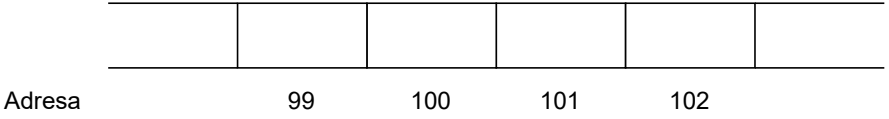
KATEDRA
INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

Adresy a ukazatele

Základy programování 2

Mgr. Markéta Trnečková, Ph.D.

Paměť



Paměť

Ukazatele

deklarace:

```
typ *jmeno;
```

Nulový ukazatel: `NULL`

výpis:

```
printf("%p", ptr);
```

operátory:

- adresy: &
- dereference: *

Ukazatele

Příklad

```
int a = 3, b, *ptr1, *ptr2;

/* ukazatel ptr1 ukazuje na promennou a  */
ptr1 = &a;

/* hodnota b je 5 (*ptr1 je rovna 3) */
b = *ptr1 + 2;

/* ptr1 a ptr2 ukazuji na stejne misto */
ptr2 = ptr1;

/* zmenime hodnotu na miste, kam ukazuje ptr2 */
*ptr2 = 5;

/* hodnota b bude 8 */
b = a + 3;
```

Scanf

Příklad

```
int a;

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

Co se zde děje?

Pointerová aritmetika

- **statické pole**: `int pole[] =` $\{1, 2, 3\}$
- pole je ukazatel na první prvek
- **konstantní pointer** - nelze mu přiřadit jinou hodnotu

Pointerová aritmetika

- K adrese lze přičíst nebo od ní odečíst nezáporné celé číslo
- Lze spočítat rozdíl adres stejného typu
- Adresy lze porovnávat stejně jako čísla

Pointerová aritmetika

Součet pointeru a celého čísla

Příklad (ptrsoucet.c)

```
char *ptr_c = 10; /* sizeof(char) = 1 */
int *ptr_i = 10; /* sizeof(int) = 2 */
float *ptr_f = 10; /* sizeof(float) = 4 */
```

Pointerová aritmetika

Přístup k prvkům pole

Příklad

```
int pole[] = {1, 2, 3};

/* pole[0] = 5 */
*pole = 5;

/* pole[1] = 5 */
*(pole + 1) = 5;

/*  chyba: pole[0] + 1 = 2 */
*pole + 1 = 5;
```

Napište program, který pomocí pointerové aritmetiky vypíše všechny prvky pole.

Pointerová aritmetika

Proč pole indexujeme od 0?

Příklad (poleindex.c)

```
int pole[] = {1, 2, 3};

printf("1. prvek je %d \n", pole[0]);
printf("1. prvek je %d \n", *pole);
```

Příklad (poleinde.c)

```
int pole[] = {1,2,3};

printf("3. prvek je %d \n", pole[2]);
printf("3. prvek je %d \n", *(pole+2));
```

Pointerová aritmetika

Proč pole indexujeme od 0?

Příklad (ptrzajimavost.c)

```c
int pole[] = {1, 2, 3};

printf("%d", 2[pole]);
```

```
pole[2] == (pole + 2) == (2 + pole) == 2[pole]
```

Pointerová aritmetika

Porovnávání dvou pointerů

Příklad (ptrpruchodpole3.c)

```
int pole[] = {1,2,3,4,5,6,7,8,9,10};
int *ptr;

for (ptr = pole; ptr < pole + 10; ptr++){
    /* hodnota prvku pole */
    printf("Hodnota *ptr = %d\n", *ptr);
    /* adresa prvku pole */
    printf("Adresa ptr = %p\n\n", ptr);
}
```

Pointerová aritmetika

Příklad

```
int pole[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
```

1. Jak zjistíte adresu 4. prvku pole?
2. Máme ukazatel, který ukazuje na některý prvek pole pole. Jak zjistíte index tohoto prvku v poli?
3. Máme dva ukazatele do stejného pole. Jak zjistíme, který z ukazatelů ukazuje na prvek, který je v poli dříve?

Příklad převrácení pořadí prvků pole

Příklad

Napište funkci, která převrátí pořadí prvků v poli. Využijte pointerové aritmetiky!

Příklad převrácení pořadí prvků pole

Příklad

```
void otoc_pole(int *p, int velikost)
{
    int i;
    int foo;
    for(i=0; i<velikost/2; i++)
    {
        foo = *(p + i);
        *(p + i) = *(p + velikost - 1 - i);
        *(p + velikost - 1 - i) = foo;
    }
}
```

Cvičení

1. Naprogramujte předchozí funkci bez pointerové aritmetiky (přístup k prvkům pole přes hranaté závorky) a srovnejte dobu běhu těchto dvou programů.
   V knihovně `time.h` je definovaná funkce `time()`.
2. Bez použití funkce sizeof zjistěte, jak velkou část paměti zabírají typy char, int a double.
3. Vytvořte libovolnou strukturu a zjistěte, jak velkou část paměti tato struktura zabírá. Bez použití funkce sizeof. Vyzkoušejte pro různé datové typy položek struktury.
4. Naprogramujte funkci `void vypis(int *pole, int zacatek, int krok, int konec)`, která vypíše prvky pole od indexu zacatek po index konec s krokem krok. Například pro pole = {1,2,3,4,5,6,7,8,9,10} a zacatek =0, krok = 2, konec = 9 vypíše prvky 1, 3, 5, 7, 9.