
- …
  - v súčasnosti 64-bitové celé čísla (8 bajtov) a adresy v pamäti (RAM môže mať 16 EB)
  - $2^{64}$ = 18 446 744 073 709 551 616 hodnôt

# Interpretácia bitov záleží na kontexte

1 bajt môže byť:

- využije sa 1 bit (napr. databáza M/Ž) a 7 bitmi sa plytvá
- 8 bitov sa používa na ukladanie čísiel z intervalu <0, 255>
- znak napr. 'W', '+', '7'…
- časť znaku z väčšej abecedy
- časť väčšieho čísla (2, 4, 8 bajtových)
- časť obrázka, zvukovej stopy
- časť príkazov počítačového programu
  - príkazy sú bity uskladnené "vedľa" dát
  - rôzne architektúry majú rôzne reprezentácie (pc, mobil, server…)
- časť adresy do pamäte počítača

# Mocniny dvojky a desiatky

1 bit = 2 možnosti

2 bity = 4 možnosti

3 bity = 8 možností

…

n bitov = $2^n$

pridanie 1 bitu zdvojnásobí rozsah

| Multiple-byte units | | | | | | | |
|---|---|---|---|---|---|---|---|
| Decimal | | | Binary | | | | |
| Value | Metric | | Value | IEC | | JEDEC | |
| 1000 | kB | **kilobyte** | 1024 | KiB | kibibyte | KB | kilobyte |
| $1000^2$ | MB | megabyte | $1024^2$ | MiB | mebibyte | MB | megabyte |
| $1000^3$ | GB | gigabyte | $1024^3$ | GiB | gibibyte | GB | gigabyte |
| $1000^4$ | TB | terabyte | $1024^4$ | TiB | tebibyte | – | |
| $1000^5$ | PB | petabyte | $1024^5$ | PiB | pebibyte | – | |
| $1000^6$ | EB | exabyte | $1024^6$ | EiB | exbibyte | – | |
| $1000^7$ | ZB | zettabyte | $1024^7$ | ZiB | zebibyte | – | |
| $1000^8$ | YB | yottabyte | $1024^8$ | YiB | yobibyte | – | |
| Orders of magnitude of data | | | | | | | |

Zdroj:
https://en.wikipedia.org/wiki/Kilobyte

$2^{10} = 1\,024 \sim 1\,000 = 1K = 10^3$

$2^{20} = 1\,048\,576 \sim 1\,000\,000 = 1M = 10^6$

$2^{30} = 1\,073\,741\,824 \sim 1\,000\,000\,000 = 1G = 10^9$

# Hexadecimálne čísla

- binárne čísla sú dlhé
  - často pracujeme s 32 alebo 64 bitovými číslami
- 4 bity v jednom čísle (zapísané ako mocnina 16)
  - efektívnejšia reprezentácia tej istej informácie

- pre čísla 10 – 15 sa používajú symboly A B C D E F

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 0000 | 1 | 0001 | 2 | 0010 | 3 | 0011 |
| 4 | 0100 | 5 | 0101 | 6 | 0110 | 7 | 0111 |
| 8 | 1000 | 9 | 1001 | A | 1010 | B | 1011 |
| C | 1100 | D | 1101 | E | 1110 | F | 1111 |

V jazyku C majú hexadecimálne čísla prefix **0x** a binárne čísla majú prefix **0b** (napr. 0xf5 je 0b11110101)

# Farba

**RGB model**

- farba je vyjadrená ako kombinácia červenej (R), zelenej (G) a modrej (B)
  - žltá = R+G, purpurová = R+B, tyrkysová = B+G, biela = R+G+B
  - tri bajty (=24 bitov) -> čierna = (0, 0, 0), oranžová = (255, 165, 0)
  - obrázok rozdelený do mriežky 1024 x 1024 zaberá 3 145 728 bajtov (= 1024×1024×3)
  - rozlíšenie obrázka je počet pixelov, ktorými je reprezentovaný (1024×1024 = 1 048 576 pixelov ~ 1 megapixel)
- farba pixelu bitmapy je vyjadrená ako tri čísla z intervalu <0, 255> reprezentujúce intenzity jednotlivých RGB komponentov
  - často vyjadrená ako hexadecimálne číslo, jednotlivé RGB komponenty sú vyjadrené nezávisle (000000 – čierna, FFFFFF – biela)
- tlačiarne používajú model CMY[K] -> cyan-magenta-yellow[-black]

# Zhrnutie

- na konci je každá informácia (uložená v počítači) reprezentovaná ako postupnosť bitov
  - bit nadobúda dve hodnoty (0, 1)
- komplikovanejšie údaje sú reprezentované ako skupina bitov
  - číslo, písmeno, slovo, obrázok, zvuk, program…
  - interpretácia skupiny bitov záleží na kontexte (formát)
  - reprezentácia údajov je ľubovoľná, štandardy charakterizujú ako daný typ údajov reprezentovať v danom formáte
- počet cifier určuje koľko rôznych objektov (hodnôt) vieme nimi reprezentovať
  - $2^{16}$, $2^{32}$, $10^{6}$

# Veľké čísla

# Motivácia

- 32 bitov dokáže reprezentovať 4 294 967 295 čísiel
- 64 bitov dokáže reprezentovať 18 446 744 073 709 551 615 čísiel

- nie vždy je to postačujúce (šifrovanie, vedecké výpočty…)

# Reprezentácia

- pole znakov konštantnej dĺžky
  - v špecifických prípadoch môžeme typ poľa zmeniť
  - konštantná dĺžka zjednoduší operácie nad takými číslami (prenos)

| | 0 | 1 | 2 | 3 | 4 | 5 | ... | 99 |
|---|---|---|---|---|---|---|---|---|
| reťazec | '1' | '2' | '3' | '4' | '\0' | ?? | ?? | ?? |
| ASCII kód | 49 | 50 | 51 | 52 | 0 | ?? | ?? | ?? |
| Dlhé číslo | 4 | 3 | 2 | 1 | 0 | 0 | 0 | 0 |

# Operácie

```
#define MAX_DLZKA_CISLA 1000

char *nacitaj(const char *str)
{
    int i, n = strlen(str);
    if (n > MAX_DLZKA_CISLA)
        return NULL; // prilis dlhy retazec
    char *dlhe = calloc(MAX_DLZKA_CISLA, 1);
    for (i = 0; i < n; i++)
        dlhe[i] = str[n - i - 1] - '0';
    return dlhe;
}
```

# Operácie

```
#define MAX_DLZKA_CISLA 1000

char *retazec(const char *dlhe)
{
    char str[MAX_DLZKA_CISLA];
    int i, j = 0;
    for (i = MAX_DLZKA_CISLA - 1; i > 0; i--)
        if (dlhe[i] > 0)
            break;
    while (i >= 0)
        str[j++] = dlhe[i--] + '0';
    str[j] = 0;
    return strdup(str);
}
```

# Operácie

```
#define MAX_DLZKA_CISLA 1000

int porovnaj(const char *a, const char *b) {
    int i;
    for (i = MAX_DLZKA_CISLA - 1; i > 0; i--)
        if (a[i] > 0 || b[i] > 0)
            break;
    while (i > 0 && a[i] == b[i])
        i--;
    if (a[i] < b[i])
        return -1;
    if (a[i] > b[i])
        return 1;
    return 0; }
```

# Operácie

```
#define MAX_DLZKA_CISLA 1000

void pripocitaj(char *a, const char *b) // a += b
{
    int i, prenos = 0;
    for (i = 0; i < MAX_DLZKA_CISLA; i++)
    {
        prenos += a[i] + b[i] ;
        a[i] = prenos % 10 ;
        prenos /= 10;
    }
}
```

# Operácie

```
#define MAX_DLZKA_CISLA 1000

char *plus_vyraz(char *str) {
    char *x = dlhecislo("0");
    char num[MAX_DLZKA_CISLA];
    int i, k;
    for (i = k = 0; str[i]; i++) {
        if (str[i] >= '0' && str[i] <= '9')
            num[k++] = str[i];
        if (k > 0 && (str[i] == '+' || str[i+1] == 0)) {
            num[k] = 0; // ukonci predchadzajuce cislo
            char *y = dlhecislo(num);
            pripocitaj(x, y); // x += y
            k = 0;}
    }
    return x;}
```

# Operácie

```
#define MAX_DLZKA_CISLA 1000

void nasob_int(char *a, int num) // a *= num
{
    int i, prenos = 0;
    for (i = 0; i < MAX_DLZKA_CISLA; i++)
    {
        prenos += num * a[i] ;
        a[i] = prenos % 10 ;
        prenos /= 10;
    }
}
```

# Operácie

```
#define MAX_DLZKA_CISLA 1000

char *binarne(const char *str)
{
    char *bin = dlhecislo("1"), *x = dlhecislo("0");
    int i, n = strlen(str);
    for (i = n - 1; i >= 0; i--)
    {
        if (str[i] == '1')
            pripocitaj(x, bin); // x += bin
        nasob_int(bin, 2); // bin *= 2
    }
    return x;
}
```

# Operácie

```
#define MAX_DLZKA_CISLA 1000

void prinasob(char *a, const char *b){ // a *= b
    char x[2 * MAX_DLZKA_CISLA] = { 0 }; // vysledok
    int i, j, prenos;
    for (j = 0; j < MAX_DLZKA_CISLA; j++) {
        for ( prenos = i = 0; i < MAX_DLZKA_CISLA; i++ ) {
            prenos += a[i] * b[j] + x[i+j] ;
            x[i+j] = prenos % 10 ;
            prenos /= 10;
        }
    }
    for (i = 0; i < MAX_DLZKA_CISLA; i++)
        a[i] = x[i];
}
```

# Námety na seminárnu prácu

- ďalšie operácie (rozdiel, podiel, mocnina, integrál, derivácia...)
- prehľad a porovnanie knižníc na prácu s veľkými číslami
  - podpora reálnych čísiel
  - iné programovacie jazyky
- aplikácie veľkých čísiel
  - vedecké výpočty
  - bankovníctvo
  - šifrovanie
- iné

# Zdroje

[1] Bou Ezzeddine, A., Tvarožek, J. Programovanie v jazyku C v riešených príkladoch (1). Bratislava: Vydavateľstvo Spektrum STU, 2018. 233 p. ISBN 978-80-227-4865-0.

# Ďakujem vám za pozornosť!

Spätná väzba:
https://forms.gle/iKbuLdF6xDtNSEDP8