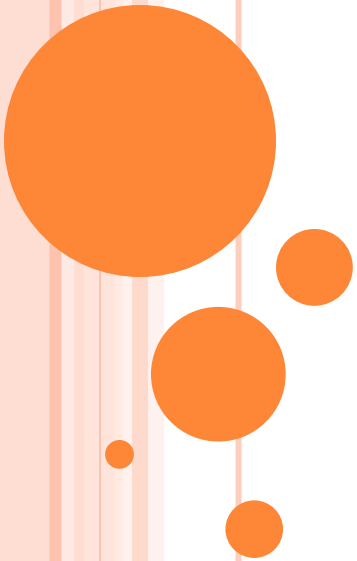

Programovacie jazyky pre vstavané systémy

Používateľom definované typy, súbory

Otázky z minulej prednášky

- Čo je to smerník?
- Aké operácie môžem robiť so smerníkmi?
- Ako spôsoby odovzdávania parametrov existujú?
- Akým spôsobom môžeme modifikovať argumenty funkcie vo vnútri funkcie?
- Ako sa odovzdávajú parametre typu pole do funkcie? Prečo?
- Aký je vzťah medzi poliami a smerníkmi?

Používateľom definované typy

- Mnohokrát si nevystačíme so vstavanými údajovými typmi a potrebujeme mať vlastný typ.
- Jazyk C umožňuje vytvoriť vlastný údajový typ ako agregáciu iných typov alebo vymenovaním prípustných hodnôt nového typu.
- Nový typ môže byť viditeľný na úrovni bloku alebo súboru.
- Nový typ viditeľný na úrovni súboru je prístupný len v súbore, v ktorom bol definovaný.
- Medzi používateľom definované typy patria:
  - štruktúra (**struct**)
  - zjednotenie (**union**)
  - vymenovaný typ (**enum**)

Dátový typ štruktúra (1)

- Deklarácia štruktúry:

  struct meno_struktury;

- Definícia štruktúry:

  struct [meno_struktury] {
  typ nazov_polozky1;
  typ nazov_polozky2;
  …
  } [zoznam_premennych];

- Ak sa neuvedie názov štruktúry, definujeme anonymnú štruktúru, ktorá sa nedá použiť neskôr.
- Zoznam premenných je voliteľný, ak chceme definovať premenné daného typu neskôr, deklarujeme ich nasledovne:

  struct meno_struktury identifikator1, identifikator2,...;

Dátový typ štruktúra (2)

- Štruktúra môže obsahovať ľubovoľné iné údajové typy (t.j. aj iné štruktúry) s výnimkou samej seba a poľa voliteľnej veľkosti (VLA).
- Štruktúra však môže obsahovať ukazovateľ na samú seba alebo VLA.
- C99 – ak má štruktúra aspoň jeden pomenovaný člen, posledným členom štruktúry môže byť pole neznámej veľkosti.
- Pri vytvorení premennej typu štruktúra sa jednotlivé položky alokujú v pamäti v takom poradí, v akom sú uvedené v štruktúre.
- Veľkosť štruktúry je **minimálne** taká, aký je **súčet** veľkostí jednotlivých položiek štruktúry.
- Operácie so štruktúrou
  - operátor priradenia – kopírovanie jednej štruktúry do inej
  - operátor prístupu k členom štruktúry (operátor . alebo ->)

Inicializácia štruktúry

- Ak je aspoň jeden prvok štruktúry inicializovaný, tak všetky neinicializované sa nastavia na default hodnoty.

```
struct osoba {
    char meno[20];
    int vek;
    struct osoba *otec;
    struct osoba *mama;
};

struct osoba otec = {"Milan"};
struct osoba mama = {.otec = NULL, NULL, .meno = "Anna", 46}; //C99
```

Ukážka práce so štruktúrou (1)

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

struct osoba {
    char meno[20];
    int vek;
    struct osoba *otec;
    struct osoba *mama;
};

void vypisOsobu(struct osoba *os) {
    printf("Osoba: %s, vek: %d", os->meno, (*os).vek);
    if ((*os).otec != NULL) {
        printf(", otec: %s", os->otec->meno);
    }
    if (os->mama != NULL) {
        printf(", mama: %s", (*(*os).mama).meno);
    }
    printf("\n");
}
```

Ukážka práce so štruktúrou (2)

```
23  int main(int argc, char** argv) {
24      struct osoba otec = {"Milan"};
25      struct osoba mama = {.otec = NULL, NULL, .meno = "Anna", 46};
26      struct osoba dieta1, dieta2;
27      struct osoba *p_otec = &otec;
28
29      strncpy(dieta1.meno, "Peter", 19);
30      dieta1.meno[19] = '\0';
31      dieta1.vek = 20;
32      dieta1.otec = p_otec;
33      dieta1.mama = &mama;
34
35      dieta2 = dieta1;
36      (*dieta2.mama).vek = 48;
37      dieta2.otec->vek = 50;
38
39      vypisOsobu(&dieta2);
40
41      return 0;
42  }
```

Dátový typ union (1)

- Deklarácia union-u:

  union meno_unionu;

- Definícia union-u:

  union [meno_unionu] {
  typ nazov_polozky1;
  typ nazov_polozky2;
  …
  } [zoznam_premennych];

- Zoznam premenných je voliteľný, ak chceme definovať premenné daného typu neskôr, deklarujeme ich nasledovne:

  union meno_unionu identifikator1, identifikator2,…;

Dátový typ union (2)

- Práca s union-om je podobná ako so štruktúrou.
- Hlavný rozdiel medzi štruktúrou a union-om je v tom, že kým štruktúra alokuje miesto pre všetky svoje zložky, tak union alokuje miesto len pre najväčšiu zložku.
- Všetky zložky zjednotenia sú uložené na tom istom pamäťovom mieste
- Veľkosť union-u je **minimálne** taká, aká je veľkosť jeho **najväčšieho** dátového typu.

Dátový typ union (3)

- Uvažujme nasledujúci príklad:

```
union slova {
    struct {
        unsigned char bajt1;
        unsigned char bajt2;
        unsigned char bajt3;
        unsigned char bajt4;
    } bajty;
   unsigned int dword;
} prem;

prem.dword = 0x12345678;
```

- Zistite, čo vypíše nasledujúci kód:

```
printf("%x\n", prem.bajty.bajt1);
```

Vymenovaný typ

- Definícia vymenovaného typu:

```
enum [meno_enumu] {
    HODNOTA1 [= INICIALIZATOR1];
    HODNOTA2 [= INICIALIZATOR2];
    ...
} [zoznam_premennych];
```

- Vymenovaný typ je celočíselný typ, pričom musí byť implementovaný tak, aby dokázal pokryť všetky vymenované hodnoty.
  - Na typ enum sa aplikujú implicitné konverzie a môže byť použitý s aritmetickými operátormi.
- Ukážka:

```
enum cis1 { JEDEN, DVA, TRI = 50, STYRI } c = JEDEN,
*d = &c;
enum cis2 { PAT = 49, SEST, SEDEM = 50, OSEM };

double c = PAT + 1.0 + JEDEN;
```

Pomenovanie dátových typov

- Ľubovoľný dátový typ je možné pomenovať menom:

  typedef T type_ident;
  type_ident D;

- Zavedieme tým nové meno type_ident, ktoré predstavuje alias pre typ T.
- Ukážka:

```
typedef int CELE_CISLO;
typedef struct polozka {
    CELE_CISLO data;
    struct polozka *dalsi;
} POLOZKA, *P_POLOZKA;

CELE_CISLO a;
POLOZKA pol, *p_pol1 = &pol;
P_POLOZKA p_pol2 = &pol;
```

Práca so súbormi <stdio.h>

- Ak chceme pracovať so súborom, musíme si zadefinovať premennú nasledujúceho typu:

  FILE *f;

- Otvorenie súboru – funkcia fopen:

  f = fopen(nazov_suboru, mod);

- Uzavretie súboru – funkcia fclose():

  fclose(f);

- Práca so súborom:
  - binárny: fread, fwrite, ftell, fgetpos, fsetpos, fseek, rewind
  - textový: fscanf, fprintf, fputs, fgets,...
  - feof, fflush,...
  - makrá stdin, stdout, stderr, EOF,...

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.