Podstawy programowania
(wykład)

Elementarz języka C – część 2

dr inż. Robert Arsoba
Robert.Arsoba@tu.koszalin.pl

Koszalin 2022
Wersja 1.0

Elementarz języka C

Część 2 – operacje we/wy i budowa programu

Operacje wyjścia

Operacje wyjścia

Standardowe wyjście **stdout** (domyślnie ekran monitora)

```
int printf(const char *format, <arg1>, <arg2>, ... );
```

```
printf( "%s" ,   "Język C jest prosty" );
printf( "%c" ,   'a' );
printf( "%d" ,   10 );
printf( "%f" ,   3.14 );
```

specyfikator formatu

informacja wysyłana na wyjście

Operacje wyjścia

Operacje wyjścia – formatowanie

```
printf("%s", "Jan Kowalski");
printf("Jan Kowalski");
```

```
printf("%6s" , "Jan");
printf("%-6s", "Jan");
```

			J	a	n
J	a	n			

```
printf("%5d" , 10);
printf("%-5d", 10);
```

			1	0
1	0			

```
printf("%f",    3.14159);
printf("%7.4f", 3.14159);
printf("%7.2f", 3.14159);
printf("%7.0f", 3.14159);
printf("%.1f",  3.14159);
```

3	.	1	4	1	5	9	0
	3	.	1	4	1	6	
			3	.	1	4	
						3	
3	.	1					

Operacje wyjścia

Operacje wyjścia – podsumowanie funkcji

- **`putchar`**
  `putc` — wysyła pojedynczy znak do **`stdout`**
- **`puts`** — wysyła napis do **`stdout`**
- **`printf`** — wysyła sformatowane dane do **`stdout`**

- **`sprintf`** — jak **`printf`**, ale wysyła dane do tablicy znakowej
- **`vprintf`** — jak **`printf`**, ale dane wysyłane do **`stdout`** pobiera z pól struktury
- **`vsprintf`** — jak **`sprintf`**, ale dane wysyłane do tablicy znakowej pobiera z pól struktury

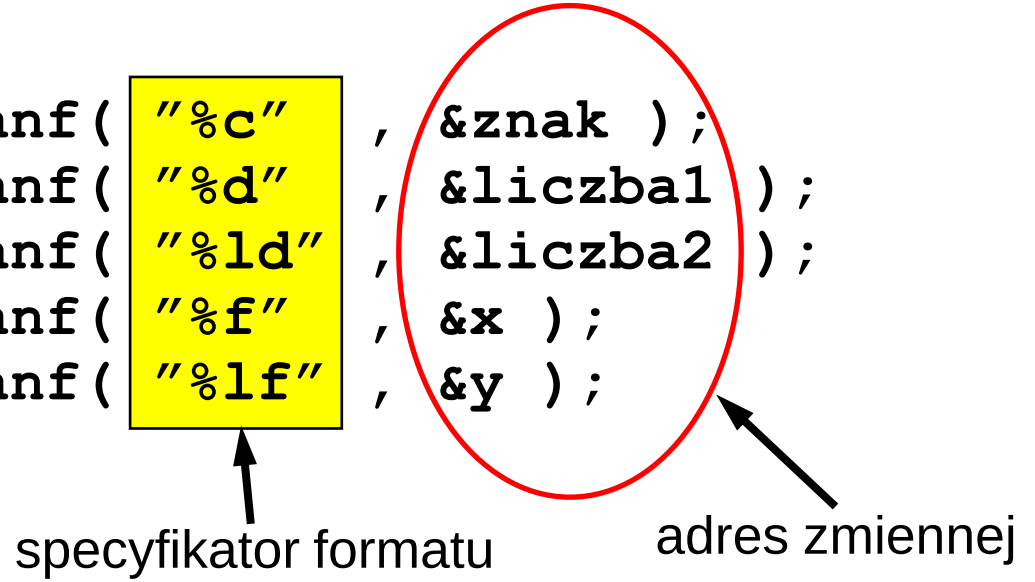
Operacje wejścia

Operacje wejścia

Standardowe wejście: **stdin** (domyślnie klawiatura)

```
int scanf(const char *format, <adr1>, <adr2>, ... );
```

```
char   znak;      scanf( "%c"  , &znak );
int    liczba1;   scanf( "%d"  , &liczba1 );
long   liczba2;   scanf( "%ld" , &liczba2 );
float  x;         scanf( "%f"  , &x );
double y;         scanf( "%lf" , &y );
```



Wczytywanie napisu:

```
char napis[10];
scanf("%s", napis);
scanf("%9s", napis);
gets(napis);
fgets(napis, 10, stdin);
```

Niebezpieczeństwo przepełnienia tablicy:

- niezalecane, działanie nieokreślone
- bezpieczniejsza wersja
- niezalecane, niedostępne od C11
- zalecana, bezpieczna metoda

Operacje wejścia

Operacje wejścia – podsumowanie funkcji

- **`fflush(stdin)`** usuwa dane z **`stdin`** (wyczyszczenie bufora klawiatury), np. przed kolejnym odczytem z wej.
- **`getchar`**, **`getc(stdin)`** pobiera pojedynczy znak z **`stdin`**
- **`gets`** pobiera napis z **`stdin`**, niedostępne od C11
- **`fgets(s, n, stdin)`** pobiera napis z **`stdin`** do **`s`**, maks. n-znaków
- **`scanf`** pobiera sformatowane dane z **`stdin`**

- **`sscanf`** jak **`scanf`**, ale pobiera dane z tablicy znakowej
- **`vscanf`** jak **`scanf`**, ale dane pobierane z **`stdin`** zapisuje w polach struktury
- **`vsscanf`** jak **`sscanf`**, ale dane pobierane z tablicy znakowej zapisuje do pól struktury

Operacje wejścia i wyjścia

Specyfikatory formatu

specyfikator	typ	opis
`%c`	`char`	pojedynczy znak
`%s`	`char *`	ciąg znaków
`%d, %i`	`int`	liczba dziesiętna
`%o`	`int`	liczba ósemkowa bez znaku
`%x, %X`	`int`	liczba szesnastkowa bez znaku
`%ld`	`long`	liczba dziesiętna
`%u`	`unsigned int`	liczba dziesiętna bez znaku

Operacje wejścia i wyjścia

Specyfikatory formatu

specyfikator	typ	opis
`%f`	`float`	liczba zmiennoprzecinkowa pojed. precyzji
`%lf`	`double`	liczba zmiennoprzecinkowa podw. precyzji
`%Lf`	`long double`	liczba zmiennoprzecinkowa
`%e, %E`		liczba zmiennoprzecinkowa (wykładnicza)
`%g, %G`		automatyczny dobór formatu `%f` lub `%e`
`%p`	`void *`	wskaźnik (adres pamięci)

Operacje wyjścia

Znaki (sekwencje) specjalne dla funkcji **`printf`**

znak	opis
`\a`	sygnał dźwiękowy
`\b`	cofnięcie kursora o jeden znak (*backspace*)
`\n`	nowy wiersz
`\r`	powrót kursora na początek wiersza
`\t`	tabulacja pozioma
`\\`	znak `\` (*backslash*)
`\"`	cudzysłów
`\'`	apostrof
`\?`	znak zapytania
`\0`	znak pusty o kodzie 0 (znak **`NULL`**)
`%%`	znak %
`\xhh`	liczba szesnastkowa **`hh`**

Najprostszy program w języku C

Funkcja **`main`**

Wersja 1:

```
int main()
{
   return 0;
}
```

Wersja 2:

```
main()
{

}
```

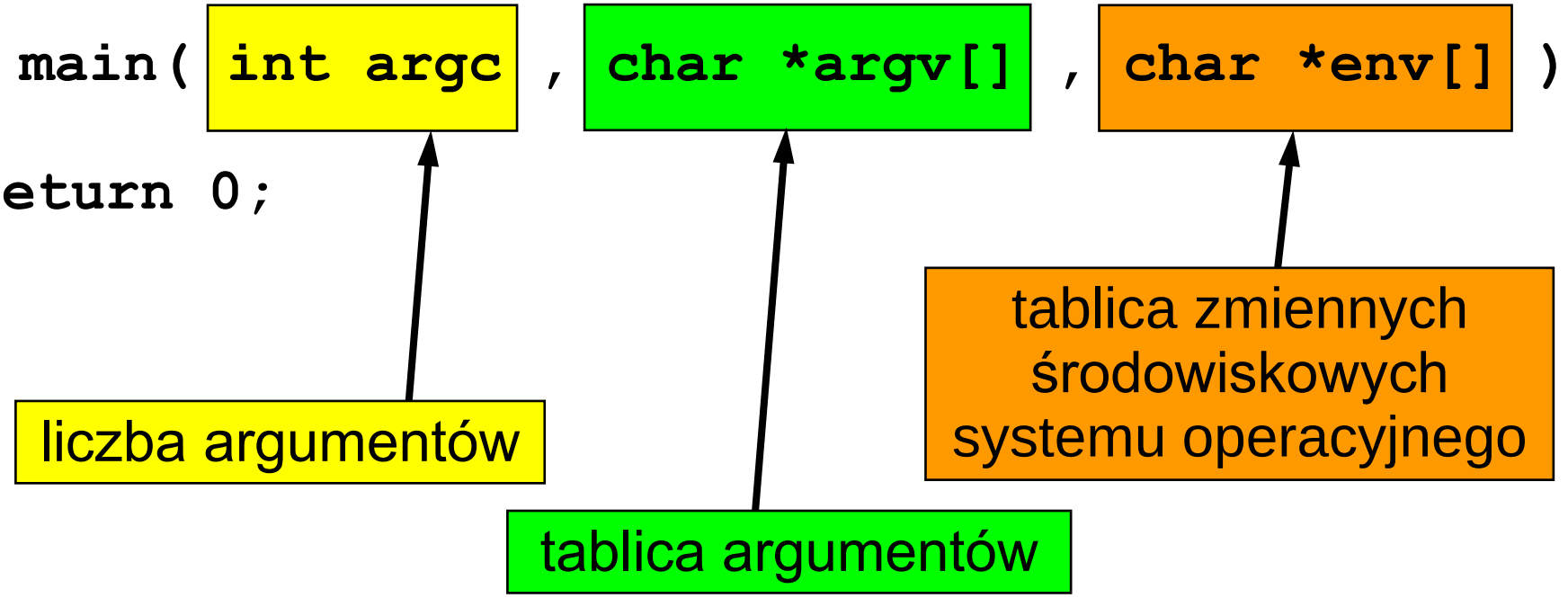
Każdy program w języku C musi zawierać funkcję **`main`**.

Wykonanie programu rozpoczyna się od funkcji **`main`**.

Argumenty funkcji **main**

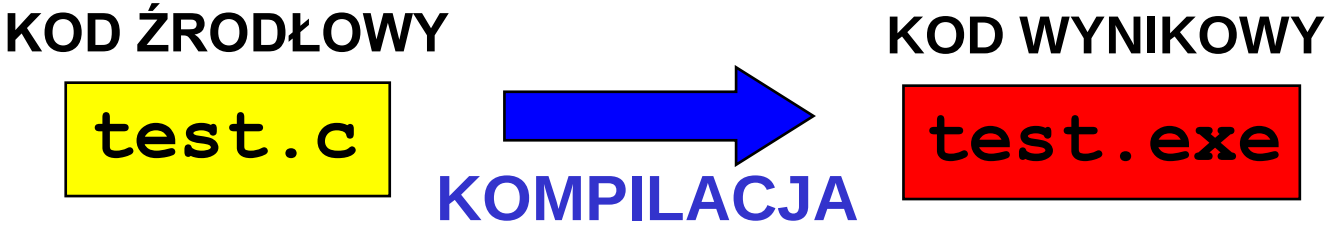

```
int main(int argc, char *argv[], char *env[])
{
   return 0;
}
```

```
int main( int argc , char *argv[] , char *env[] )
{
   return 0;
}
```



Argumenty funkcji **main**


```
int main(int argc, char *argv[], char *env[])
{
   return 0;
}
```



test.exe

argc	1
argv[0]	test.exe

test.exe raz dwa

argc	3
argv[0]	test.exe
argv[1]	raz
argv[2]	dwa

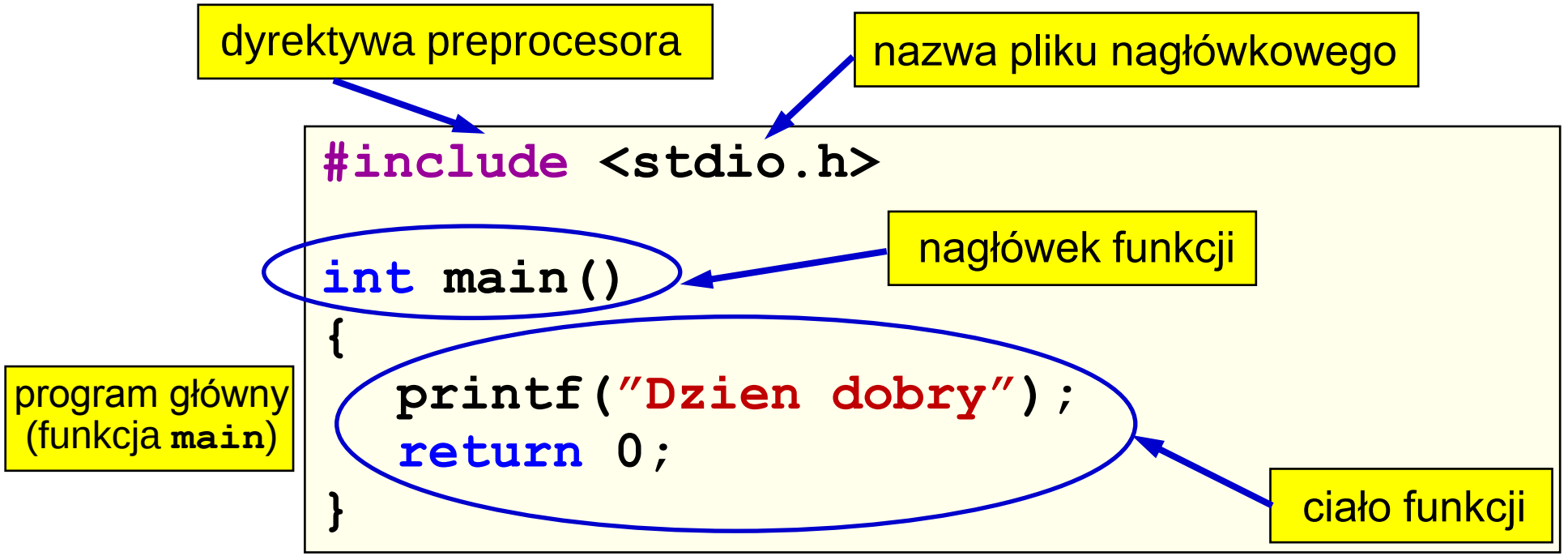
Pierwszy program


```
#include <stdio.h>              /* printf */

int main()
{
   printf("Dzien dobry");       /* napis  */
   return 0;
}
```

```
Dzien dobry_
```

Pierwszy program




```
Dzien dobry_
```

Drugi program


```
#include <stdio.h>   /* printf, scanf */

int a;

int main()
{
   printf("Podaj liczbe: ");
   scanf("%d", &a);
   printf("%d do kwadratu to %d\n", a, a*a);
   return 0;
}
```

- Dyrektywy preprocesora
- Deklaracje globalne
- Funkcja **main** (program główny)

```
Podaj liczbe: 2
2 do kwadratu to 4

_
```

Podsumowanie

Trzeci program

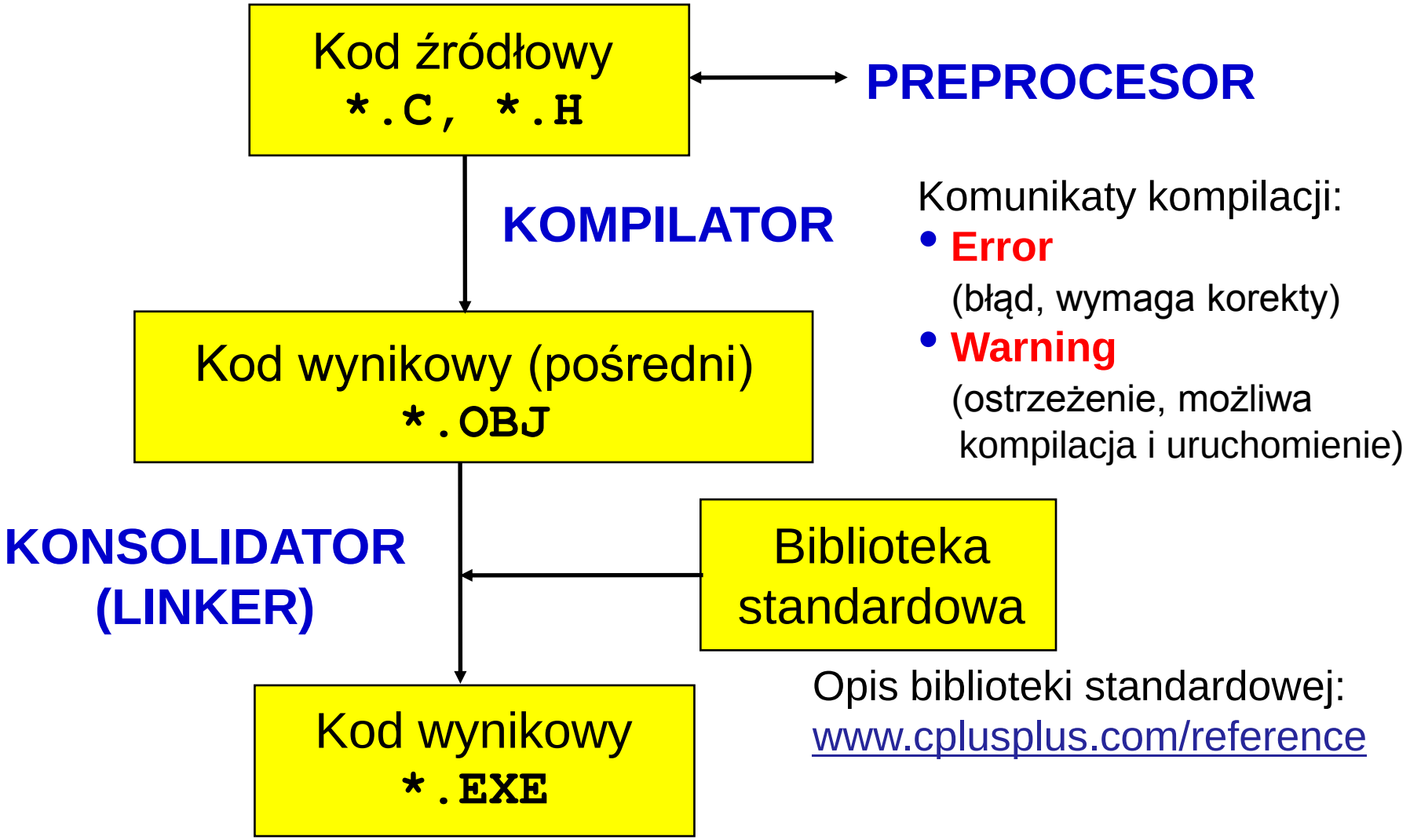
```
#include <math.h>          /* pow, M_PI      */
#include <stdio.h>         /* printf, scanf */

float r, obj;              /* deklaracja zmiennych */
const float PI = 3.14;     /* deklaracja stałej    */

int main()
{
   printf("Podaj promien kuli: ");  /* napis       */
   scanf("%f", &r);                 /* odczyt      */
   obj = 4.0/3.0*PI*pow(r, 3);      /* wyr.arytm.  */
   printf("Objetosc kuli to %5.1f\n", obj);
   return 0;
}
```

Kompilacja w języku C

Specyfika kompilacji w języku C



Komunikaty kompilacji:

- **Error**
  (błąd, wymaga korekty)
- **Warning**
  (ostrzeżenie, możliwa kompilacja i uruchomienie)

Opis biblioteki standardowej:
www.cplusplus.com/reference