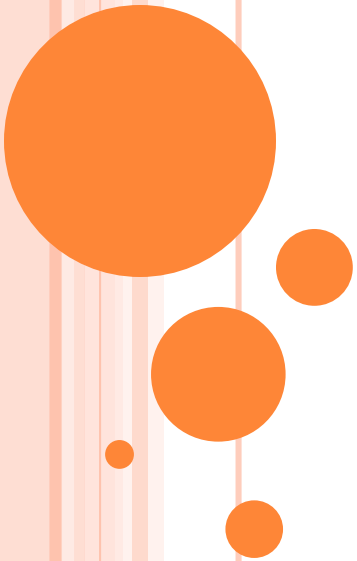

Programovacie jazyky pre vstavané systémy

Cvičenie 5

Náplň cvičenia

1. Smerníky a typové konverzie.
2. Endianita.
3. Súbory.
4. Úlohy na prácu so vstupno-výstupnými parametrami, s reťazcami a textovými súbormi.

Otázky

- Čo je to smerníková aritmetika?
- Čo je to endianita?
- Aký spôsob odovzdávania parametrov podporuje jazyk C?
- Môže funkcia v jazyku C vrátiť pole?
- Akými spôsobmi je možné vrátiť z funkcie výsledok výpočtu v jazyku C?
- Akým spôsobom je možné zadefinovať v jazyku C symbolickú konštantu?

Konverzie medzi smerníkmi

- Ľubovoľný smerník na objekt môže byť pretypovaný na ľubovoľný iný typ smerníka na objekt.
- Vo všeobecnosti je však bezpečné pretypovávať len smerníky z väčších dátových typov na menšie (napr. int* na char*).
- Zistite, čo robí nasledujúci kód.

```c
#include <stdlib.h>
#include <stdio.h>
#include <limits.h>

int main(int argc, char* argv[]) {
    int i = INT_MAX, *pi = &i;
    short *ps = (short *)pi;
    int *pi_2 = (int *)ps;

    printf("*pi = %d\n", *pi);
    printf("*ps = %d, *(ps + 1) = %d\n", *ps, *(ps + 1));
    printf("*pi_2 = %d\n", *pi_2);

    printf("pi = %p\n", pi);
    printf("ps = %p, ps + 1 = %p\n", ps, ps + 1);
    printf("pi_2 = %p\n", pi_2);

    return 0;
}
```

Jednoduchý test endianity

- Vysvetlite, ako pracuje nasledujúci kód:

```c
#include <stdlib.h>
#include <stdio.h>

_Bool jeLittleEndian() {
    int cislo = 1;
    return ((char*)&cislo)[0] == (char)1;
}

int main(int argc, char** argv) {
    if (jeLittleEndian())
        printf("Poradie bajtov je little endian.\n");
    else
        printf("Poradie bajtov je big endian.\n");

    return 0;
}
```

Argumenty programu

- Každý program môžeme spustiť s argumentmi, ktoré môže bližšie špecifikovať, čo má program robiť.
- Argumenty, s ktorými sa program spustí, sa odovzdávajú do funkcie main ako pole reťazcov.

```
1  #include <stdlib.h>
2  #include <stdio.h>
3
4  int main(int argc, char* argv[]) {
5      for (int i = 0; i < argc; i++) {
6          printf("%s\n", argv[i]);
7      }
8  }
```

- Prepíšte predchádzajúci kód pomocou smerníkovej aritmetiky.

SÚBORY

- Práca so súbormi:
  - otvorenie súboru:

    FILE *f = fopen(nazov_suboru, mod);
  - uzavretie súboru:

    fclose(f);
  - odstránenie súboru:

    remove(nazov_suboru);
  - funkcie pre prácu so súborom:
    - binárne súbory: fread(), fwrite()
    - textové súbory: fscanf(), fprintf(), fgetc(), fputc(), fgets(), fputs()
    - **test konca súboru**: feof(), konštanta **EOF** – **preskúmajte rozdiel**
    - ostatné funkcie: http://en.cppreference.com/w/c/io

Práca so súbormi <stdio.h>

Mód	Význam	Vysvetlenie	Ak súbor existuje	Ak súbor neexistuje
"r"	read	otvor súbor pre čítanie	číta od začiatku	otvorenie zlyhá
"w"	write	vytvor súbor pre zápis	vymaže	vytvorí nový
"a"	append	pripoj na koniec súboru	zapisuje na koniec	vytvorí nový
"r+"	read extended	otvor súbor pre čítanie/zápis	číta od začiatku	chyba
"w+"	write extended	vytvor súbor pre čítanie/zápis	vymaže	vytvorí nový

- Ak chceme pracovať s binárnymi súbormi, k módu je nutné pripojiť reťazec "b".
- C11 – príznak "x" vo "w" a "w+" móde zabráni zmazaniu súboru, ak existuje.

Jednoduché úlohy

- Vytvorte jednoduché funkcie (navrhnite parametre a návratový typ), ktoré umožnia:
  - vymeniť obsah dvoch premenných typu int;
  - vypočítať korene kvadratickej rovnice nad oborom komplexných čísel (korene vráťte cez parametre);
  - vypočítať súčet dvoch komplexných polynómov (súčet vráťte cez parameter);
  - vrátiť minimum a jeho index v poli celých čísel odovzdanom ako parameter;
  - vrátiť súčasne medián a modus v poli celých čísel odovzdanom ako parameter;
  - naplniť pole char-ov náhodnými alfanumerickými znakmi tak, aby sa jednalo o reťazec,
  - zistiť, či dátový typ double je v pamäti ukladaný ako little endian alebo big endian.
- Vytvorte jednoduchý program, ktorý umožní:
  - vykonať frekvenčnú analýzu textu zadaného z klávesnice, t.j. z klávesnice sa načíta dlhý reťazec a následne sa zistia relatívne početnosti jednotlivých písmen anglickej abecedy a číslic.

Ďalšie úlohy

- Vytvorte nasledujúce funkcie:
  - **int* runningSum(int n, int data[n], int runningSum[n])** – naplní pole *runningSum* tak, aby sa na indexe *i* nachádzal súčet všetkých prvkov z poľa *data*, ktorých index je menší alebo rovný *i*, funkcia vráti adresu prvého prvku z poľa *runningSum*;
  - **char* intToStr(int number, int n, char* dest)** – uloží číslo *number* do reťazca *dest*, ktorý môže mať maximálne *n* znakov, funkcia vráti adresu prvého znaku v reťazci *dest*;
  - **int* intToArray(int number, int n, int array[n])** – rozloží číslo *number* na číslice a prvých *n* číslic uloží do *array*, funkcia vráti adresu prvého prvku z poľa *array*;
  - **_Bool areAnagrams(const char* strA, const char* strB)** – funkcia zistí, či reťazce *strA* a *strB* sú anagramy, t. j. jeden reťazec je možné získať z druhého zmenou poradia znakov;
  - **double gaussIntegral(double a, double b)** – funkcia vypočíta metódou numerickej integrácie určitý integrál z funkcie e^(-x^2) na intervale <*a*, *b*>;
  - **int* merge(int m, int sortedArrayA[m], int n, int sortedArrayB[n], int destArray[m+n])** – funkcia spojí utriedené polia *sortedArrayA* a *sortedArrayB* do poľa *destArray* tak, aby bolo pole *destArray* utriedené, funkcia vráti adresu prvého prvku z poľa *destArray*.
- Vytvorte program, ktorý umožní:
  a) sčítať 2 celé čísla, ktoré môžu mať najviac *n* cifier (+ znamienko);
  b) odčítať 2 celé čísla, ktoré môžu mať najviac *n* cifier (+ znamienko);
  - parameter *n* bude predstavovať argument odovzdaný programu pri jeho spustení, pričom:
    - ak *n* je reťazec alebo číslo menšie ako 1, program vypíše chybové hlásenie,
    - ak *n* nie je definované, implicitne sa bude predpokladať hodnota 50.

Úlohy – textové súbory

- Vytvorte jednoduché programy, ktoré umožnia (prípadné chyby vypíšte na štandardný chybový výstup):
  - skopírovať súbor (názvy zdrojového a cieľového súboru budú vystupovať ako **argumenty programu**);
  - presunúť súbor (názvy zdrojového a cieľového súboru budú vystupovať ako **argumenty programu**);
  - náhodne poprehadzovať riadky v textovom súbore (názov súboru bude vystupovať ako **argument programu**);
  - nahradiť všetky výskyty daného reťazca iným reťazcom (názov súboru ako aj oba reťazce budú vystupovať ako **argumenty programu**);
  - odstrániť daný riadok v súbore (názov súboru ako aj číslo riadku budú vystupovať ako **argumenty programu**);
  - zašifrovať a dešifrovať textový súbor pomocou cézarovej šifry (názvy zdrojového a cieľového súboru ako aj posun budú vystupovať ako **argumenty programu**);
  - vykonať frekvenčnú analýzu textového súboru (názov analyzovaného súboru bude vystupovať ako **argument programu**);
  - vypočítať nasledujúce charakteristiky pre textový súbor (názov textového súboru bude vystupovať ako **argument programu**):
    - počet písmen, bielych znakov a celkový počet znakov,
    - počet riadkov,
    - počet slov,
    - najdlhšie a najkratšie slovo.

Úlohy – bonusové

- Do programu pre sčítavanie a odčítavanie celých čísel dorobte funkcie, pomocou ktorých bude možné:
  a) vynásobiť 2 celé čísla, ktoré môžu mať najviac *n* cifier (+ znamienko) (0.5 bodu);
  b) celočíselne vydeliť 2 celé čísla, ktoré môžu mať najviac *n* cifier (+ znamienko) (0.5 bodu);
  c) vypočítať zvyšok po celočíselnom delení 2 celých čísel, ktoré môžu mať najviac *n* cifier (+ znamienko) (0.5 bodu).
- Vytvorte program, ktorý z klávesnice načíta reťazec a zistí, či sa jedná o platné rodné číslo (0.5 bodu).
- Vytvorte funkciu **const char* longestSubstr(const char* strA, const char* strB, int* length)**, ktorá nájde najdlhší spoločný podreťazec reťazcov *strA* a *strB*, cez parameter *length* vráti jeho dĺžku a cez návratovú hodnotu vráti adresu prvého znaku tohto podreťazca v reťazci *strA*. Ak taký podreťazec neexistuje funkcia vráti hodnotu NULL (0.5 bodu).