KATEDRA INFORMATIKY
PŘÍRODOVĚDECKÁ FAKULTA
UNIVERZITA PALACKÉHO V OLOMOUCI

Ukazatelé

8. cvičení

Jiří Zacpal

KMI/ZP1 – Základy programování 1

2. úkol

Vytvořte strukturu osoba, do které budete ukládat tyto informace o osobě: `jméno, příjmení, adresa bydliště, datum narození, telefon a e-mail`. Vytvořte pole, do kterého budete osoby ukládat. Napište tyto funkce:

```
vytvor_seznam(osoba s[])
```

- která všechny prvky pole nastaví na tzv. nulovou osobu (všechny údaje „ „ nebo 0).

```
osoba vytvor osobu(char jmeno[], char prijmeni[], char adresa[], char den, char mesic, int rok, char telefon[], char email[])
```

- která vytvoří novou osobu.

```
vloz(osoba s[],osoba o)
```

- která vloží osobu o do seznamu s.

```
Bool najdi_osobu(char kde[], char co[], osoba s[])
```

- která vrátí TRUE pokud v seznamu existuje osoba podle zadaných parametrů: v proměnné zadané parametrem kde (jméno, příjmení, adresa, ...), obsahuje řetězec, zadaný parametrem co.

```
osoba nejmladsi(osoba s[])
```

- která vrátí nejmladší osobu ze seznamu osob.

```
void tisk(osoba s[])
```

- která vytiskne seznam seznam_osob.

Všechny funkce otestujte a použijte ve funkci main. Kostru programu si můžete stáhnout z Vyuka\Jiří Zacpal\kmi_zp1\ukol_2_vzor.c

Ukazatel

- v literatuře se často vyskytuje také anglický ekvivalent pojmu ukazatel – **pointer**
- jde **datový typ**, který slouží k uložení adresy
- v jazyku C používáme ukazatele na konkrétní datový typ
  (pro uložení adresy, na které bude v paměti proměnná daného typu)

Deklarace ukazatele

- syntaxe:

```
typ *identifikátor = inic_adresa;
```

- příklady:

```
int *pI;
char *ret = "bla bla";

int pole[10];
int *p = pole;

long cislo, *pL1, *pL2 ;
```

Práce s ukazateli

Operátor adresy &

- slouží pro zjištění adresy dané proměnné
- příklady:

```
int i=2, *pi = &i, j;
pi = &j;

scanf("%i", &i);
```

Operátor dereference *

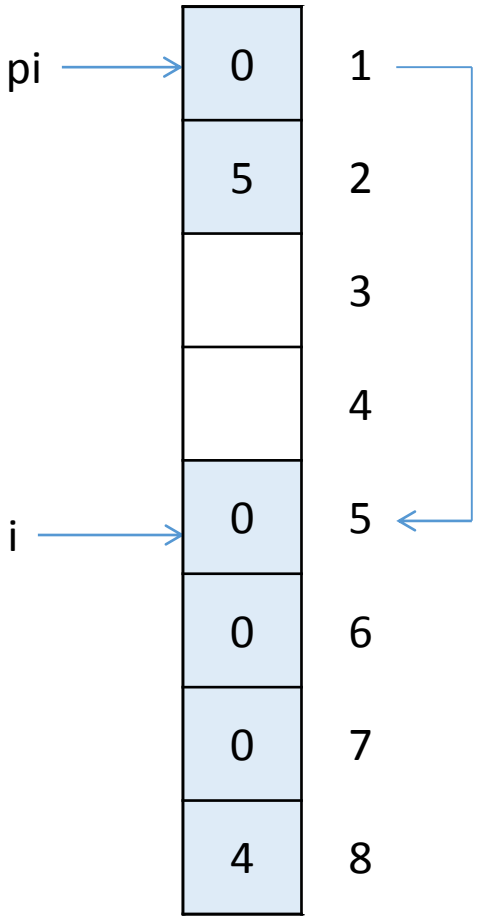
- slouží pro přístup k hodnotě na dané adrese
- příklady:

```
*pi = 3;
(*pi)++;

printf("%i\n", *pi);
```

Příklad 1

```
main()
{
  int i=2,*pi;
  pi = &i;
  *pi = 3;
  (*pi)++;
  printf(„pi=%i
  i=%i\n", *pi,i);
  printf("%i\n", pi);
}
```



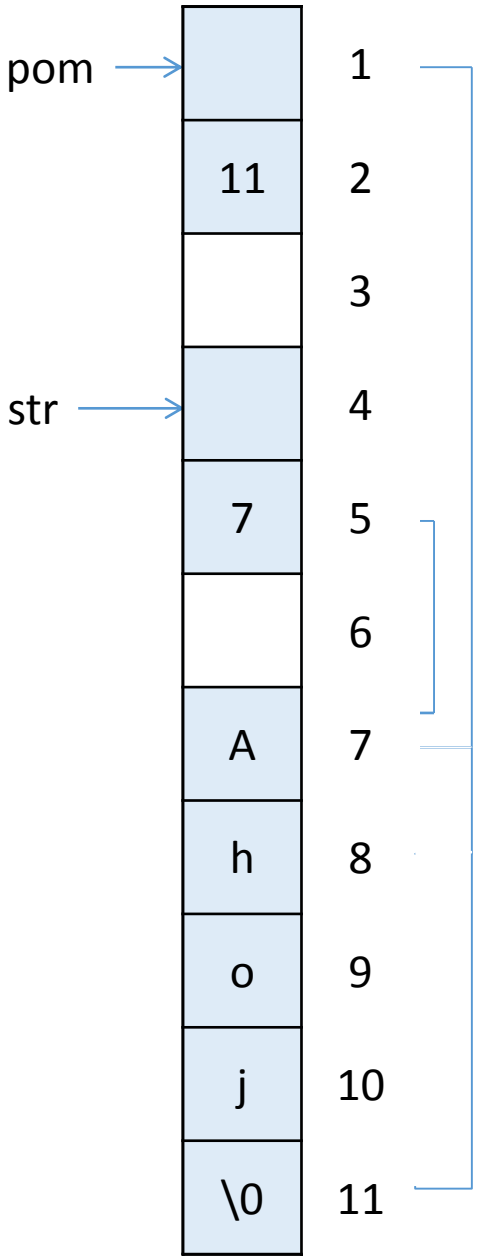
Pointerová aritmetika

- ukazatele reprezentují adresy, proto s nimi lze provádět pouze některé aritmetické operace:
- přičtení (resp. odečtení) celočíselného výrazu k ukazateli (resp. od ukazatele)
- rozdíl dvou ukazatelů stejného typu
- příklad:

```
char str[] = "Ahoj!";
char *pom = str;
while(*pom!='\0') {
   printf("%c\n", *pom);
   pom++;
}
printf("Délka řetězce je %i.\n", pom-
str);
```

Příklad 2

```
main()
{
char str[] = "Ahoj";
char *pom = str;
while(*pom!='\0')
{
printf("%c\n", *pom);
pom++;
}
printf("Délka řetězce
je %i.\n", pom-str);
}
```



Ukazatele a pole

- identifikátor pole se chová jako konstantní ukazatel na první prvek pole
- místo operátoru indexu lze pro přístup k prvkům pole využít pointerové aritmetiky a operátoru dereference
- příklad:

```
char str[] = "Ahoj Svete!", *pom;

int i;
for (i=0; i<(int)strlen(str);i++)
   printf("%c", str[i]);

pom = str;
while (*pom!='\0') {
   printf("%c", *pom);
   pom++;
}
```

Ukazatele na strukturu

- definujeme stejně jako ukazatel na jakýkoli jiný typ
- přístup ke členům pointeru na strukturu:

```
(*id_struktury).id_clenu
(id_struktury)->id_clenu
```

- příklad:

```
typedef struct {char Den, Mesic;}
Datum;
Datum den = {12,2};
Datum *narozen =&den;

narozen->Den = 25;
```

Struktura „uvnitř sebe sama“

- členem struktury je ukazatel na strukturu stejného typu
- příklady definice:

```
struct s1{int data; struct s1 *dalsi;};

typedef struct s1{
   int data;
   struct s1 *dalsi;
} seznam;
```

- příklady použití:

```
struct s1 a = {1,NULL};
seznam b = {2, &a};

b.dalsi->data = 0;
```