

FUNKCJA *scanf()*

FUNKCJA *scanf()*

Prototyp funkcji *scanf()* jest zadeklarowany w pliku nagłówkowym *stdio.h*. Jego postać jest następująca:

```
int scanf( const char *format, ... );
```

Bardziej poglądowo prototyp ten można przedstawić następująco:

```
scanf("format", &arg1 [, &arg2 [, ...]] );
```

Funkcja *scanf()* czyta ciąg znaków ze standardowego wejścia *stdin* (najczęściej klawiatury), przekształca na odpowiednie wartości w zależności o specyfikatorów formatu zawartych w pierwszym argumentcie i przypisuje je kolejnym argumentom, których adresy są przekazane jako *&arg1* itd. Należy pamiętać, aby liczba specyfikatorów formatu była równa liczbie argumentów *arg1*, *arg2* itd.

Jeśli w specyfikatorach formatu nie wyszczególni się szerokości pola (tak się czyni najczęściej) to interpretacja znaków wejściowych odpowiadających kolejnemu argumentowi kończy się, gdy zostanie napotkany znak spacji, tabulacji, nowej linii lub znak nie dający się właściwie zinterpretować (np. litera w liczbie).

Specyfikatory formatu używane w funkcji *scanf()* są takie same jak w funkcji *printf()*. Poniżej zestawiono najczęściej używane specyfikatory formatu wykorzystywane w funkcji *scanf()*.

Specyfikator formatu	Typ argumentu
%d	int
%i	int
%x	int, unsigned int
%o	int, unsigned int
%u	unsigned int
%f	float
%e	float
%E	float
%c	char
%s	char*

Użycie modyfikatora *l* w specyfikatorach formatu typu całkowitego (*%ld*, *%li*, *%lx*, *%lo*, *%lu*) powoduje, że typ argumentu wejściowego jest traktowany jako *long int* lub *unsigned long int*.

Użycie modyfikatora *l* w specyfikatorach formatu typu zmiennoprzecinkowego (*%lf*, *%le*, *%lE*) powoduje, że typ argumentu wejściowego jest traktowany jako *double*.

Przykład:

Jeśli dany jest fragment programu:

```
double xd;  
int i1, i2;  
  
scanf("%d%lf%d", &x1, &xd, &x2);
```

to wprowadzenie z klawiatury ciągu znaków:

```
-13 12.67 1 <Enter>
```

spowoduje przypisanie zmiennej *x1* wartości *-13*, *xd* wartości *12.67* i zmiennej *x2* wartości *1*.