

Funkcja: printf()

biblioteka: <stdio.h>

wysyła sformatowane dane do standardowego strumienia wyjściowego (stdout)

```
int printf ( tekst_sterujący , argument_1 , argument_2 , . . . ) ;
```

tekst sterujący → jest to stała łańcuchowa (w cudzysłowach) zawierająca:

- zwykłe znaki (które są po prostu kopiowane na ekran)
- kody formatujące kolejnych argumentów:

- %c** – pojedynczy znak
- %s** – łańcuch znaków
- %d** – liczba dziesiętna ze znakiem
- %f** – liczba zmiennoprzecinkowa (notacja dziesiętna)
- %e** – liczba zmiennoprzecinkowa (notacja wykładnicza)
- %g** – liczba zmiennoprzecinkowa (krótszy z formatów %f %e)
- %u** – liczba dziesiętna bez znaku
- %x** – liczba w kodzie szesnastkowym (bez znaku)
- %o** – liczba w kodzie ósemkowym (bez znaku)
- l** – przedrostek (long) stosowany przed: **d u x o**

przykład:

```
#include <stdio.h>
void main(void)
{
    int x = 10;
    long y = 20;
    double s;
    s = x + y;
    printf ( "%s obliczen %d + %ld = %f" , "Wynik" , x , y , s );
}
```

efekt na ekranie → **Wynik obliczen 10 + 20 = 30.000000**

Aby określić ilość drukowanych cyfr do kodu formatującego można dodać kody długości: **%Xd** **%X.Xf**

- np. **%4d** – liczba dziesiętna na czterech pozycjach
- %10f** – liczba rzeczywista na 10 pozycjach
- %10.2f** – liczba rzeczywista na 10 pozycjach, 2 cyfry po przecinku
- %.3f** – liczba rzeczywista z dokładnością do 3 cyfr po przecinku

Funkcja: scanf()

<stdio.h>

odczytuje dane ze standardowego strumienia wejściowego (stdin) w/g zadanego formatu i zapamiętuje je pod zadanymi adresami pamięci

```
int scanf ( tekst_sterujący , adres_1 , adres_2 , ... );
```

tekst sterujący → jest to stała łańcuchowa (w podwójnych cudzysłowach) zawierająca polecenia jak traktować kolejne dane wczytywane ze strumienia (jakie typy zmiennych są pod adresami *adres_1*, *adres_2*, ...)

Kody formatujące (podobne jak dla **printf()**):

- %c** – pojedynczy znak
- %s** – łańcuch znaków
- %d** – liczba dziesiętna ze znakiem
- %f** lub **%e** – liczba zmiennoprzecinkowa
- %u** – liczba dziesiętna bez znaku
- %x** – liczba w kodzie szesnastkowym (bez znaku)
- %o** – liczba w kodzie ósemkowym (bez znaku)
- l** – przedrostek stosowany przed: **d u x o** (long int)
- l** – przedrostek stosowany przed: **f e** (double)
- L** – przedrostek stosowany przed: **f e** (long double)

& – operator adresowania (zwraca adres zmiennej podanej po operatorze)

przykład:

```
#include <stdio.h>
void main(void)
{
    int x;
    double y;
    char znak;
    printf( "Podaj jedna liczbe calkowita: " );
    scanf ( "%d" , &x );
    printf( "Podaj jedna liczbe rzeczywista i jeden znak: ");
    scanf ( "%lf %c" , &y , &znak );
}
```

Wydruk → **Podaj jedna liczbe calkowita:**

Odczyt ← **123 ↵**

Wydruk → **Podaj jedna liczbe rzeczywista i jeden znak:**

Odczyt ← **456.789 a ↵**

Wynik wczytywania: **x == 123, y == 456.789, znak == 'a'**

PODSTAWOWE INSTRUKCJE JEZYKA C++

- „instrukcja” grupująca - nawiasy klamrowe { } są używane do grupowania wielu deklaracji i instrukcji w jedną instrukcję złożoną (jeden blok).

przykład:

```
#include <stdio.h>
void main( void )
{
    int a = 10, b = 20 ;
    {
        int a = 30 ;           // 'przesłonięcie' poprzedniej definicji zmiennej 'a'
        printf( "A = %d, B = %d \n" , a , b );      // wydruk:  A=30, B=20
    }
    printf( "A = %d, B = %d \n" , a , b );          // wydruk:  A=10, B=20
    . . .
    if( a > 0 )
    {
        printf( "Podaj nową wartość A =" );
        scanf( "%d" , &a );
    }
}
```

- Instrukcja warunkowa (może mieć jedną z dwu postaci) prosta:

```
if ( wyrażenie )
    instrukcja_wewnętrzna ;
```

instrukcja ta sprawdza czy wyrażenie jest prawdziwe (ma wartość różną od zera)

tzn. if (wyrażenie) jest równoważne if (wyrażenie != 0)

przykład:

```
#include <stdio.h>
void main( void )
{
    int liczba;
    printf( "Podaj dowolną liczbę całkowitą A =" ); scanf( "%d" , &liczba );
    if( liczba % 2 == 0)                               // jeżeli reszta z dzielenia przez 2 jest 0
        printf( "Podana liczba jest parzysta" );
}
```

lub instrukcja warunkowa złożona:

```
if ( wyrażenie )
    instrukcja_1 ;
else
    instrukcja_2 ;
```

przykład:

```
#include <stdio.h>
void main( void )
{
    char z;
    printf( "Podaj dowolna duza litere Z = " );    scanf( "%c" , &z );
    if( z >= 'A' && z <= 'Z' )
        printf( "\n\n Dobrze! To jest duza litera" );
    else
        printf( "\n\n Zle! To NIE jest duza litera" );
    printf( "\n\n Nacisnij ENTER, aby zakonczyc program" );
    fflush(stdin); // wyczyszczenie bufora strumienia <stdin> tzn. klawiatury
    getchar( );
}
```

Dalsze przykłady dla instrukcji warunkowej:

```
#include <stdio.h>

void main(void)                                // Wartość maksymalna z trzech wczytanych liczb
{
    int A, B, C;
    printf( "Podaj pierwsza liczbe: " ); scanf( "%d" , &A );
    printf( "Podaj druga liczbe: " );    scanf( "%d" , &B );
    printf( "Podaj trzecia liczbe: " );  scanf( "%d" , &C );

    if( A > B && A > C ) printf( "Maksimum = %d", A );
    if( B > A && B > C ) printf( "Maksimum = %d", B );
    if( C > A && C > B ) printf( "Maksimum = %d", C );

    printf( "\n\n Nacisnij ENTER, aby zakonczyc program" );
    fflush(stdin); getchar();
}

// Co się stanie w przypadku, gdy dwie lub trzy liczby będą równe ?
```

Inne wersje tego samego programu

```
include <stdio.h>
void main(void)           // Wartość maksymalna z trzech wczytanych liczb
{
    int A, B, C;
    printf( "Podaj pierwsza liczbe: " ); scanf( "%d" , &A );
    printf( "Podaj druga liczbe: " );    scanf( "%d" , &B );
    printf( "Podaj trzecia liczbe: " );  scanf( "%d" , &C );

    if( A > B )
        if( A > C )
            printf( "Maksimum = %d" , A );
        else
            printf( "Maksimum = %d" , C );
    else
        if( B > C )
            printf( "Maksimum = %d" , B );
        else
            printf( "Maksimum = %d" , C );

    fflush(stdin); getchar();
}
```

```
#include <stdio.h>           // Wartość maksymalna z trzech wczytanych liczb
void main(void)
{
    int A, B, C, max;
    printf( "Podaj pierwsza liczbe: " );
    scanf( "%d" , &A );
    printf( "Podaj druga liczbe: " );
    scanf( "%d" , &B );
    printf( "Podaj trzecia liczbe: " );
    scanf( "%d" , &C );

    max = A;
    if( max < B ) max = B;
    if( max < C ) max = C;

    printf( "\n Maksymalna wartosc = %d" , max );
    fflush(stdin); getchar();
}
```

/ Program wyliczający pierwiastki trójmianu kwadratowego $Ax^2+Bx+C=0$
z ilustracją zagnieżdżenia instrukcji warunkowych */*

```
#include <stdio.h>
#include <conio.h> // biblioteka firmy Borland >> zawiera funkcję: clrscr()
#include <math.h>
```

```
void main( void )
```

```
{
```

```
    double a, b, c, delta, x1, x2;
```

```
    clrscr();
```

```
    printf( "Podaj pierwsza liczbe A= " );
```

```
    scanf( "%lf" , &a );
```

```
// Uwaga !!! %lf a nie %f
```

```
    printf( "Podaj druga liczbe    B= " );
```

```
    scanf( "%lf" , &b );
```

```
    printf( "Podaj trzecia liczbe  C= " );
```

```
    scanf( "%lf" , &c );
```

```
    delta = b*b - 4*a*c;
```

```
    if( delta < 0 )
```

```
        printf( "\n Brak rozwiazan" );
```

```
    else
```

```
        if( delta == 0 )
```

```
        {
```

```
            x1 = x2 = -b/(2*a);
```

```
            printf( "Jest jedno rozwiazanie x1=x2= %f", x1 );
```

```
        }
```

```
    else
```

```
    {
```

```
        x1 = (-b - sqrt(delta)) / (2*a);  x2 = (-b + sqrt(delta)) / (2*a);
```

```
        printf( "Sa dwa rozwiazania x1= %.2f, x2= %.2f", x1, x2 );
```

```
    }
```

```
    printf( "\n\n Nacisnij ENTER, aby zakonczyc program" );
```

```
    fflush( ); getchar( );
```

```
// fflush( ) można użyć zamiast fflush(...)
```

```
// ale tylko w kompilatorze Borland C++
```

```
}
```

- Konstrukcja else-if:

```
if ( wyrażenie_1 )
    instrukcja_1;
else
if ( wyrażenie_2 )
    instrukcja_2;
else
if ( wyrażenie_3 )
    instrukcja_3;
else
    instrukcja_4;
```

- Instrukcja wyboru:

```
switch ( wyrażenie_całkowite )
{
    case wartość_1 : instrukcja_1;
                    break;

    case wartość_2 :
    case wartość_3 :
    case wartość_4 : instrukcja_234;
                    break;

                    default : instrukcja_domyslna;
                    break;
}
```

```
#include <stdio.h>
void main( void )
{
    int liczba;
    printf( "Podaj wartość liczby całkowitej A =" );
    scanf( "%d" , &liczba );
    switch( liczba )
    {
        case 0 : printf( "Podałeś liczbę zerową" ); break;
        case -5 : printf( "Podałeś liczbę minus pięć" ); break;
        case 7 : printf( "Podałeś liczbę siedem" );
                break;
        case 9 : printf( "Podałeś liczbę dziewięć" ); break;
        default: printf( "Podałeś inną liczbę niż: 0, -5, 7, 9" );
                break;
    }
}
```

Przykład dla instrukcji wyboru:

```
#include <stdio.h>           // Program „kalkulator” zawierający proste „menu”
void main( void )
{
    char znak;
    double a, b, wynik;

    printf( "Podaj pierwsza liczbe A ="); // wczytanie dwóch liczb z klawiatury
    scanf( "%lf", &a );
    printf( "Podaj druga liczbe B =");
    scanf( "%lf", &b );

    printf( "\n\nMozliwe operacje:"); // wyswietlenie „menu”
    printf( "\n (+) wynik = A + B");
    printf( "\n (-) wynik = A - B");
    printf( "\n (*) wynik = A * B");
    printf( "\n (/) wynik = A / B");
    printf( "\n\nPodaj znak operacji: ");

    fflush(stdin);

    znak = getchar( ); // wczytanie znaku wybranej operacji

    switch( znak ) // instrukcja wyboru jednej z operacji arytmetycznych
    {
        case '+': wynik = a + b; break;
        case '-': wynik = a - b; break;
        case '*': wynik = a * b;
                break;
        case '/': wynik = a / b; break;
        default: wynik = 0;
                printf( "\nBład operatora: podano zły znak operacji" );
                break;
    }

    // wydruk liczb i wyniku z zadana dokladnoscia miejsc po przecinku
    printf( "\nWynik obliczen: %.1f %c %.1f = %.2f ",
           a , znak , b , wynik );

    printf( "\n\nKoniec programu. Nacisnij dowolny klawisz" );
    fflush( stdin );
    getchar( );
}
```