STU FIIT

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Základy procedurálneho programovania 1

Úlohy na precvičenie

17. 10. 2016

zimný semester 2016/2017

Načítaj celý riadok (aj medzery) do reťazca

- Použijeme funkciu **fgets**
- Program si napíšeme ako cvičenie ...

Úloha: Načítaj binárne číslo na vstupe do intu

- Na vstupe je binárne číslo zapísané ako reťazec znakov 0 a 1, načítaj ho do premennej typu **int**.
- Program si napíšeme ako cvičenie ...

STU FIIT

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Základy procedurálneho programovania 1

Dlhé čísla

17. 10. 2016

zimný semester 2016/2017

Dlhé čísla reprezentujeme ako reťazec číslic

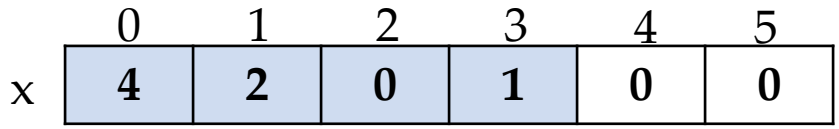
- Výpočet
  22 222 222 222 222 222 222 222 +
  77 777 777 777 777 777 777 777 =
  premenné akého dátového typu môžem použiť, aby bol výsledok správny?

- Aké číslo je zapísané v reťazci x?

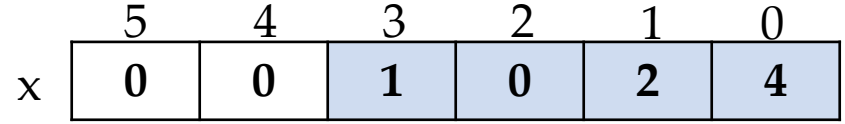
	0	1	2	3	4	5	6	7	8	9
x	**49**	**48**	**50**	**52**	**0**	**55**	**-109**	**113**	**3**	**0**

Dlhé číslo ako pole cifier

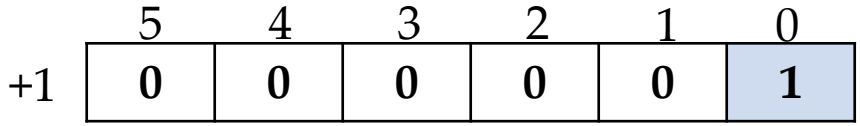
- Číslo 1024



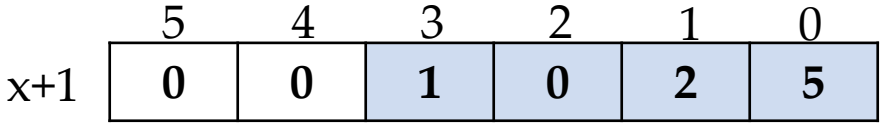
- Pre jednoduchšie (ľudské) uvažovanie si budem pole predstavovať opačne:



- Pripočítame



- Výsledok



Úloha: prevod reťazec ⇔ pole cifier

- Program pre prevod číselného reťazca na pole cifier.

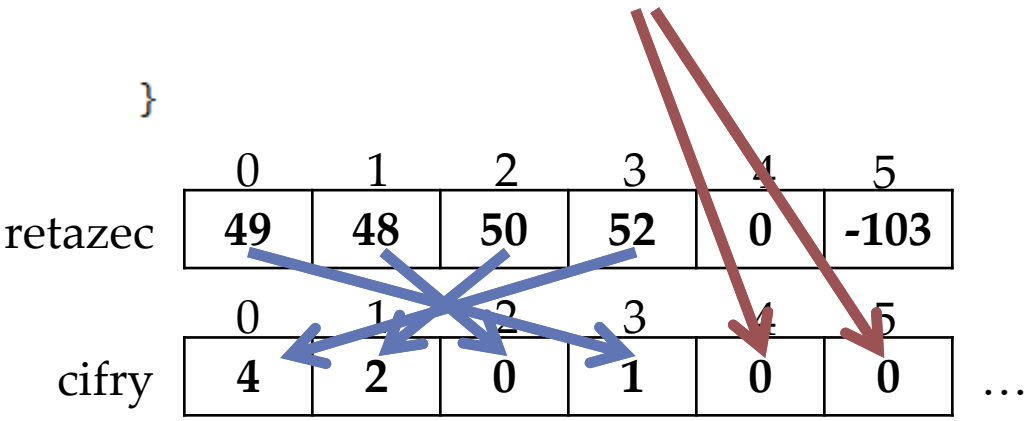
- Potrebujem:
  - Miesto na reťazec
    char str[100]
  - Miesto na cifry
    char x[100]
  - Načítať reťazec
  - Previesť na cifry
  - Kontrola správnosti
    - Previesť (naspäť) na reťazec
    - Vypísať

```
int main(void)
{
    char str[100]; // retazec
    char x[100]; // pole cifier

    scanf("%s", str);
    preved_na_cifry(str, x, 100);

    char tmp[100]; // retazec
    preved_na_retazec(x, 100, tmp);
    printf("%s\n", tmp);
    return 0;
}
```

Úloha: prevod reťazec ⇔ pole cifier

- Funkcia preved_na_cifry prevedie reťazec čísel **retazec** na pole (s dĺžkou **maxlen**) cifier **cifry**.

```
void preved_na_cifry(char *retazec, char *cifry, int maxlen)
{
    int i, len = strlen(retazec);
    for (i = len-1; i >= 0; i--)
        cifry[len-1-i] = retazec[i]-'0';
}
```




```
int main(void)
{
    char str[100]; // retazec
    char x[100]; // pole cifier

    scanf("%s", str);
    preved_na_cifry(str, x, 100);

    char tmp[100]; // retazec
    preved_na_retazec(x, 100, tmp);
    printf("%s\n", tmp);
    return 0;
}
```

Úloha: prevod reťazec ⇔ pole cifier

- Funkcia preved_na_cifry prevedie reťazec čísel **retazec** na pole (s dĺžkou **maxlen**) cifier **cifry**.

```
void preved_na_cifry(char *retazec, char *cifry, int maxlen)
{
    int i, len = strlen(retazec);
    for (i = len-1; i >= 0; i--)
        cifry[len-1-i] = retazec[i]-'0';
    while (len-1-i < maxlen)
    {
        cifry[len-1-i] = 0;
        i--;
    }
}
```

len-1-i je len komplikovaný výpočet postupnosti 0, 1, ... Nahradíme obyčajným počítadlom **j**

- Jednoduchšia verzia

```
void preved_na_cifry(char *retazec, char *cifry, int maxlen)
{
    int i, len = strlen(retazec), j = 0;
    for (i = len-1; i >= 0; i--)
        cifry[j++] = retazec[i]-'0';
    while (j < maxlen)
        cifry[j++] = 0;
}
```

Úloha: prevod reťazec ⇔ pole cifier

- Funkcia preved_na_retazec prevedie pole cifier **cifry** (s dĺžkou **maxlen**) na reťazec čísel **retazec**.

```
void preved_na_retazec(char *cifry, int maxlen, char *retazec)
{
    int i, j;
    for (i = maxlen-1; i >= 0; i--)
        if (cifry[i] > 0)
            break;
    for (j = i; j >= 0; j--)
        retazec[i-j] = cifry[j] + '0';
    retazec[i+1] = 0;
}
```

Cyklus nájde najvyššiu cifru (i)

Cyklus skopíruje od najvyššej cifry (i) všetky cifry (j = i,…,0) do reťazca

```
int main(void)
{
    char str[100]; // retazec
    char x[100]; // pole cifier

    scanf("%s", str);
    preved_na_cifry(str, x, 100);

    char tmp[100]; // retazec
    preved_na_retazec(x, 100, tmp);
    printf("%s\n", tmp);
    return 0;
}
```

Úloha: Deliteľnosť tromi

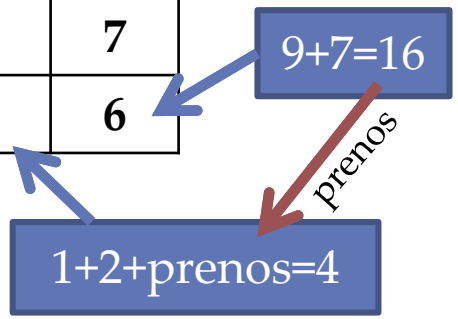
- Napíš v jazyku C zdrojový kód funkcie **del3**, ktorá zistí či dlhé číslo zapísane v reťazci je deliteľné tromi. Vysvetli aké má funkcia vstupné argumenty a návratovú hodnotu.

```
int del3(char *cifry, int maxlen)
{
    int i, cifsucet = 0;
    for (i = 0; i < maxlen; i++)
        cifsucet += cifry[i];
    return cifsucet % 3 == 0;
}
```

Úloha: Sčítanie dvoch dlhých čísel

- Ako sčitujeme (dve) čísla na papieri?
- Napíšeme pod seba, sčitujeme od najnižších rádov, ak je súčet cifier väčší ako 9, tak výsledná cifra bude zvyšok po delení 10 (napr. pre 13 to bude 3) a do ďalšieho rádu si prenesieme 1 navyše.

0	0	0	0	1	9
0	0	0	0	2	7
				4	6



- Program napíšeme spolu...

Úloha: Sčítanie dvoch dlhých čísel

- Program:

```
// scitanie pripocita a = a + b
void pripocitaj(char* a, char* b, int maxlen)
{
    int n,zvysok;
    for(n = 0; n < maxlen; n++){
        a[n] += b[n];
        while(a[n] >= 10){
            a[n] -= 10;
            a[n+1] += 1;
        }
    }
}
```

Úloha: Násobenie dlhých čísel

- Ako násobíme (dve) čísla na papieri?
- Napíšeme pod seba, násobíme od najnižších rádov.
- Výpočet pre jeden rád: jeden rád v druhom čísle zodpovedá jednej cifre, a teda vynásobíme prvé číslo jednou cifrou, výsledok násobenia pripočítame k výsledku.
- Vyššie rády počítame rovnako, výsledok posunieme o príslušný počet miest: desiatky o jedno miesto, stovky o dve miesta, ...

Úloha: Násobenie dlhých čísel

- Ako násobíme (dve) čísla na papieri?

0	0	0	9	8	7
0	0	0	6	5	4
			36	32	28
		3	9	4	8
		45	40	35	
	4	9	3	5	
	54	48	42		
5	9	2	2		
5	13	14	14	9	8
6	4	5	4	9	8

- Program napíšeme spolu...

Úloha: Násobenie dlhých čísel

- Program napíšeme spolu...

```
// nasobenie: a = a * b
void prinasobenie(char* a, char* b, int maxlen)
{
    int n, i;
    char x[200]; // vysledok
    for(n = 0; n < maxlen*2; n++)
        x[n] = 0;
    for(n = 0; n < maxlen; n++)
    {
        for(i = 0; i < maxlen; i++)
        {
            x[i+n] += a[i] * b[n];
            x[i+n+1] += x[i+n] / 10;
            x[i+n] %= 10;
        }
    }
    for(n = 0; n < maxlen; n++)
        a[n] = x[n];
}
```

STU FIIT

SLOVENSKÁ TECHNICKÁ UNIVERZITA V BRATISLAVE
FAKULTA INFORMATIKY A INFORMAČNÝCH TECHNOLÓGIÍ

Základy procedurálneho programovania 1

Úlohy z predchádzajúcich priebežných testov

17. 10. 2016

zimný semester 2016/2017

- Napíš program v jazyku C, ktorý načíta dve stociferné čísla a vypíše ich súčet. Čísla sú na vstupe ako reťazce znakov, pričom nezačínajú cifrou nula. Čísla sú na vstupe oddelené medzerou. (4 body)

- Napíš program v jazyku C, ktorý zistí koľko je na vstupe čísel. Na vstupe sú čísla oddelené medzerami, na výstup vypíše jedno číslo – počet všetkých čísel na vstupe. (2 body)

- Napíš program v jazyku C, ktorý zistí, koľko samohlások je na vstupe. Na výstup vypíše jedno číslo, počet samohlások, ktoré sa nachádzajú na vstupe. Na vstupe sú ASCII reťazce dĺžky najviac 20 znakov. (4 body)

- Napíš program v jazyku C, ktorý načíta stociferné číslo a jednu číslicu a vypíše ich súčin. Čísla sú na vstupe ako reťazce znakov, pričom nezačínajú cifrou nula. Čísla sú na vstupe oddelené medzerou. (4 body)

- Na vstupe sú čísla zapísané v rozličných číselných sústavách oddelené medzerami, niektoré sú binárne čísla – čísla zložené len z číslic 0 alebo 1. Napíš program v jazyku C, ktorý zistí koľko je na vstupe binárnych čísel a na výstup vypíše ich počet. (3 body)

- Napíš program v jazyku C, ktorý načíta binárne číslo na vstupe, a na výstup vypíše jeho hodnotu v desiatkovej sústave. Napr. 10 je 2, 11 je 3, 100 je 4, 101 je 5, 110 je 6, 111 je 7. Ak v binárnom čísle je i-ta (i sa počíta od nuly pre rád jednotiek) číslica 1, znamená to, že desiatkové číslo je o 2^i väčšie, teda napr. číslo 111 je o 4 väčšie ako 011. Na vstupe je binárne číslo ako ASCII reťazec dĺžky najviac 20 znakov. (3 body)