Informatika 3

12

Metapríkazy

Metapríkazy

- Sú vykonávané preprocesorom
- Začínajú znakom ‘#’
- Typy metapríkazov:
  - symbolické konštanty
  - makrá
  - vkladanie súboru
  - podmienený preklad
  - direktívy pre prekladač

Symbolické konštanty

- Nahrádza identifikátor textom uvedeným za ním
- Zvykom je písať ich identifikátory veľkými písmenami

```
#define PI 3.1415926535
#define BIELA “biela”
```

- Hodnota nemusí byť uvedená – využíva sa hlavne pri podmienenom preklade

```
#define WIN32
```

- V hodnote môžu byť komentáre /*…*/ aj //...

```
#define MIN 2 /* minimalna hodnota */
#define MAX 10 // maximalna hodnota

for(int i=MIN ; i<MAX ; i++){
   …
}
```

- platnosť symbolickej konštanty končí makroinštrukciou “undef”

```
#undef identifikátor
```

KST

Makroinštrukcie - Makrá

```
#define identifikátor(zoznam parametrov) reťazec
```

- definícia makra s formálnymi parametrami
- v programe sa pri volaní makra dosadia za formálne parametre skutočné parametre

```
#define skoc(riadok,stlpec) gotoxy(stlpec,riadok)
```

- použitie

```
skoc (3,2)  // preprekladac nahradi gotoxy(2,3)
```

- operátor ## spája argumenty makra do jedného výrazu:

```
#define FUNKCIA(i,j) funkcia (i##j)
FUNKCIA (x,9)          // preprekladac nahradi funcia (x9)
```

- ak je pred formálnym parametrom znak #, skutočný parameter je konvertovaný na reťazec

```
#define vypis(premenna) printf(#premenna"=%d\n",premenna)
x=10;
vypis(x);              // preprekladac nahradi printf("x""=%d\n",x);
```

- ako identifikátory nesmú byť použité štandardné makrá C-jazyka (_STDC, _DATE, ...)
- platnosť makra končí makroinštrukciou “undef”

```
#define pocet 200
....
#undef pocet
```

KST

Makroinštrukcie - Makrá

- Pri použití je odporúčané používať zátvorky:
  - každý použitý parameter
  - celé makro

```
#define max(a,b) ((a)>(b) ? (a) : (b))
x=max(a,max(b,c));          // ked nebudu zatvorky?
```

- Výhody
  - rýchle operácie
  - podobné ako inline funkcie
  - jedno makro pre viaceré typy parametrov
- Nevýhody
  - zle sa hľadá chyba

KST

Podmienený preklad

```
#if konštantný výraz
...
#elif konštantný výraz
...
#elif konštantný výraz
...
#else
...
#endif
```

- predkladá sa iba jedna vetva, v ktorej je konštantný výraz nenulový
- ak všetky konštantné výrazy sú nulové, bude sa preklať vetva #else
- vetva #else je nepovinná
- je možné využiť metaoperátor

```
defined (identifikátor)          // zátvorky sú nepovinné
```

  - kde:

```
defined(id) = 1      //ak bolo definované #define id, inak
defined(id) = 0      //ak id nie je definované
```

- skrátenie:

```
#if defined(id)    <=> #ifdef id
#if !defined(id)   <=> #ifndef id
```

Vloženie súboru do zdrojového programu

- #include "meno súboru”
- vkladaný súbor vyhľadáva prednostne v aktuálnom adresári, až potom v adresároch, nastavených pomocou OPTIONS/ENVIRONMENT/INCLUDE/DIRECTORIES resp. -I<dir>

- #include <meno súboru>
- súbor vyhľadáva v adresároch OPT./ENV./INCLUDE DIRECTORIES resp. -I<dir>
- namiesto mena súboru je možné použiť makro, obsahujúce názov súboru

- #define subor "c:\\student\\vsuvka.c“
- # include subor
- <=>
- #include "c:\\student\\vsuvka.c"
- vložený súbor môže opäť obsahovať direktívu #include (nie je však povolená rekurzia - ani priama, ani nepriama)

Nastavenie čísla riadku

#line konštanta súbor

- nasledujúcemu riadku priradené číslo "konštanta" a za zdrojový súbor sa považuje "súbor"
- použitie - pri zisťovaní výskytu chýb v programe

```
#line 10
    printf( "This code is on line %d, in file %s\n", __LINE__, __FILE__ );
```

Generovanie chyby pri preklade

#error hlásenie

- Zobrazenie chybového hlásenia pri preklade a kompilácia sa považuje za neúspešnú
- Používa sa pre upozornenie na nekorektný preklad

```
#ifndef __cplusplus
#error C++ compiler required.
#endif
```

Direktíva pragma

- Umožňuje zahrnúť do programu direktívy, ktoré sú pre konkrétnu implementáciu prekladača špecifické
- Ak direktíva nie je implementovaná, nedôjde k výskytu chyby - len sa nevykoná
- #pragma message(„Preklad main")
  - Vypíše správu počas prekladu

```
1>------ Rebuild All started: Project: Spec, Configuration: Debug x64 ------
1>A.cpp
1>cDatum.cpp
1>Main.cpp
1>Preklad main
```

- #pragma once
  - Preklad zdrojového súboru iba raz (hlavičkový súbor)

- #pragma saveregs
  - vkladá sa pred volanie funkcie
  - zabezpečí, že po ukončení volania funkcie budú registre mať rovnakú hodnotu, ako pred volaním funkcie
- ...