KATEDRA
INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

# Práce s preprocesorem

## Základy programování 2

Mgr. Markéta Trnečková, Ph.D.

# Zpracování zdrojového kódu

- preprocesor → překladač → linker
- direktivy preprocesoru: #

**Příklad**

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

# Zpracování zdrojového kódu

## Preprocesor

- zpracovává zdrojový kód před překladem
- nekontroluje syntaktickou správnost
- provádí záměnu textů
- odstraňuje z kódu komentáře
- připravuje podmíněný překlad

# Makra

## Makra bez parametru (symbolické konstanty)

### Příklad

```
o = 2 * 3.14 * r;
```

```
#define JMENO hodnota
```

### Příklad (priklad1.c)

```
#define PI 3.14

int main(){
    float r = 3;
    float o = 2 * PI * r;  /* o = 2*3.14*r; */
    return 0;
}
```

```
gcc priklad1.c -E -o priklad1.txt
```

# Makra

## Makra bez parametru (symbolické konstanty)

### Příklad (priklad2.c)

```c
#define JMENO "Marketa"

int main(){
    /* Vypise Moje jmeno je JMENO */
    printf("Moje jmeno je JMENO");
    return 0;
}
```

Jak vypsat konstantu `JMENO`?

# Makra

## Makra bez parametru (symbolické konstanty)

### Předefinování hodnoty

**Příklad (priklad3.c)**

```
#define JMENO "Marketa"

#undef JMENO

#define JMENO "TRNECKOVA"
```

### Dlouhé konstanty

**Příklad**

```
#define DLOUHA_KONSTANTA 1.23456798\
910111213
```

# Makra

## Předdefinované konstanty

- `__LINE__` – číslo právě zpracovávaného řádku programu (desítkové číslo),
- `__FILE__` – jméno právě zpracovávaného programu (řetězec),
- `__DATE__` – aktuální datum (řetězec ve tvaru mmm dd yyyy),
- `__TIME__` – čas překladu (řetězec ve tvaru hh:mm:ss),
- `__STDC__` – má hodnotu 1 pokud se jedná o ANSI C .

# Makra

## Makra s parametry (inline funkce)

- inline funkce – menší režije, než klasické funkce
- není možné použít rekurzi
- konvence – názvy malým písmem

```
#define jmeno(arg_1, arg_2, ..., arg_n) telo_makra
```

### Příklad (na2.c)

```
#define na2(x) ((x) * (x))
#define na2a(x) x * x

na2(f + g);  /* ((f+g)*(f+g)) */
na2a(f + g); /* f+g*f+g */
```

# Makra

## Makra s parametry (inline funkce)

### Příklad (na2.c)

Co bude výsledkem?

```
#define na2(x) ((x)*(x))

int i = 2;
na2(i++);
```

# Makra

## Makra s parametry (inline funkce)

### Příklad (m.c)

```
#define m(x) ((x) < 0 ? m(-x) : m(x))
```

### Příklad (plus.c)

```
#define add(x,y) (x)+(y)
#define plus(x,y) add(x,y)
```

# Podmíněný překlad

```
#if podminka
    kod
#endif
```

```
#if podminka1
    cast_1
#elif podminka2
    cast_2

    ...

#else
    cast_n
#endif
```

# Podmíněný překlad

## Podmíněný překlad řízený konstantním výrazem

```
#if konstantni_vyraz
    cast_1
#else
    cast_2
#endif

/* NEBO */

#if konstantni_vyraz
    cast_1
#elif konstantni_vyraz2
    cast_2
#else
    cast_3
#endif
```

# Podmíněný překlad

## Podmíněný překlad řízený konstantním výrazem

Příklad (podmineny.c)

```
#define ENG 1

#if ENG
    #define ERROR "error"
#else
    #define ERROR "chyba"
#endif
```

# Podmíněný překlad

## Podmíněný překlad dle symbolické konstanty

`#ifdef`, `#ifndef`

**Příklad**

```
#define ENG 1

#ifdef ENG
    #define ERROR "error"
#else
    #define ERROR "chyba"
#endif
```

# Podmíněný překlad

## Podmíněný překlad dle symbolické konstanty

`defined`

**Příklad**

```
#define ENG 1

#if defined(ENG) && ENG
    #define ERROR "error"
#else
    #define ERROR "chyba"
#endif
```

# Vkládání souborů

- `#include <nazev>`
- `#include "nazev"`

# Cvičení

1. Pro porovnání různého chování maker a funkcí naprogramujte funkci `fna2`, která bude vracet druhou mocninu. Co bude jejím výsledkem pro argument i++?
2. Upravte kód ze slidu 12 tak, aby rozeznával čtyři různé jazyky.
3. Napište makro `na_3(x)`, které bude počítat třetí mocninu. Proměnné `i` a `j` předem definujte. Vyzkoušejte ho na následujících výrazech.
   ```
   na_3(3)
   na_3(i)
   na_3(2 + 3)
   na_3(i * j + 2)
   ```
4. Definujte makro `je_velike(c)`, které vrátí `0` není-li znak velké písmeno a `1`, pokud je.
5. Definujete makro `cti_int(i)`, které čte z klávesnice celé číslo. Makro musí jít použít i ve výrazu
   ```
   if((j=cti_int(k)) == 0)
   ```
   Nápověda: možná budete potřebovat operátor čárky.

## Cvičení

6 Následující kód upravte tak, aby se ve funkci `funkce()` hodnoty proměnných vypisovaly jen v případě, že je makro `VERBOSE` nastaveno na 1.

Příklad

```
#include <stdio.h>
int funkce(int a, int b){
    int i, j = 0;
    for(i = 0; i < a; i = i + b){
        printf("i = %d \n", i);
        if(i > j){
            j = j + i;
        }
        printf("j = %d \n", j);
    }
    return i + j;
}
int main(){
    printf("Vysledek: %d ", funkce(50,5));
    return 0;
}
```

# Cvičení

7 Přeložte předchozí programy s použitím přepínače `-E` (jak už víme, při překladu zdrojového kódu se provede jen předzpracování pomocí preprocesoru). Podívejte se na výsledek.

Příklad

```
gcc program.c -E -o program.txt
```