

Informatyka 1 (EZ1E2008)

Politechnika Białostocka - Wydział Elektryczny
Elektrotechnika, semestr II, studia niestacjonarne I stopnia
Rok akademicki 2021/2022

Wykład nr 1 (04.03.2022)

dr inż. Jarosław Forenc

Dane podstawowe

- dr inż. Jarosław Forenc
- Politechnika Białostocka, Wydział Elektryczny,
Katedra Elektrotechniki, Energoelektroniki i Elektroenergetyki
ul. Wiejska 45D, 15-351 Białystok
WE-204
- e-mail: j.forenc@pb.edu.pl
- tel. (0-85) 746-93-97
- <http://jforenc.prv.pl>
 - Dydaktyka - slajdy z wykładu
- konsultacje:
 - środa, 13:00-15:00, WE-204 / Teams
 - piątek, 11:00-12:00, WE-204 / Teams
 - piątek, 15:00-16:30, WE-204 / Teams (studia zaoczne)

Program wykładu (1/2)

1. Programowanie w języku C. Deklaracje i typy zmiennych, operatory i wyrażenia arytmetyczne, operacje wejścia-wyjścia, operatory relacyjne i logiczne, wyrażenia logiczne, instrukcja warunkowa if, instrukcja switch, operator warunkowy, pętle (for, while, do .. while), tablice jednowymiarowe.
2. Informacja analogowa i cyfrowa. Pozycyjne i niepozycyjne systemy liczbowe. Konwersje pomiędzy systemami liczbowymi.
3. Jednostki informacji cyfrowej. Kodowanie informacji. Kodowanie znaków.
4. Kodowanie liczb. Reprezentacja liczb w systemach komputerowych: stałoprzecinkowa i zmiennoprzecinkowa. Standard IEEE 754.
5. Architektura komputerów. Klasyfikacja systemów komputerowych (taksonomia Flynna). Architektura von Neumana i architektura harwardzka.

Program wykładu (2/2)

6. Budowa i zasada działania komputera. Procesor, pamięć wewnętrzna i zewnętrzna. Komunikacja z urządzeniami zewnętrznymi, interfejsy komputerowe.
7. System operacyjny. Funkcje i zadania systemu operacyjnego. Zarządzanie procesami, pamięcią i dyskami.
8. Sieci komputerowe. Technologie, protokoły, urządzenia. Zasada działania sieci Internet.
9. Algorytmy. Definicja algorytmu. Klasyfikacje i sposoby przedstawiania algorytmów. Rekurencja. Złożoność obliczeniowa. Wybrane algorytmy sortowania.
10. **Sprawdzian.**

Literatura (1/2)

1. S. Prata: „Język C. Szkoła programowania. Wydanie VI”. Helion, Gliwice, 2016.
2. R. Kawa, J. Lembas: „Wykłady z informatyki. Wstęp do informatyki”. PWN, Warszawa, 2021.
3. W. Kwiatkowski: „Wprowadzenie do kodowania”. BEL Studio, Warszawa, 2010.
4. A.S. Tanenbaum: „Strukturalna organizacja systemów komputerowych”. Helion, Gliwice, 2006.
5. W. Stallings: „Systemy operacyjne. Architektura, funkcjonowanie i projektowanie. Wydanie IX”. Helion, Gliwice, 2018.

Literatura (2/2)

6. J. Kurose, K. Ross: „Sieci komputerowe. Ujęcie całościowe. Wydanie VII”. Helion, Gliwice, 2018.
7. G. Coldwin: „Zrozumieć programowanie”. PWN, Warszawa, 2021.
8. P. Wróblewski: „Algorytmy, struktury danych i techniki programowania. Wydanie VI”. Helion, Gliwice, 2019.

Zaliczenie wykładu - efekty uczenia się

EU1

identyfikuje i opisuje zasadę działania podstawowych elementów systemu komputerowego

EU2

opisuje podstawowe zadania systemu operacyjnego oraz strukturę sieci komputerowych

EU3

formułuje algorytmy komputerowe rozwiązujące typowe zadania inżynierskie występujące w elektrotechnice

- Szczegółowe zasady zaliczania znajdują się na stronie internetowej (<http://jforenc.prv.pl/dydaktyka.html>) oraz w systemie USOS

Zaliczenie wykładu

- Zaliczenie wykładu odbędzie się na podstawie wyników sprawdzianu pisemnego
- Sprawdzian odbędzie się w trybie bezpośrednim (na terenie Uczelni)
- Termin sprawdzianu:
 - zjazd nr 10 (EU1, EU2, EU3)
- Na zaliczeniu każdy efekt uczenia się (EU1, EU2, EU3) będzie oceniany oddzielnie
- Za każdy efekt uczenia się można otrzymać od 0 do 100 pkt.
- Na podstawie otrzymanych punktów wystawiana jest ocena:

Punkty	Ocena	Punkty	Ocena
91 - 100	5,0	61 - 70	3,5
81 - 90	4,5	51 - 60	3,0
71 - 80	4,0	0 - 50	2,0

Zaliczenie wykładu

- Każdy efekt uczenia się musi być zaliczony na ocenę pozytywną (min. 51 punktów)
- Ocena końcowa wyznaczana jest na podstawie sumy punktów:

Punkty	Ocena	Punkty	Ocena
273 - 300	5,0	183 - 212	3,5
243 - 272	4,5	153 - 182	3,0
213 - 242	4,0	0 - 152	2,0

Terminy zajęć

- Wykład nr 1 - 04.03.2022
- Wykład nr 2 - 11.03.2022
- Wykład nr 3 - 18.03.2022
- Wykład nr 4 - 01.04.2022
- Wykład nr 5 - 08.04.2022
- Wykład nr 6 - 06.05.2022
- Wykład nr 7 - 13.05.2022
- Wykład nr 8 - 20.05.2022
- Wykład nr 9 - 27.05.2022
- Wykład nr 10 - 10.06.2022

Plan wykładu nr 1

- Język C
 - historia, struktura programu
 - kompilacja, zapis kodu
 - sekwencje sterujące, komentarze
 - identyfikatory (nazwy), słowa kluczowe
 - typy danych

Język C - krótka historia (1/2)

- **1969** - język BCPL - Martin Richards, University Mathematical Laboratories, Cambridge
- **1970** - język B - Ken Thompson, adaptacja języka BCPL dla pierwszej instalacji systemu Unix na komputer DEC PDP-7
- **1972** - język NB (New B), nazwany później C - Dennis Ritchie, Bell Laboratories, New Jersey, system Unix na komputerze DEC PDP-11
 - 90% kodu systemu Unix oraz większość programów działających pod jego kontrolą napisane w C
- **1978** - książka „The C Programming Language” (Kernighan, Ritchie), pierwszy podręcznik, nieformalna definicja standardu (**K&R**)

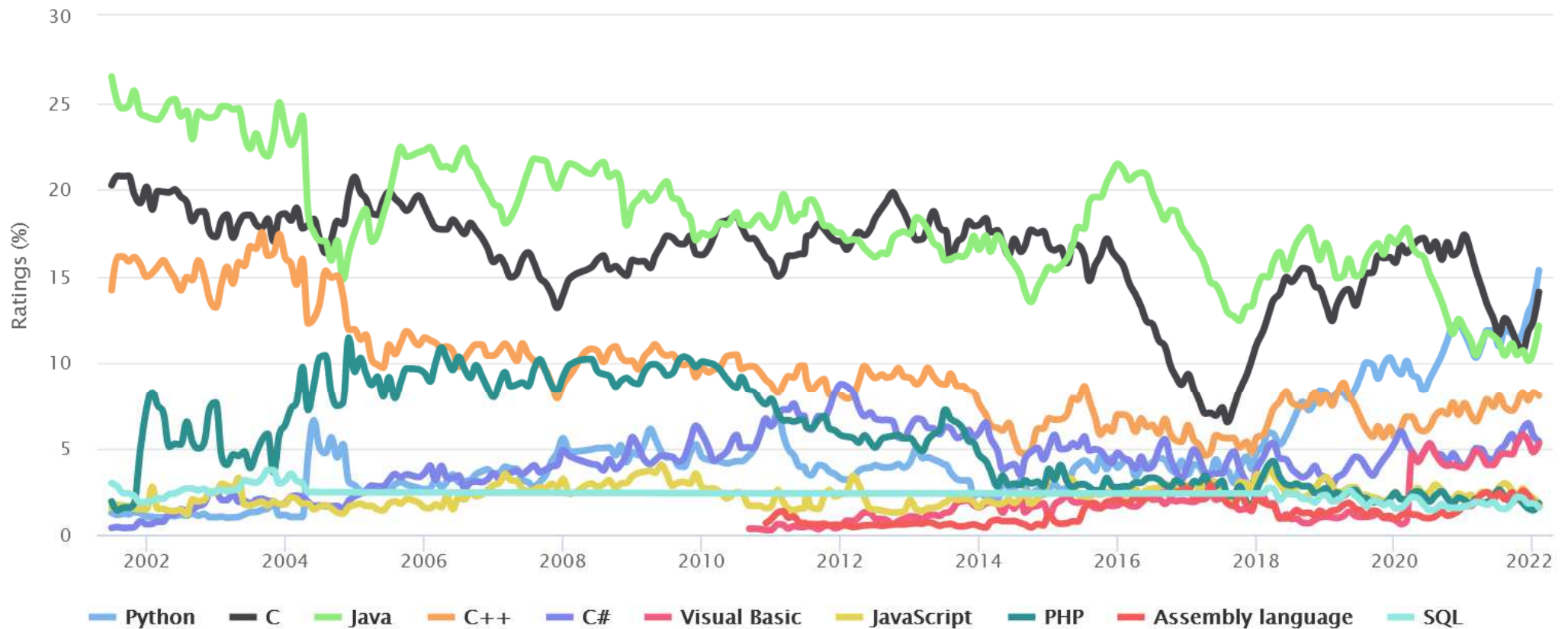
Język C - krótka historia (2/2)

- **1989** - standard ANSI X3.159-1989 „Programming Language C” (ANSI C, C89)
- **1990** - adaptacja standardu ANSI C w postaci normy ISO/IEC 9899:1990 (C90)
- **1999** - norma ISO/IEC 9899:1999 (C99)
- **2011** - norma ISO/IEC 9899:2011 (C11)
- **2018** - norma ISO/IEC 9899:2018 (C18 lub C17)

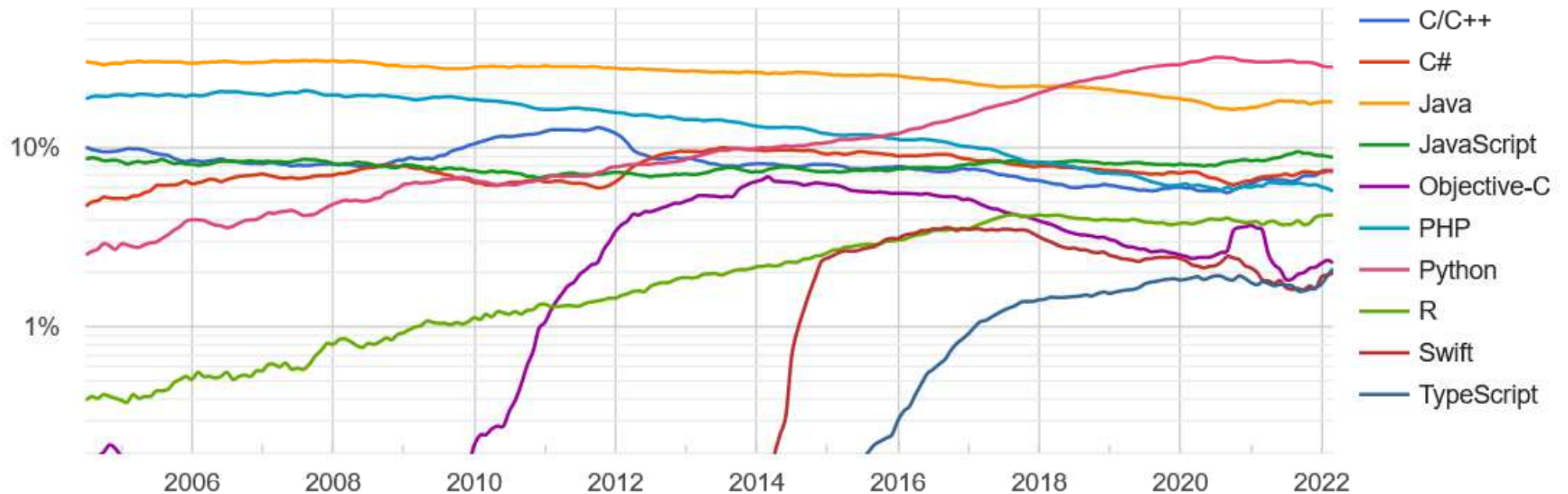
Język C - TIOBE Programming Community Index

TIOBE Programming Community Index

Source: www.tiobe.com



Język C - PYPL Popularity of Programming Language



1	Python
2	Java
3	JavaScript
4	C#
5	C/C++

6	PHP
7	R
8	Objective-C
9	TypeScript
10	Swift

The PYPL Index is created by analyzing how often language tutorials are searched on Google: the more a language tutorial is searched, the more popular the language is assumed to be.

<https://pypl.github.io/PYPL.html>

Język C - pierwszy program

- Niesformatowany plik tekstowy o odpowiedniej składni i mający rozszerzenie **.c**
- Kod najprostszego programu:


```
#include <stdio.h>

int main(void)
{
    printf("Witaj świecie\n");
    return 0;
}
```

- Program konsolowy - wyświetla w konsoli tekst **Witaj świecie**

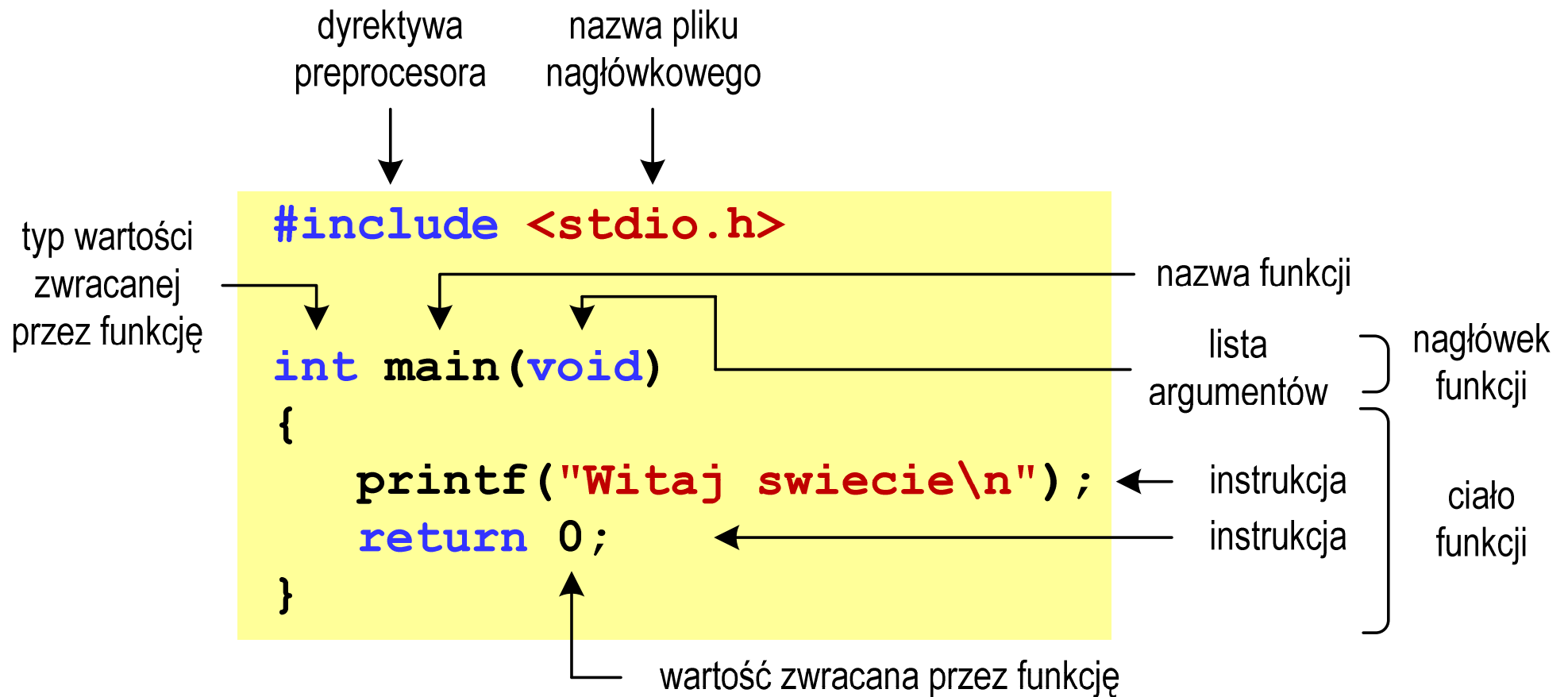
Język C - pierwszy program

- Wynik uruchomienia programu:

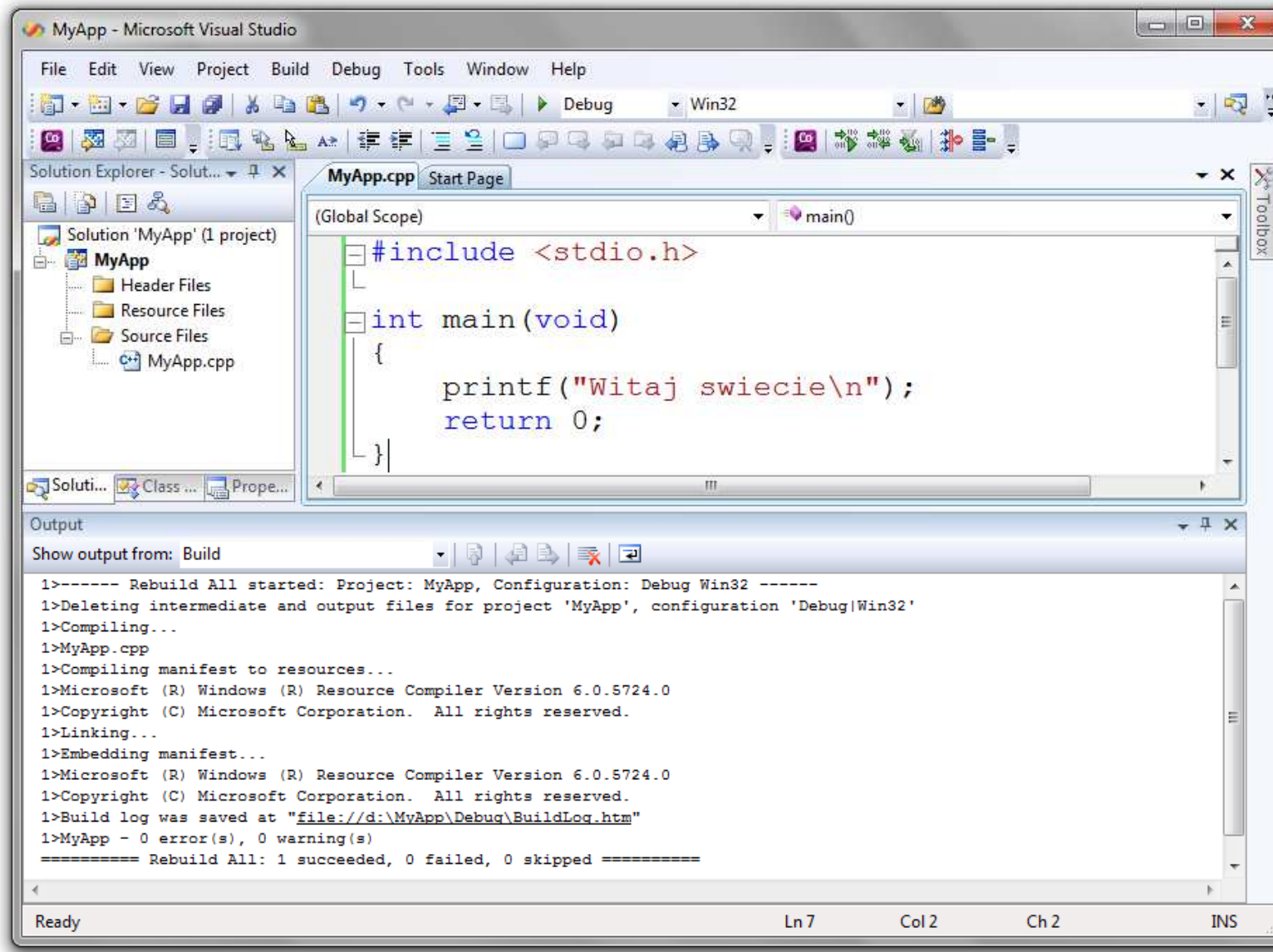


```
C:\Windows\system32\cmd.exe
Witaj swiecie
Aby kontynuować, naciśnij dowolny klawisz . . .
```

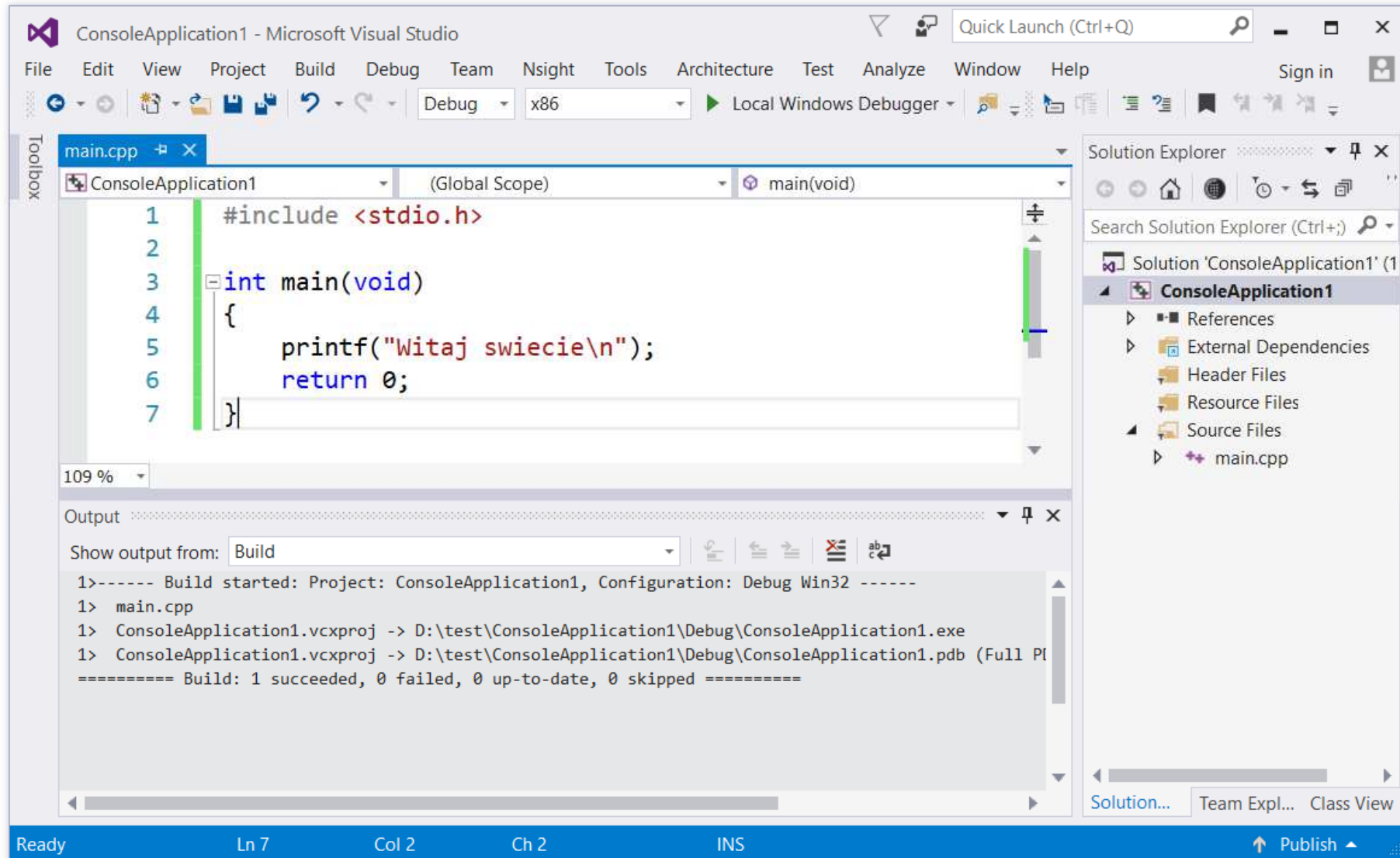
Język C - struktura programu



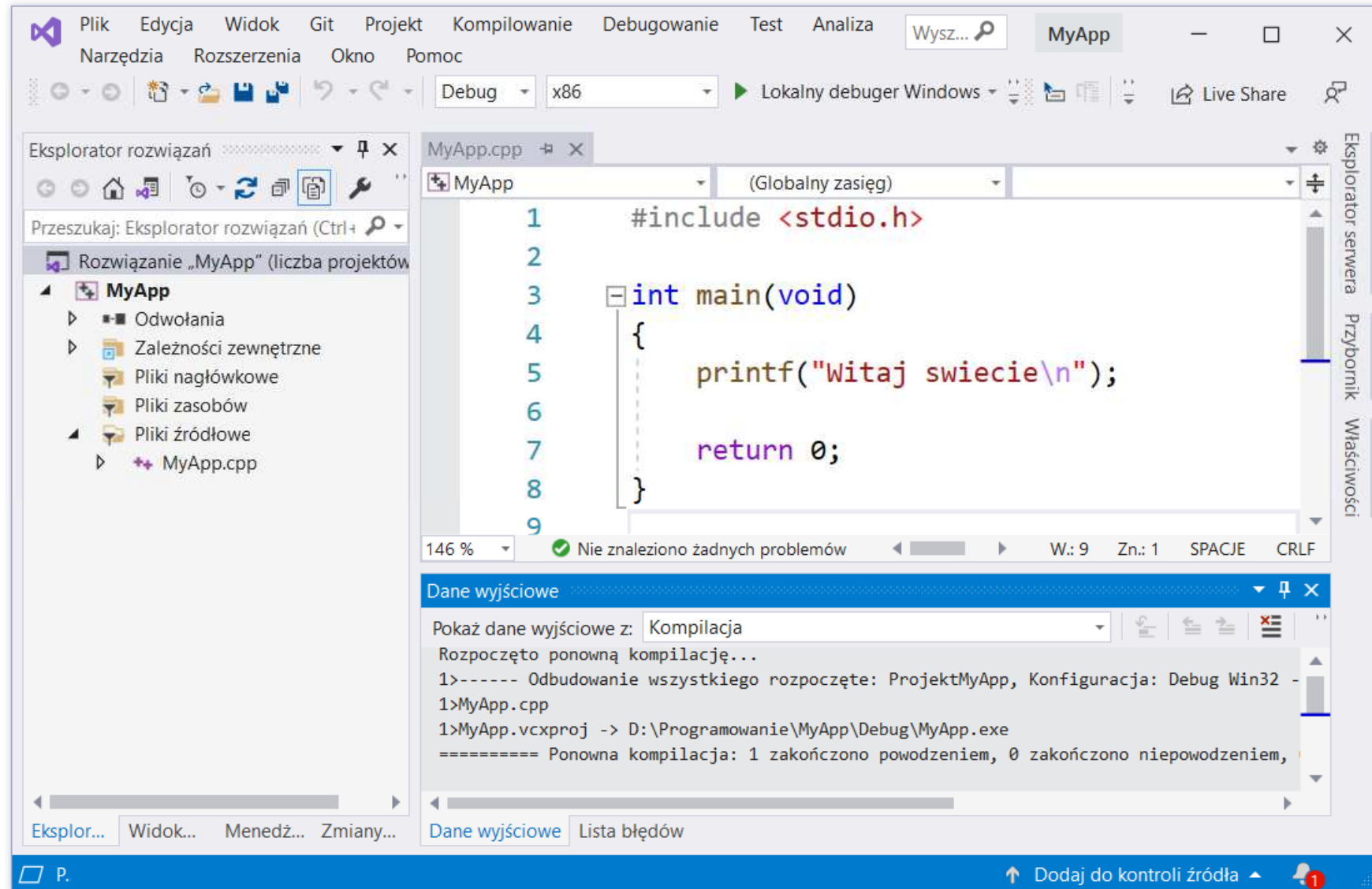
Microsoft Visual Studio 2008



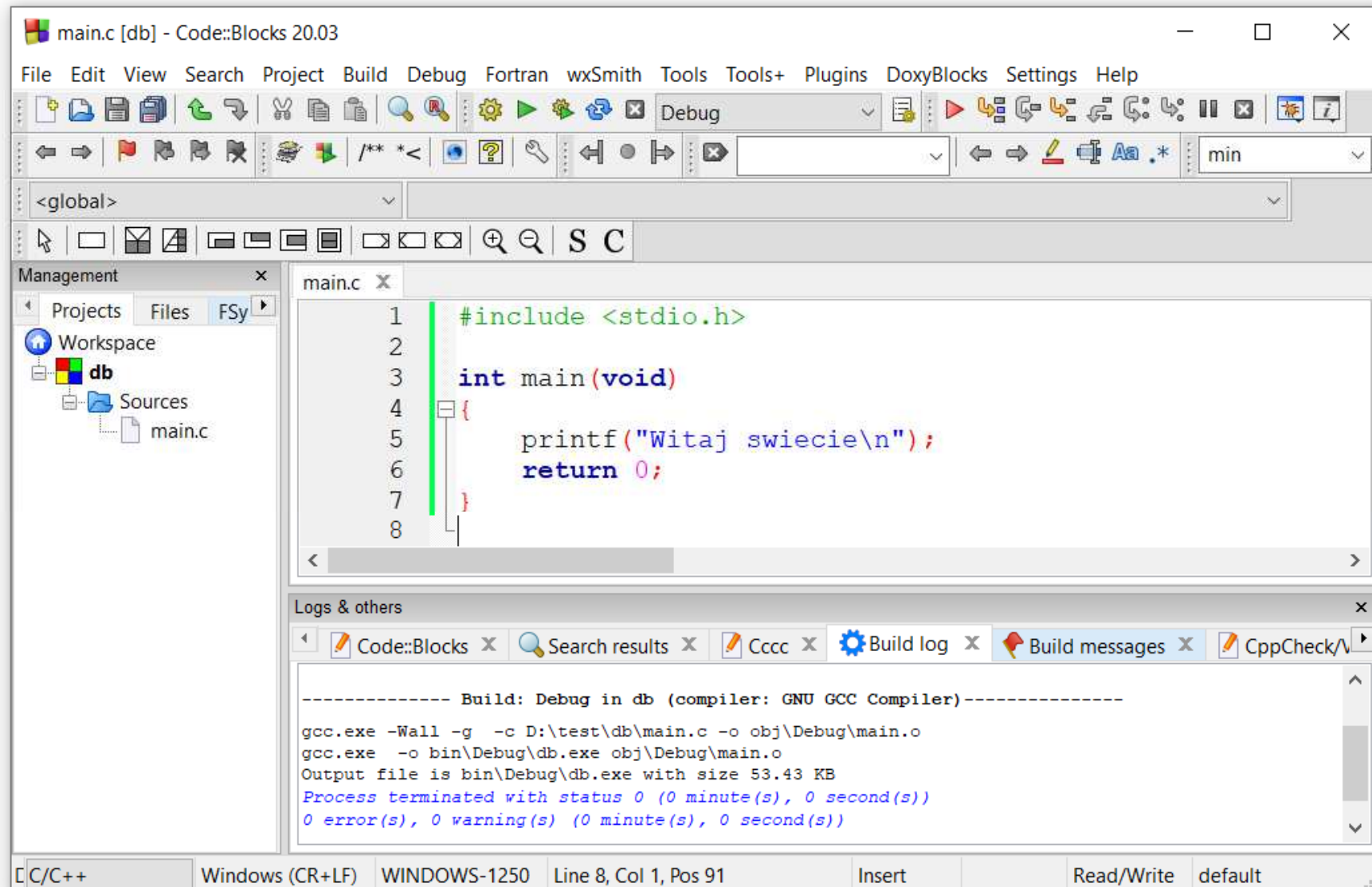
Microsoft Visual Studio 2015



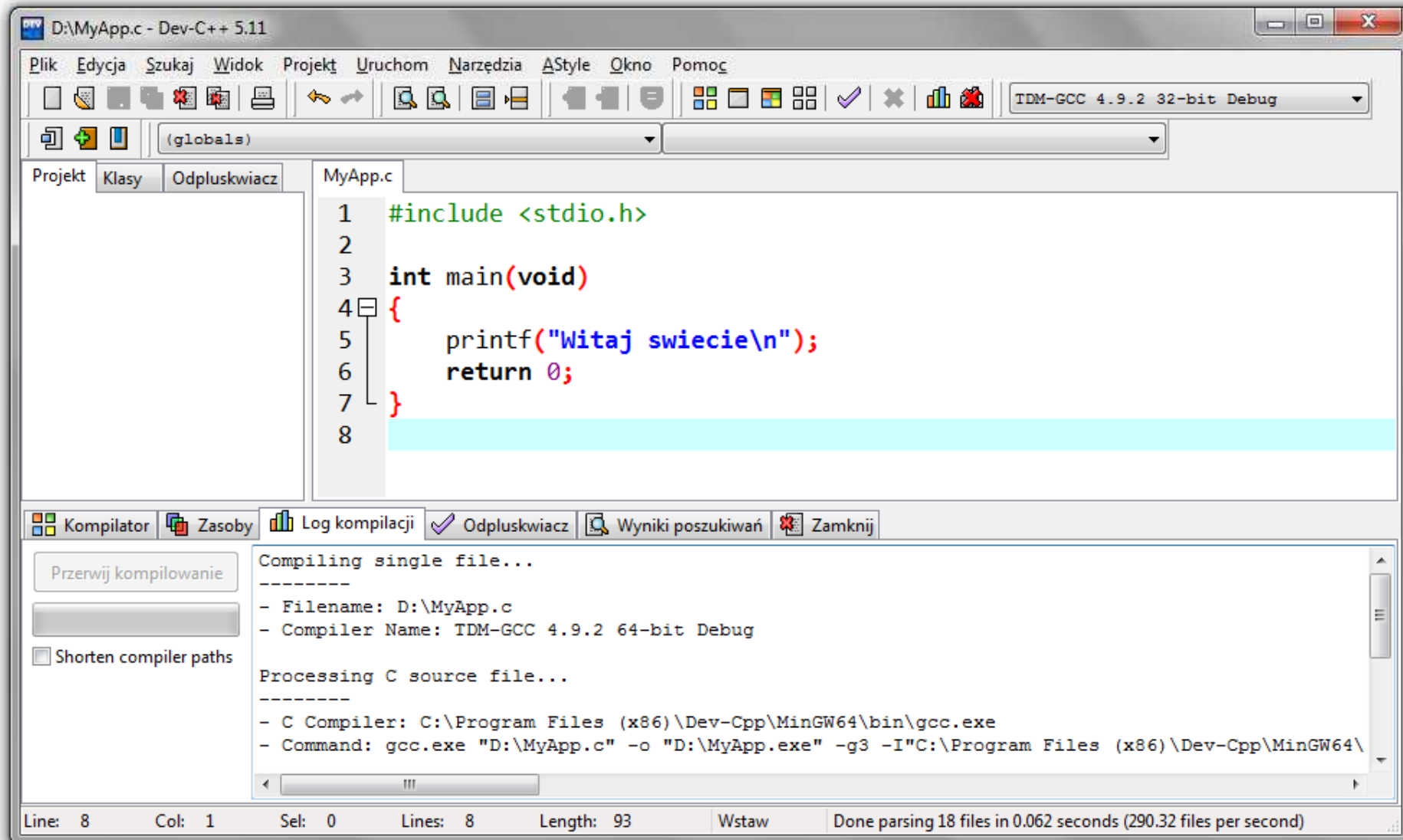
Microsoft Visual Studio 2019



Code::Blocks 20.03

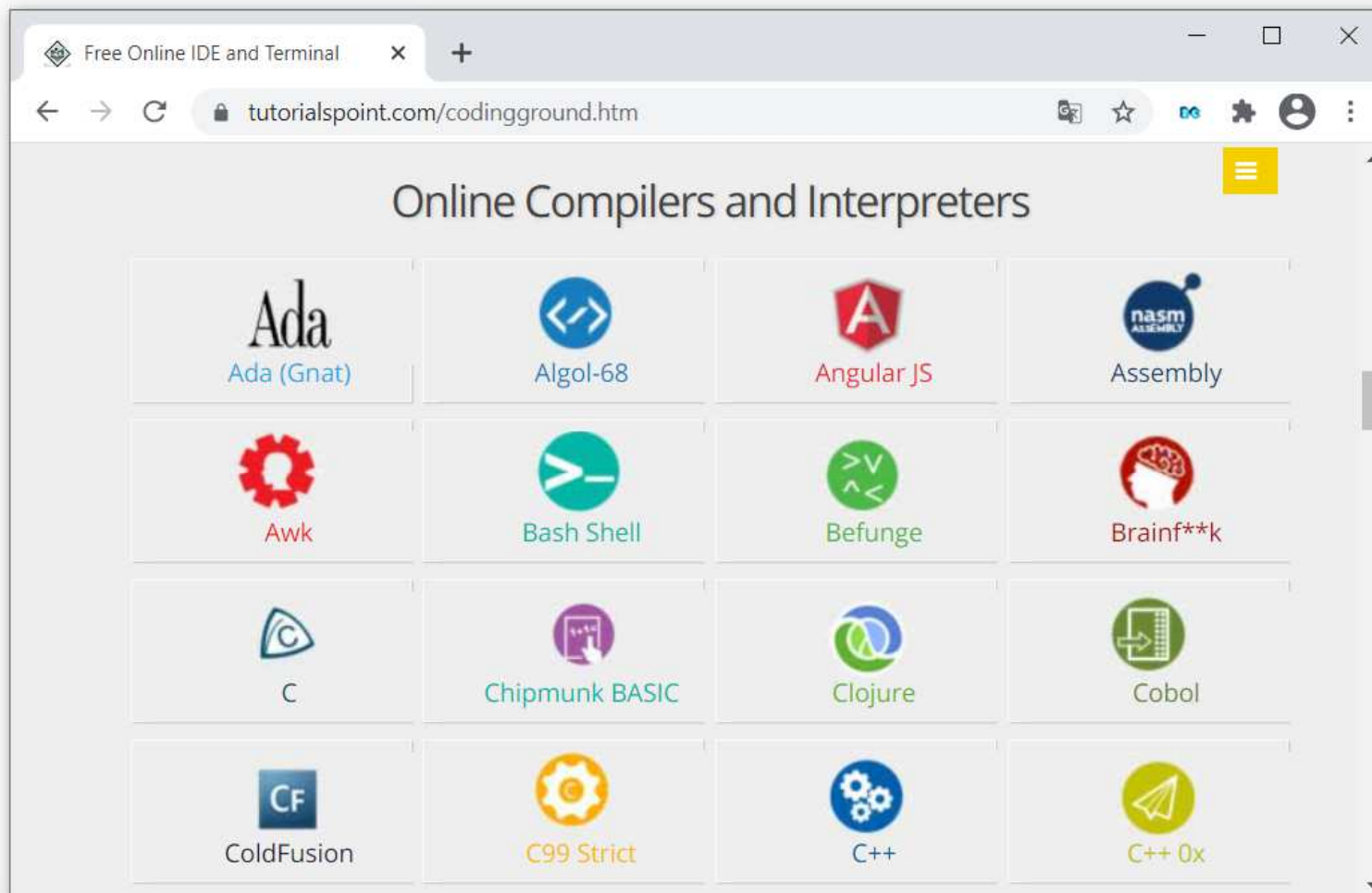


Dev-C++ 5.11



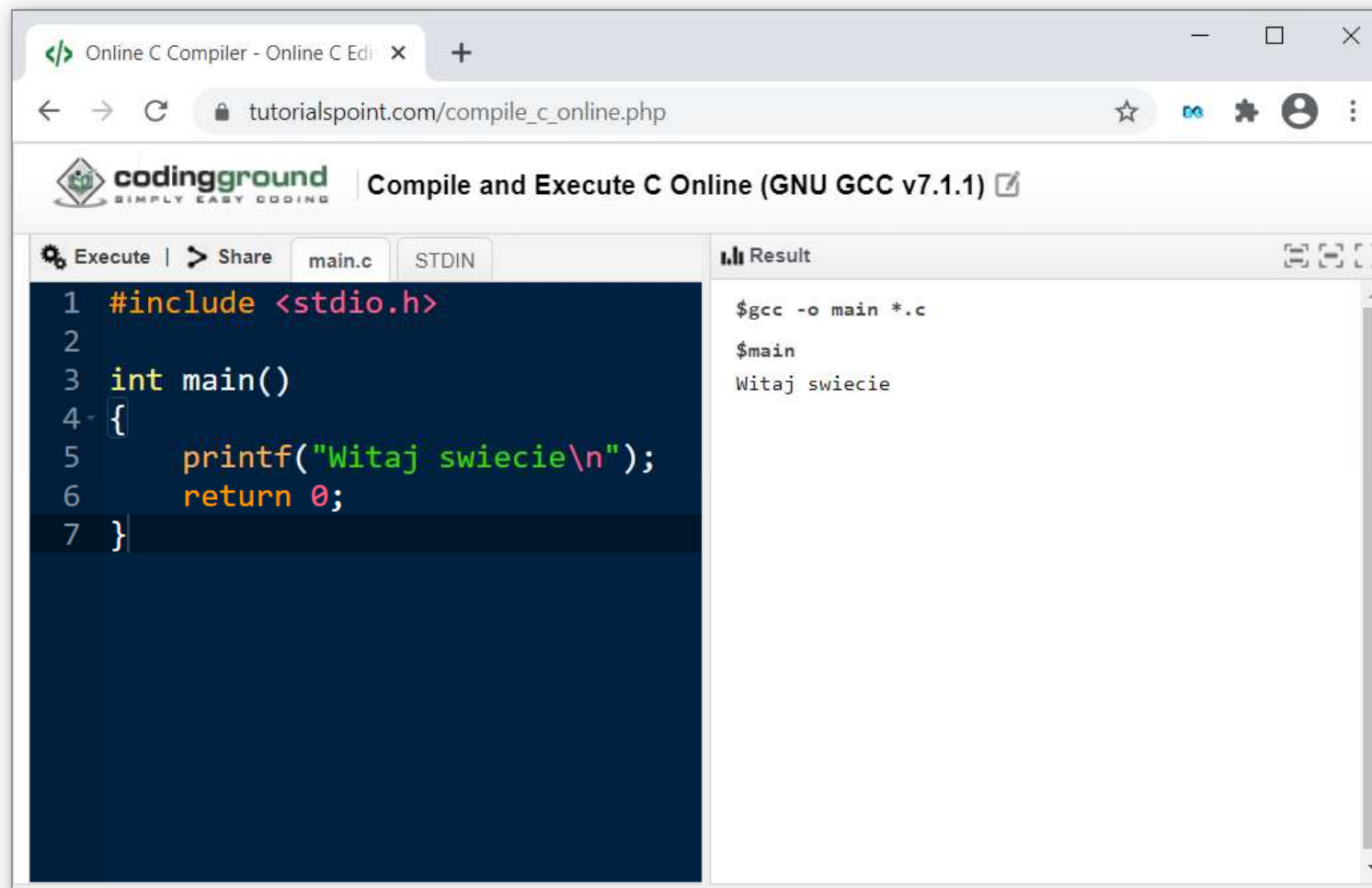
Kompilatory on-line

- <https://www.tutorialspoint.com/codingground.htm>



Kompilatory on-line

- <https://www.tutorialspoint.com/codingground.htm>



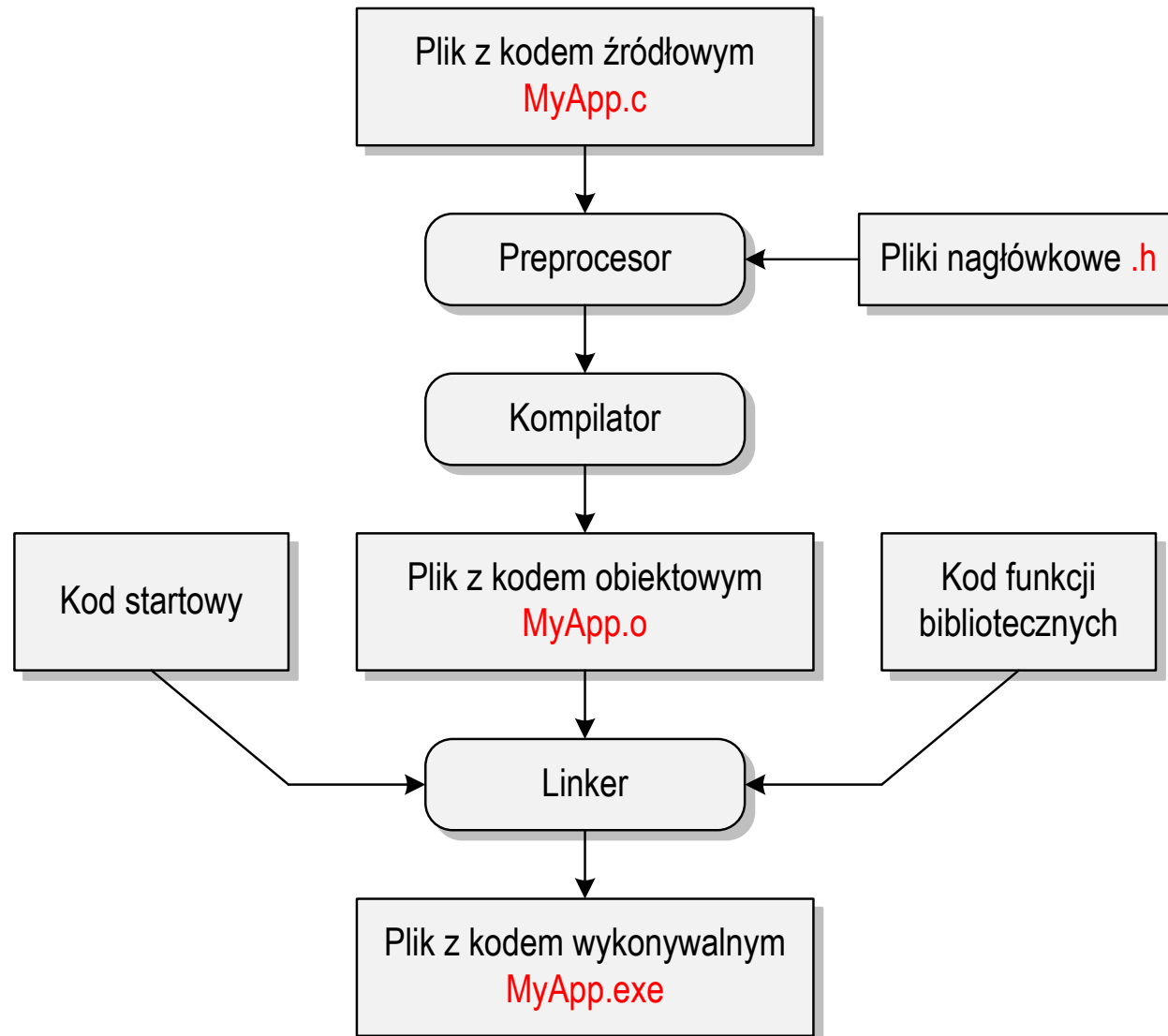
The screenshot shows a web browser window with the URL `tutorialspoint.com/compile_c_online.php`. The page title is "Compile and Execute C Online (GNU GCC v7.1.1)". The interface includes a "codingground" logo and a navigation bar with "Execute", "Share", "main.c", and "STDIN" tabs. The main editor area contains the following C code:

```
1 #include <stdio.h>
2
3 int main()
4 {
5     printf("Witaj swiecie\n");
6     return 0;
7 }
```

The "Result" panel on the right shows the compilation and execution output:

```
$gcc -o main *.c
$main
Witaj swiecie
```

Język C - kompilacja programu



Język C - zapis kodu programu

- Sposób zapisu kodu programu wpływa tylko na jego przejrzystość, a nie na kompilację i wykonanie
- W takiej postaci program także skompiluje się:

```
#include <stdio.h>
int main(void) {printf("Witaj swiecie\n");return 0;}
```

- W Microsoft Visual Studio 2008 można automatycznie sformatować kod źródłowy programu - **Ctrl + K + F**
- Język C rozróżnia **wielkość liter** - poniższy kod nie skompiluje się:

```
#include <stdio.h>
int Main(void) {printf("Witaj swiecie\n");return 0;}
```

Język C - Wyświetlanie tekstu (printf)

- Znak przejścia do nowego wiersza `\n` może pojawić w dowolnym miejscu łańcucha znaków

```
printf("Witaj swiecie\n");
```

```
Witaj swiecie
```

```
—
```

```
printf("Witaj\nswiecie\n");
```

```
Witaj  
swiecie
```

```
—
```

```
printf("Witaj ");  
printf("swiecie");  
printf("\n");
```

```
Witaj swiecie
```

```
—
```

Język C - Sekwencje sterujące

- Istnieją także inne sekwencje sterujące (ang. escape sequence)

Opis znaku	Zapis w printf()
Alarm (ang. alert), głośniczek wydaje dźwięk	<code>\a</code>
Backspace	<code>\b</code>
Wysunięcie strony (ang. form feed)	<code>\f</code>
Przejdźcie do nowego wiersza (ang. new line)	<code>\n</code>
CR - Carriage Return (powrót na początek wiersza)	<code>\r</code>
Tabulacja pozioma (odstęp) (ang. horizontal tab)	<code>\t</code>
Tabulacja pionowa (ang. vertical tab)	<code>\v</code>

Język C - Wyświetlenie znaków specjalnych

- Niektóre znaki pełnią specjalną funkcję i nie można wyświetlić ich w tradycyjny sposób

Opis znaku	Znak	Zapis w printf()
Cudzysłów	"	\"
Apostrof	'	\'
Ukośnik (ang. backslash)	\	\\
Procent	%	%%

```
Sciezka dostepu: "C:\dane\plik.txt"
```

```
printf("Sciezka dostepu: \"C:\\dane\\plik.txt\\\"\n");
```

Język C - Wyświetlenie znaku o podanym kodzie

- Można wyświetlić dowolny znak podając jego kod w systemie ósemkowym lub szesnastkowym

Znaczenie	Zapis
Znak o podanym kodzie ASCII (system ósemkowy)	<code>\ooo</code>
Znak o podanym kodzie ASCII (system szesnastkowy)	<code>\xhh</code>

```
printf("\127\151\164\141\152\040");  
printf("\x73\x77\x69\x65\x63\x69\x65\x21\x0A");
```

```
Witaj swiecie!
```

Język C - Wyświetlenie tekstu

```
#include <stdio.h>

int main(void)
{
    printf("-----\n");
    printf(" | Punkty | Ocena | \n");
    printf("-----\n");
    printf(" | 91-100 | 5,0 | \n");
    printf(" | 81-90 | 4,5 | \n");
    printf(" | 71-80 | 4,0 | \n");
    printf(" | 61-70 | 3,5 | \n");
    printf(" | 51-60 | 3,0 | \n");
    printf(" | 0-50 | 2,0 | \n");
    printf("-----\n");

    return 0;
}
```

Punkty	Ocena
91-100	5,0
81-90	4,5
71-80	4,0
61-70	3,5
51-60	3,0
0-50	2,0

Język C - Komentarze

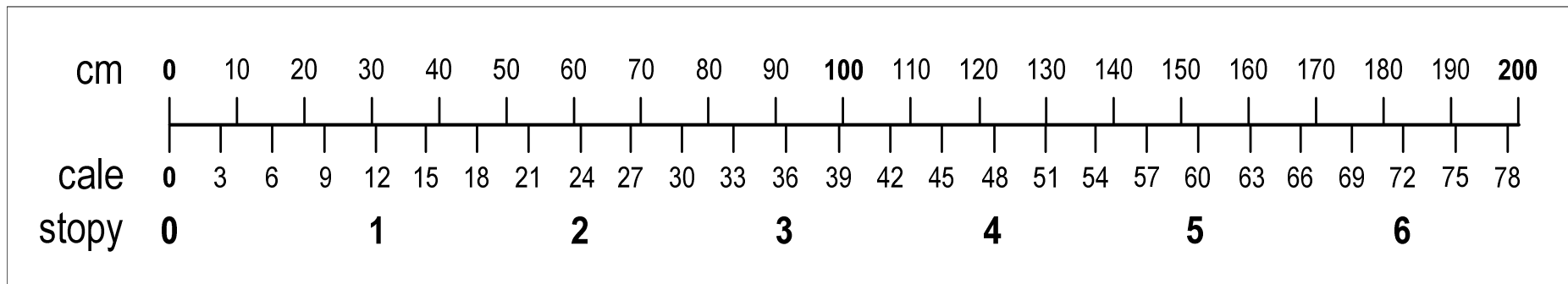
- Komentarze są pomijane podczas kompilacji

```
/*  
  Nazwa: MyApp.c  
  Autor: Jarosław Forenc, Politechnika Białostocka  
  Data: 04-03-2022 17:00  
  Opis: Program wyświetlający tekst "Witaj świecie"  
*/  
  
#include <stdio.h>      // zawiera deklarację printf()  
  
int main(void)          // nagłówek funkcji main()  

```

Przykład: zamiana wzrostu w cm na stopy i cale

- Wybrane jednostki długości w brytyjskim systemie miar:
 - 1 cal (inch) [in] = 2,54 [cm]
 - 1 stopa (foot) [ft] = 12 cali = 30,48 [cm]



- 1 jard (yard) [yd] = 3 stopy = 91,44 [cm]
- 1 furlong [fur] = 660 stóp = 201,168 [m]
- 1 mila (mile) [mi] = 8 furlongów = 1609,344 [m]

Przykład: zamiana wzrostu w cm na stopy i cale

```
#include <stdio.h>
```

```
int main(void)  
{
```

```
    float cm;        /* wzrost w cm */  
    float stopy;     /* wzrost w stopach */  
    float cale;      /* wzrost w calach */
```

```
    printf("Podaj wzrost w cm: ");  
    scanf("%f", &cm);
```

```
    stopy = cm / 30.48f;  
    cale = cm / 2.54f;
```

```
    printf("%f [cm] = %f [ft]\n", cm, stopy);  
    printf("%f [cm] = %f [in]\n", cm, cale);
```

```
    return 0;
```

```
}
```

Podaj wzrost w cm: 175

175.000000 [cm] = 5.741470 [ft]

175.000000 [cm] = 68.897636 [in]

Język C - identyfikatory (nazwy)

- Dozwolone znaki: **A-Z, a-z, 0-9, _** (podkreślenie)
- Długość nie jest ograniczona (rozdzielalne są 63 pierwsze znaki)
- Poprawne identyfikatory:

`temp` `u2` `u_2` `pole_kola` `alfa` `Beta` `XyZ`

- Pierwszym znakiem nie może być cyfra
- W identyfikatorach nie można stosować spacji, liter diakrytycznych
- Błędne identyfikatory:

`2u` `pole kola` `pole_koła`

Język C - identyfikatory (nazwy)

- Nie zaleca się, aby pierwszym znakiem było podkreślenie
- Identyfikatory nie powinny być zbyt długie

```
_temp    __temp    temperatura_w_skali_Celsjusza
```

- Nazwa **zmiennej** powinna być związana z jej zawartością
- Język C rozróżnia wielkość liter więc poniższe zapisy oznaczają inne identyfikatory

```
tempc    Tempc    TempC    TEMPC    TeMpC
```

- Jako nazw zmiennych nie można stosować **słów kluczowych** języka C

Język C - słowa kluczowe języka C

- W standardzie C11 zdefiniowane są 43 słowa kluczowe

auto	extern	short	while
break	float	signed	_Alignas
case	for	sizeof	_Alignof
char	goto	static	_Bool
const	if	struct	_Complex
continue	inline	switch	_Generic
default	int	typedef	_Imaginary
do	long	union	_Noreturn
double	register	unsigned	_Static_assert
else	restrict	void	_Thread_local
enum	return	volatile	

Język C - Typy danych

Nazwa	Rozmiar (bajty)	Zakres wartości
<code>char</code>	1	-128 ... 127
<code>int</code>	4	-2147483648 ... 2147483647
<code>float</code>	4	$-3,4 \cdot 10^{38} \dots 3,4 \cdot 10^{38}$
<code>double</code>	8	$-1,7 \cdot 10^{308} \dots 1,7 \cdot 10^{308}$
<code>void</code>	-	-

- Słowa kluczowe wpływające na typy:
 - `signed` - liczba ze znakiem (dla typów `char` i `int`), np. `signed char`
 - `unsigned` - liczba bez znaku (dla typów `char` i `int`), np. `unsigned int`
 - `short`, `long`, `long long` - liczba krótka/długa (dla typu `int`), np. `short int`
 - `long` - większa precyzja (dla typu `double`), `long double`

Język C - Typy danych

- Zależnie od środowiska programistycznego (kompilatora) zmienne typów **int** i **long double** mogą zajmować różną liczbę bajtów

Środowisko	int (bajty)	long double (bajty)
Microsoft Visual Studio 2008	4	8
Microsoft Visual Studio 2019	4	8
Dev-C++ 5.11	4	16*
Code::Blocks 20.03	4	16*
Borland Turbo C++ 2006	4	10
Borland C++ 3.1	2	10

Język C - Typy danych (sizeof)

- **sizeof** - operator zwracający liczbę bajtów zajmowanych przez obiekt lub zmienną podanego typu

```
sizeof(nazwa_typu)
sizeof(nazwa_zmiennej)
sizeof nazwa_zmiennej
```

- Operator **sizeof** zwraca wartość typu **size_t**
- Zależnie od środowiska programistycznego typ **size_t** może odpowiadać typowi **unsigned int** lub **unsigned long int**
- W standardach C99 i C11 wprowadzono specyfikator formatu **%z**, który określa, że występujący po nim specyfikator (**d, i, o, u, x, X**) dotyczy wyświetlania wartości typu **size_t** (np. **%zd**)

Język C - Typy danych (sizeof)

```
#include <stdio.h>
```

```
int main(void)
```

```
{
```

```
    int x;
```

```
    printf("int: %d\n", sizeof(int));
```

```
    printf("int: %d\n", sizeof(x));
```

```
    printf("int: %d\n", sizeof x);
```

```
    printf("long double: %d\n", sizeof(long double));
```

```
    return 0;
```

```
}
```

```
int: 4  
int: 4  
int: 4  
long double: 8
```

Koniec wykładu nr 1

Dziękuję za uwagę!