Pretprocesor

Osnovna svojstva

- Prije nego što prevodilac krene prevoditi izvorni kod poziva se pretprocesor koji izvršava pretprocesorske naredbe.
- Svaka linija izvornog koda koja počinje znakom `#` predstavlja naredbu pretprocesora. Znaku `#` mogu prethoditi bjeline.
- Pretprocesorska naredba završava krajem linije, a ne znakom točka-zarez.
- Opći oblik pretprocesorskih naredbi je

  `#naredba` parametri

  i one nisu sastavni dio jezika C te ne podliježu sintaksi jezika.

Skup pretprocesorskih naredbi

- Pretprocesorske naredbe definirane standardom ANSI:

```
#if, #ifdef, #ifndef #else, #elif
#include
#define, #undef(ine)
#line
#error
#pragma
```

Naredba `#include`

- Naredba `#include` može se pojaviti u dva oblika:
  - `#include` "*ime_datoteke*"
  - `#include` <*ime_datoteke*>
- U oba slučaja pretprocesor će obrisati liniju s `#include` naredbom i uključiti sadržaj datoteke *ime_datoteke* u izvorni kod.
- Ako je ime datoteke navedeno unutar navodnika onda pretprocesor datoteku traži u direktoriju u kojem se nalazi izvorni program.

Naredba `#define`

- Ime datoteke navedeno unutar oštrih zagrada signalizira da se radi o sistemskoj datoteci, pa će pretprocesor datoteku tražiti na mjestu određenom operacijskim sustavom.
- Naredba `#define`:
  - `#define` *ime tekst_zamjene*
- Pretprocesor će od mjesta na kome je `#define` naredba napisana pa do kraja datoteke svako pojavljivanje imena *ime* zamijeniti s *tekst_zamjene*.
- Do zamjene neće doći unutar znakovnih nizova, tj. unutar dvostrukih navodnika.

Primjer:

- Tako će npr. dio izvornog koda:

```
#define PI 3.14
……
x = 2*r*PI;
printf("PI");
```

prije prevođenja biti zamijenjen s

```
……
x = 2*r*3.14;
printf("PI");   /* Nema zamjene. */
```

Primjer: uključivanje datoteke

```
#include <stdio.h>
#define TRUE 1
#if TRUE
#include "dat.h"
program
#endif

/* sadržaj datoteke dat.h */
#define program main() \
                    {printf("hello\n");}
```

Naredba `#undef`

- Definicija nekog imena može se poništiti naredbom `#undef`.

  Primjer:

```
#define dim 1000
double x[dim];
.....
#undef dim
```

- Pomoću pretprocesorskih naredbi `#if`, `#else`, `#elif` možemo uključivati ili isključivati pojedine dijelove izvornog koda.

Naredbe za uvjetno prevođenje

- Naredba `#if`:
  - `#if` *uvjet*

    blok naredbi

    `#endif`

- Ako je uvjet ispunjen, blok naredbi između `#if` *uvjet* i `#endif` bit će uključen u izvorni kod. Ukoliko uvjet nije ispunjen, blok naredbi između `#if` *uvjet* i `#endif` neće biti uključen u izvorni kod.
- *uvjet* je konstantan cjelobrojni izraz: nula se interpretira kao laž a svaka vrijednost različita od nule kao istina.

Primjer:

- Najčešće se uključivanje/isključivanje dijela programa čini u ovisnosti o tome je li neka varijabla definirana ili ne:

  `defined`(*ime*)

```
#if !defined max
    #error  "max nije definiran!"  /* #error string */
#endif
```

- Ovo je standardna tehnika kojom se izbjegava višestruko uključivanje . h datoteka:

```
#if !defined datoteka
    #define datoteka "abc.h"
    ......
#endif
```

Naredbe `#ifdef` i `#ifndef`

- Budući da se konstrukcije `#if defined` i `#if !defined` često pojavljuju, postoje kraći izrazi s istim značenjem:
  - `#ifdef` i `#ifndef`

Primjer:

`#if defined` max && `!defined` min

.....

`#ifdef` max && `#ifndef` min

.....

  - Zagrade oko varijabli nisu obavezne.

Naredbe `#else` i `#elif`

Primjer:

```
#ifdef debug
  printf("debug = %d\n", debug);
#endif
```

- Naredba `#else` ima isto značenje kao u C-u, a naredba `#elif` ima značenje `else if`.

Primjer:

```
#ifdef stroj1
  #define INTSIZE 32
#elif defined stroj2
  #define INTSIZE 16
#endif
```

Parametrizirane makro naredbe

- U makro naredbi simboličko ime i tekst koji zamjenjuje simboličko ime sadrže argumente koji se definiraju prilikom poziva makro naredbe.

  Primjer:

  `#define` max(a,b) ((a) > (b) ? (a) : (b))

  Ako se u kodu pojavi naredba

  x = max(a1, a2);

  pretprocesor će ju zamijeniti s

  x= ((a1) > (a2) ? (a1) : (a2));

Parametrizirane makro naredbe (2)

a naredbu

```
x = max(a1 + a2, a1 - a2);
```

s

```
x= ((a1 + a2) > (a1 - a2) ? (a1 + a2) : (a1 - a2));
```

- Argumente makro naredbe treba stavljati u zagrade.
- Makro naredba je efikasnija od funkcije jer u njoj nema prenošenja argumenata.

Primjer:

```
#include <stdio.h>

#define SQ1(x) x * x
#define SQ2(x) (x) * (x)
#define SQ3(x) ((x) * (x))

int main()
{
    printf("%d\n", SQ1(1 + 1));
    printf("%d\n", 4/SQ2(2));
    printf("%d\n", 4/SQ3(2));
    return 0;
}
```

Makro naredbe i funkcije

- Ako bismo makro naredbu pozvali na sljedeći način:

  x = max(*i*++, *j*++);

  varijable *i* i *j* ne bi bile inkrementirane samo jednom (kao pri funkcijskom pozivu) već bi veća varijabla bila inkrementirana dva puta.
- Kod makro naredbe nema kontrole tipa argumenata.
- Neke „funkcije” deklarirane u `stdio.h` su u stvari makro naredbe (npr. `getchar`, `putchar`) te većina „funkcija” u `ctype.h`.

Makro naredbe i funkcije (2)

- U naredbi `#define` tekst zamjene se prostire od imena koje definiramo do kraja linije.
- Ako želimo da ime bude zamijenjeno s više linija teksta moramo koristiti kosu crtu (\) na kraju savakog reda osim posljednjeg.

Primjer: Makronaredba za inicijalizaciju polja.

```
#define INIT(polje, dim)  for(i = 0; i < dim; ++i) \
                                  polje[i] = 0.0;
```

Primjer:

```
printf("%s", __DATE__);
printf("%s", __FILE__);
printf("%d", __LINE__);
printf("%s", __TIME__);
```