KATEDRA INFORMATIKY
PŘÍRODOVĚDECKÁ FAKULTA
UNIVERZITA PALACKÉHO V OLOMOUCI

Funkce

6. cvičení

Jiří Zacpal

KMI/ZP1 – Základy programování 1

Funkce

- osamostatněné části programu
- „komunikují“ s jinými funkcemi prostřednictvím volání těchto funkcí
- při volání funkce dojde k provedení příkazů jejího těla
- parametry funkce = vstup funkce
- návratová hodnota = výstup funkce
- také `main` je funkce (hlavní funkce programu)

Deklarace funkce

- specifikace identifikátoru funkce, typů parametrů a typu návratové hodnoty dané funkce
- prototyp funkce
- syntaxe:

```
typ_navrat id_fce(typ_1, …, typ_N);
```

- `návratová hodnota povinná v C99`
- příklady:

```
double sqrt(double);
size_t strlen(char *);
int moje_fce(int, char []);
```

Definice funkce

- kompletní popis funkce
- údaje z deklarace jsou doplněny identifikátory formálních parametrů a celého těla (příkazů) funkce
- uvnitř těla funkce přistupujeme k parametrům jako k proměnným s daným identifikátorem
- syntaxe:

```
typ_n id_fce(typ_1 název_1,…, typ_N
název_N)
{ příkazy_těla_funkce }
```

- příklad:

```
unsigned int delka(char str[]){
    unsigned int index = 0;
    while (str[index]!='\0') index++;
    return index;
}
```

Volání funkce a návratová hodnota

- funkci voláme pomocí operátoru volání funkce () uvedeným za identifikátorem dané funkce; uvnitř závorek dále uvedeme skutečné parametry oddělené čárkou
- příklad:
```
delka("Ahoj světe!");
```
- pro opuštění funkce s danou návratovou hodnotou slouží příkaz `return`
- není povinné mít vracet hodnotu z každého místa funkce, ale může to být zdroj chyb
- volání funkce může být součástí složitějšího výrazu, takto lze využít návratovou hodnotu dané funkce
- příklady:
```
int d = delka("Ahoj!");
printf("%i\n", delka("Ahoj!"));
```

Poznámky

- v C99 je povinné používat jen ty funkce, které jsou dříve deklarované

Příklad 1

```c
#include <stdio.h>
#include<math.h>
#define POCET 10

main()
{
int i,  mocniny[POCET], cisla[POCET]={8,4,5,6,7,1,2,9,10,24};

tisk(cisla,POCET,"cisla");
printf("Soucet cisel v poli je: %d\n",soucet(cisla,POCET));
for(i=0; i<POCET;i++) mocniny[i]=(int)pow((long double)cisla[i],2);
tisk(mocniny,POCET,"mocniny");
printf("Soucet cisel v poli je: %d\n",soucet(mocniny,POCET));
}

int tisk(int pole[], int pocet, char nazev[])
{
int i;
printf("Pole %s obsahuje tato cisla: ", nazev);
for(i=0; i<pocet-1;i++) printf("%d, ",pole[i]);
printf("%d\n",pole[i]);
return 0;
}

int soucet(int pole[], int pocet)
{
int i,soucet=0;
for(i=0; i<pocet-1;i++) soucet+=pole[i];
return soucet;
}
```

Příklad 1

```
int tisk(int pole[], int pocet, char nazev[]);
int soucet(int pole[], int pocet);
```

- stačí i takto:

```
int tisk(int [], int, char []);
int soucet(int [], int);

main()
{
   int i,   cisla[POCET]={8,4,5,6,7,1,2,9,10,24};
   float mocniny[POCET];
   tisk(cisla,POCET,"cisla");
   printf("Soucet cisel v poli je:
        %d\n",soucet(cisla,POCET));
   for(i=0; i<POCET:i++) mocniny[i]=(float)pow((long
        double)cisla[i],2);
   tisk(mocniny,POCET,"mocniny");
   printf("Soucet cisel v poli je:
        %d\n",soucet(mocniny,POCET));
}
```

Rozsah platnosti 1/2

- týká se všech identifikátorů (proměnné, typy, funkce)
- funkce nemohou být lokální (vnořené do jiné funkce)
- Lokální proměnná
  - deklarujeme ji uvnitř bloku (cyklus, funkce, ...)
  - při každém vstupu do daného bloku je proměnná vytvořena, je ji přidělena paměť a provedena případná inicializace
  - při opuštění bloku (i před další iterací cyklu) je paměť uvolněna

Rozsah platnosti 2/2

- Globální proměnná
  - deklarujeme ji mimo veškeré bloky
  - při spuštění programu je proměnná vytvořena, je ji přidělena paměť a provedena případná inicializace
  - paměť proměnné se uvolní až při ukončení programu
- Statická proměnná
  - deklarujeme ji na lokální úrovni (uvnitř bloku) se specifikací static (např. `static int pocet;`)
  - při prvním vstupu do daného bloku je proměnná vytvořena a inicializována, zrušena je až ukončením programu
  - platnost identifikátoru proměnné je ovšem lokální, k proměnné nelze přistupovat z jiných částí programu

Příklad 2

```
int cisla[100];

main()
{
    int p,i;
    p=nacti_cislo();
    p=nacti_cislo();
    p=nacti_cislo();
    for(i=0;i<p;i++)
    printf("%d, ",cisla[i]);
}

int nacti_cislo()
{
    static int pocet=0;
    if(pocet<100)
    {
        printf("\nZadej cislo:");
        scanf("%d",&cisla[pocet++]);
    }
    else
    printf("Kapacita pole je jiz naplnena.");
    return pocet;
}
```

Příklad 2

```
main()
{
    int cisla[100],p,i;
    p=nacti_cislo(cisla);
    p=nacti_cislo(cisla);
    p=nacti_cislo(cisla);
    for(i=0;i<p;i++)
    printf("%d, ",cisla[i]);
}

int nacti_cislo(int pole[])
{
    static int pocet=0;
    if(pocet<100)
    {
        printf("\nZadej cislo:");
        scanf("%d",&pole[pocet++]);
    }
    else
    printf("Kapacita pole je jiz naplnena.");
    return pocet;
}
```