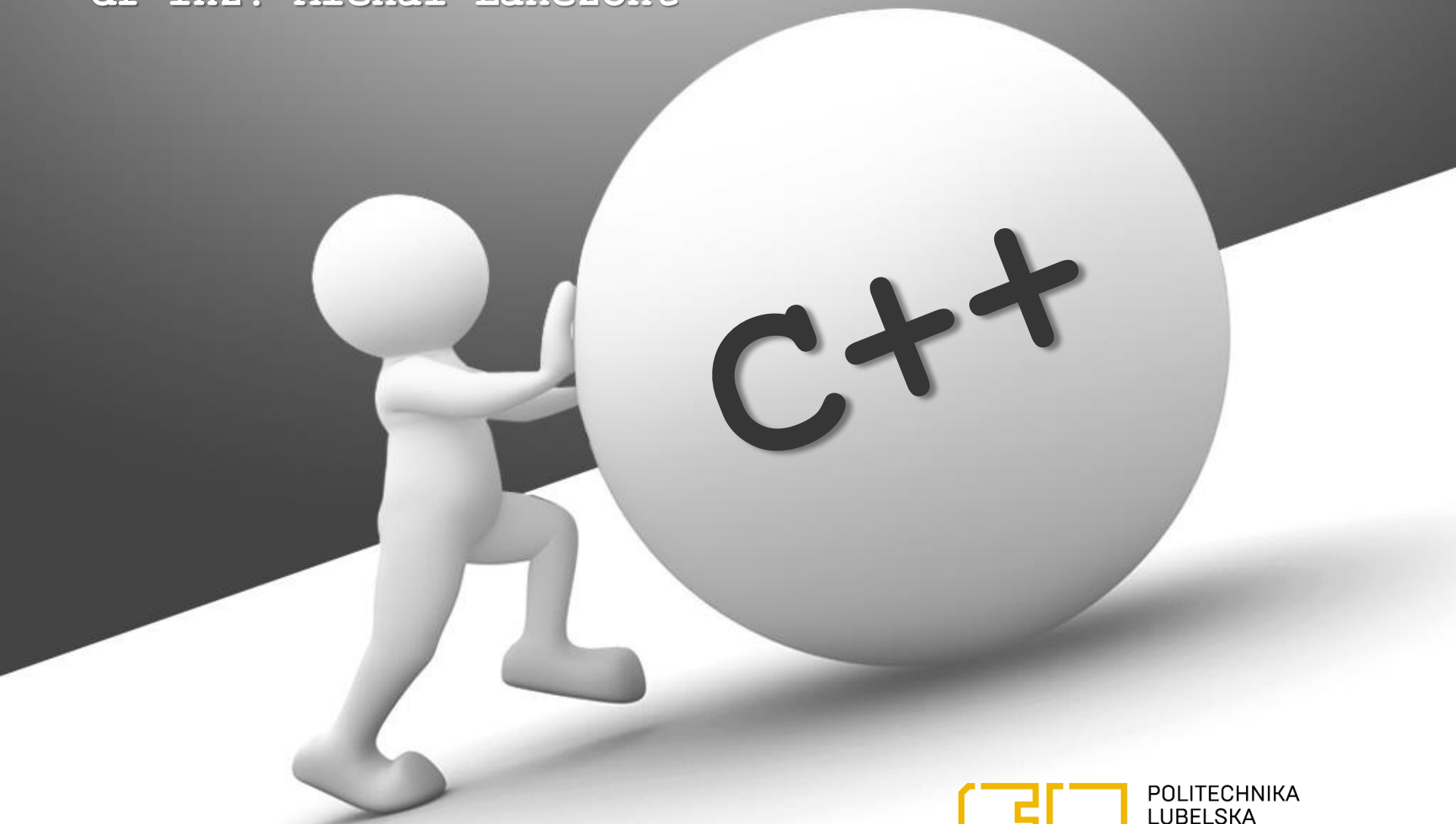
Informatyka II

dr inż. Michał Łanczont

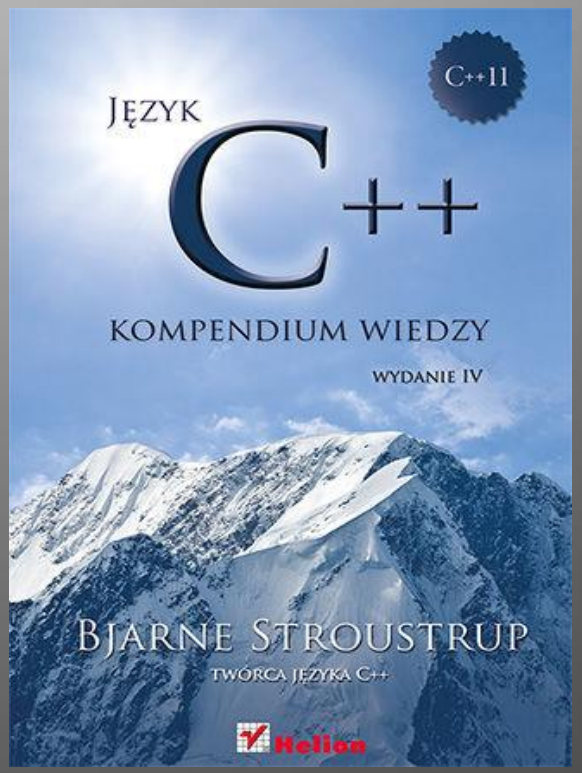


POLITECHNIKA
LUBELSKA
WYDZIAŁ ELEKTROTECHNIKI
I INFORMATYKI

Zaliczenie przedmiotu

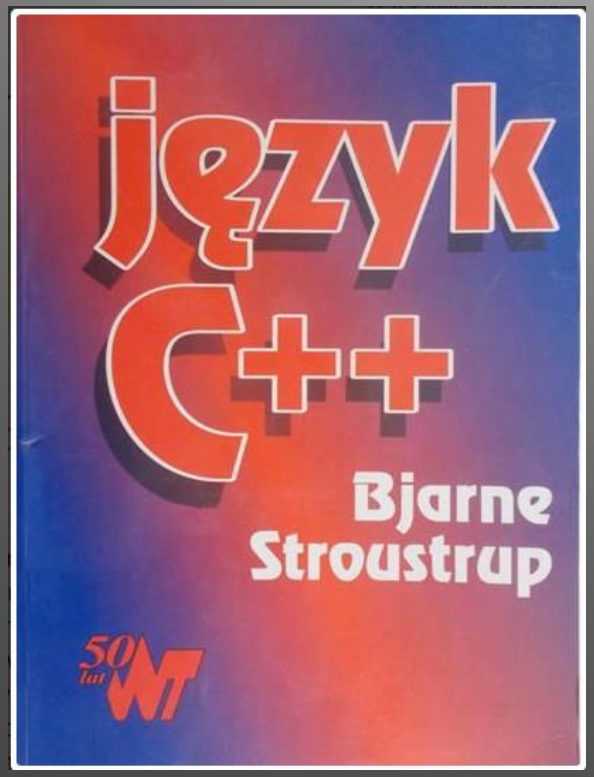
- **Wykład – egzamin**
  - Egzamin pisemny
  - Pierwszy termin w sesji letniej
  - W sesji egzamin poprawkowy I i II (wrześniowy)
- **Laboratoria – Określone przez prowadzącego**
  - Zaliczenie ćwiczeń cząstkowych
  - Zaliczenie końcowe na ostatnich zajęciach

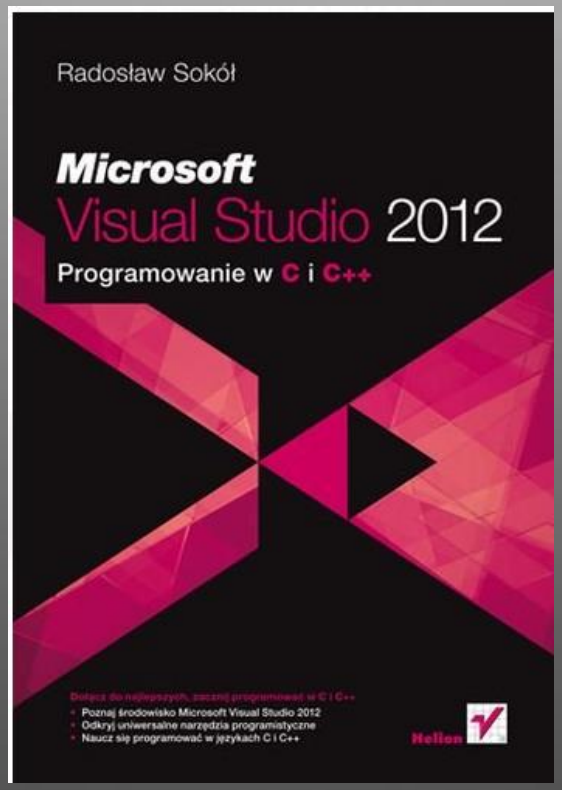
Literatura

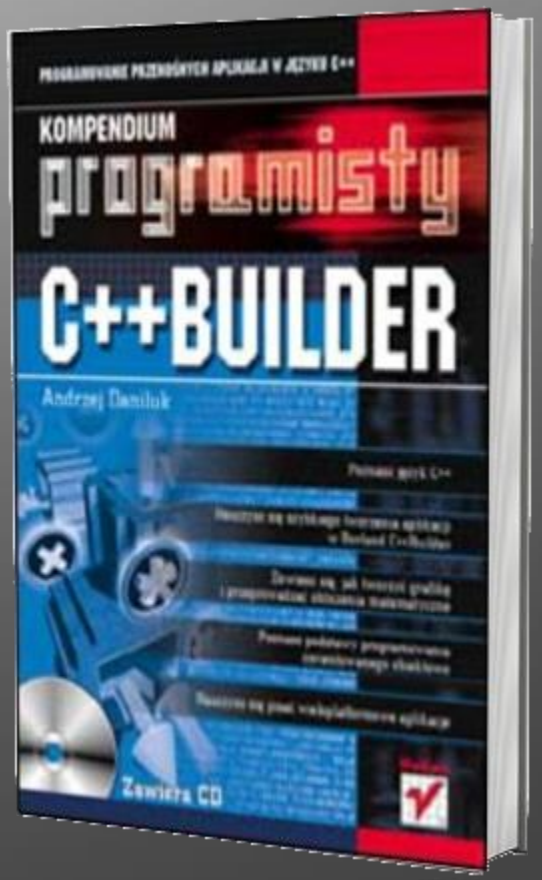


- Stephen Prata, *Język C++. Szkoła programowania*. Wydanie VI, Helion
- Bjarne Stroustrup, *Język C++. Kompendium wiedzy*. Wydanie IV, Helion

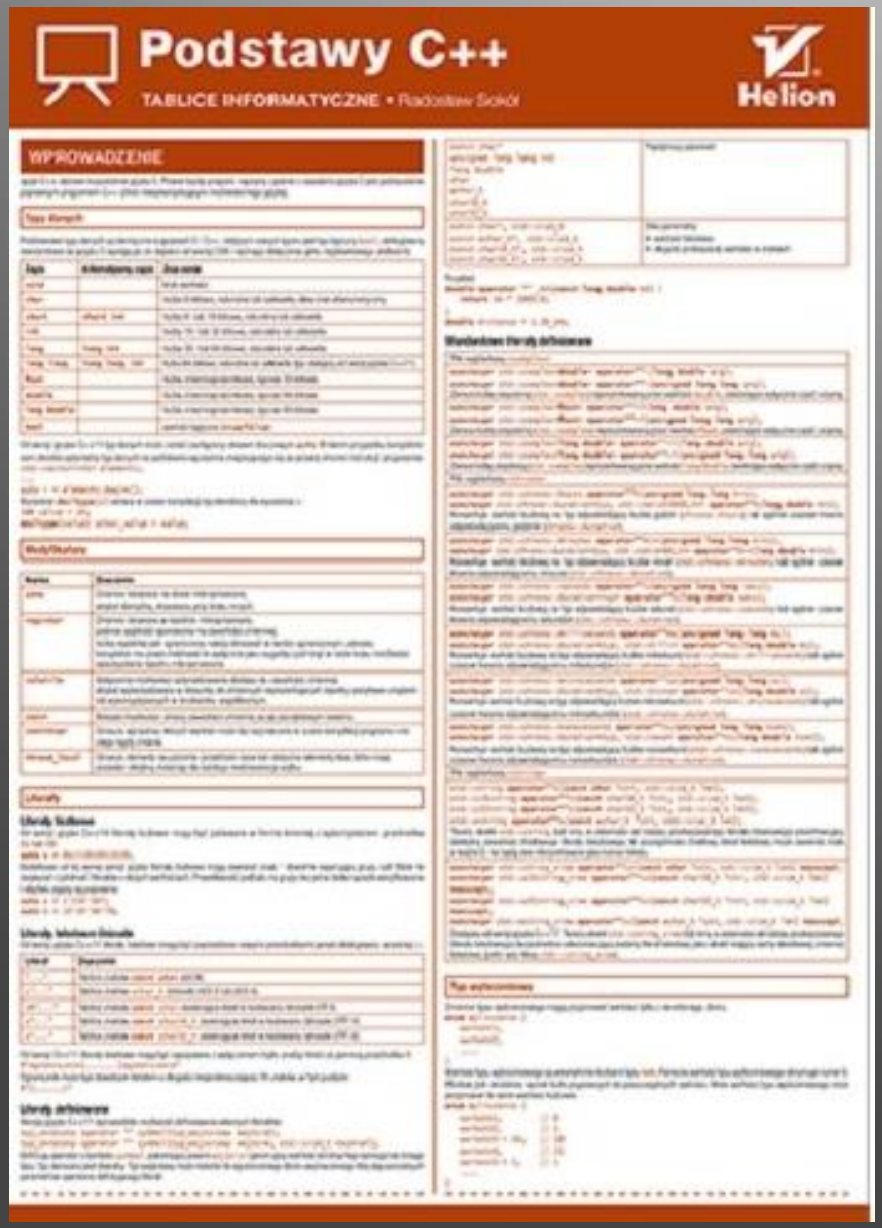
Literatura


język
C++
Bjarne
Stroustrup
50 lat WT


Radosław Sokół
Microsoft
Visual Studio 2012
Programowanie w C i C++
Poznaj środowisko Microsoft Visual Studio 2012
Odkryj uniwersalne narzędzia programistyczne
Naucz się programować w językach C i C++
Helion


KOMPENDIUM
programisty
C++BUILDER
Andrzej Daniluk
Zawiera CD

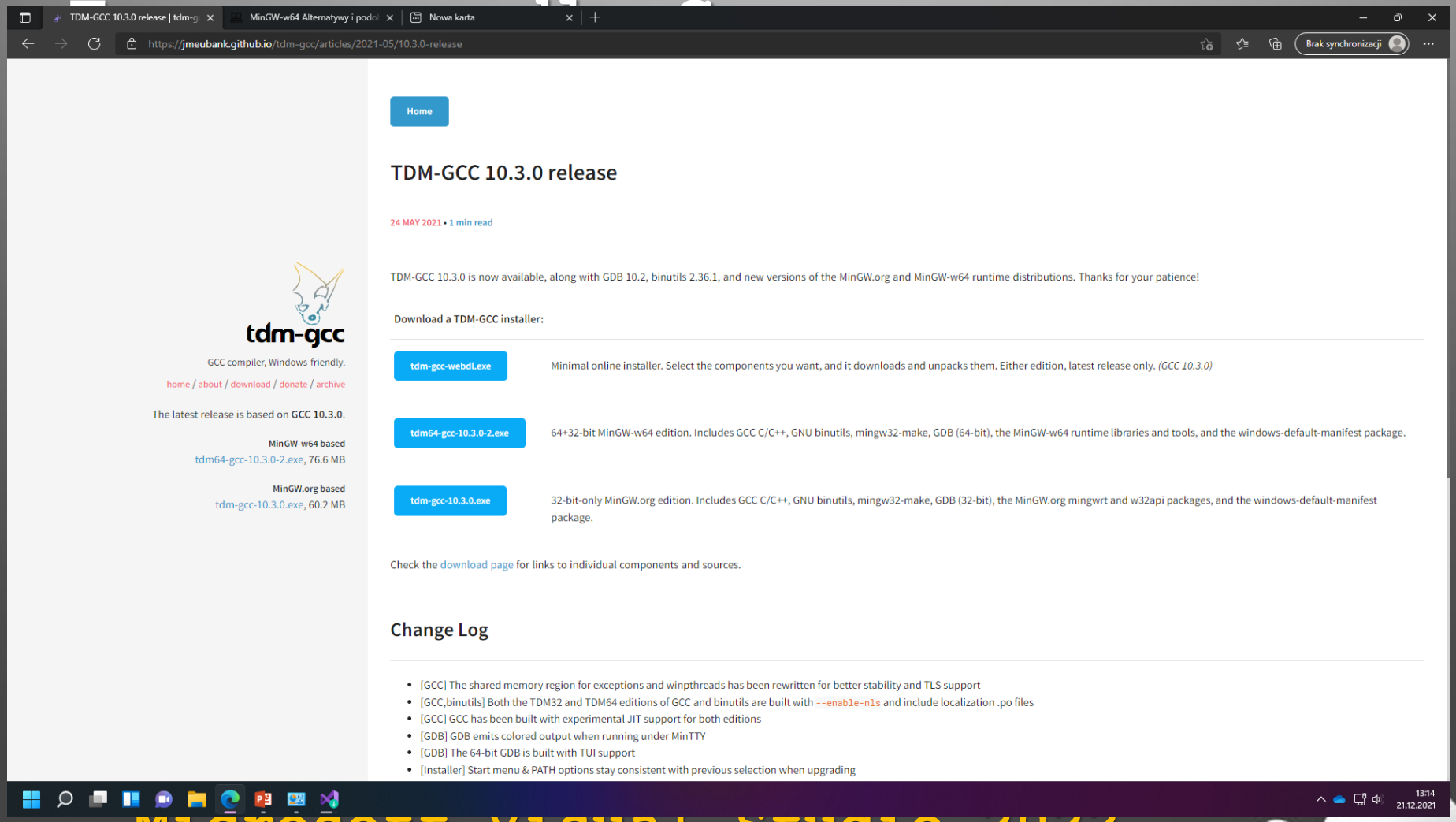
Literatura


Podstawy C++
TABLICE INFORMATYCZNE • Radosław Sokół
Helion
WPROWADZENIE


C++
TABLICE INFORMATYCZNE
Helion

20:44

Środowisko programowania

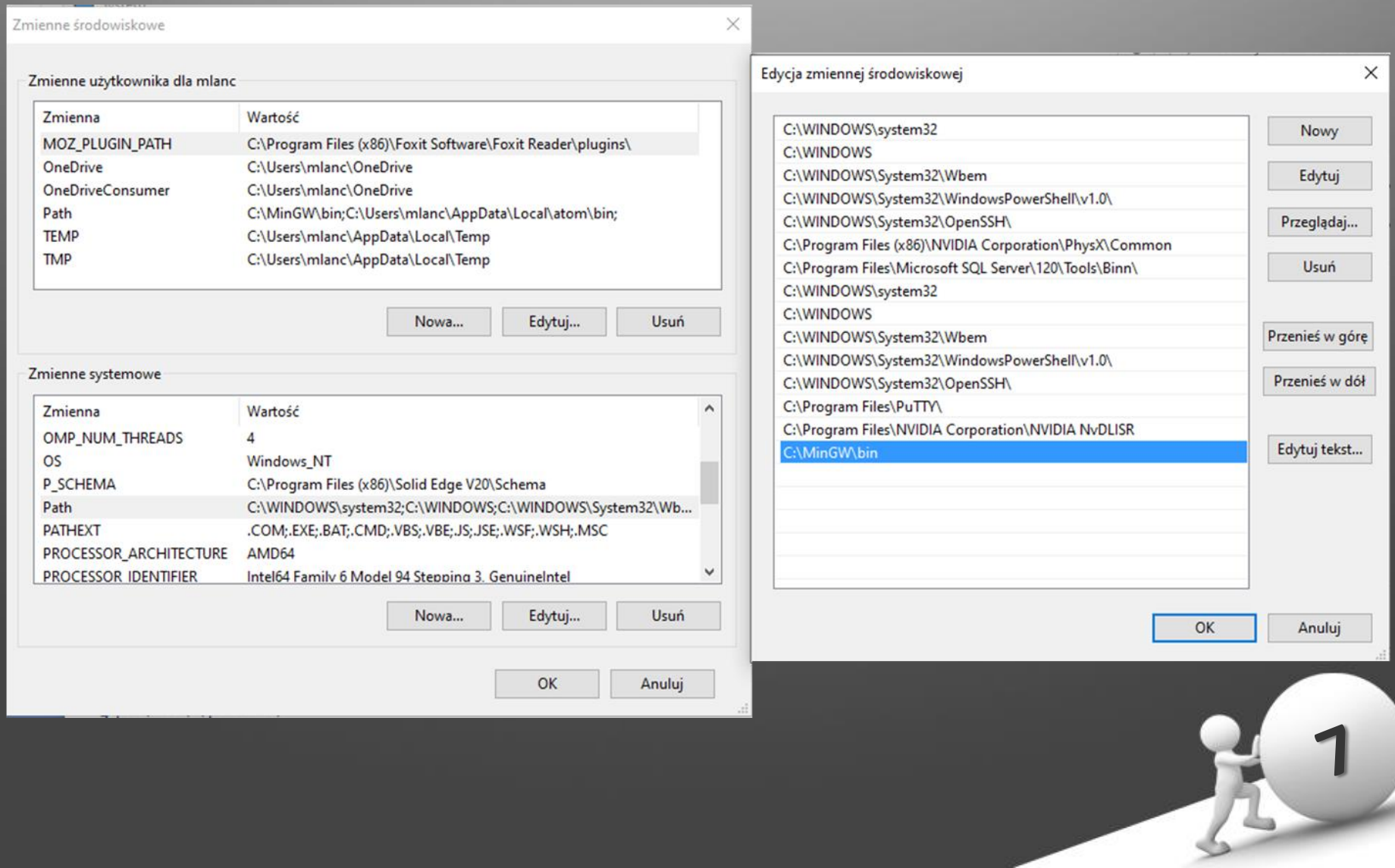


– Microsoft Visual Studio 2022

(Community – darmowa wersja)

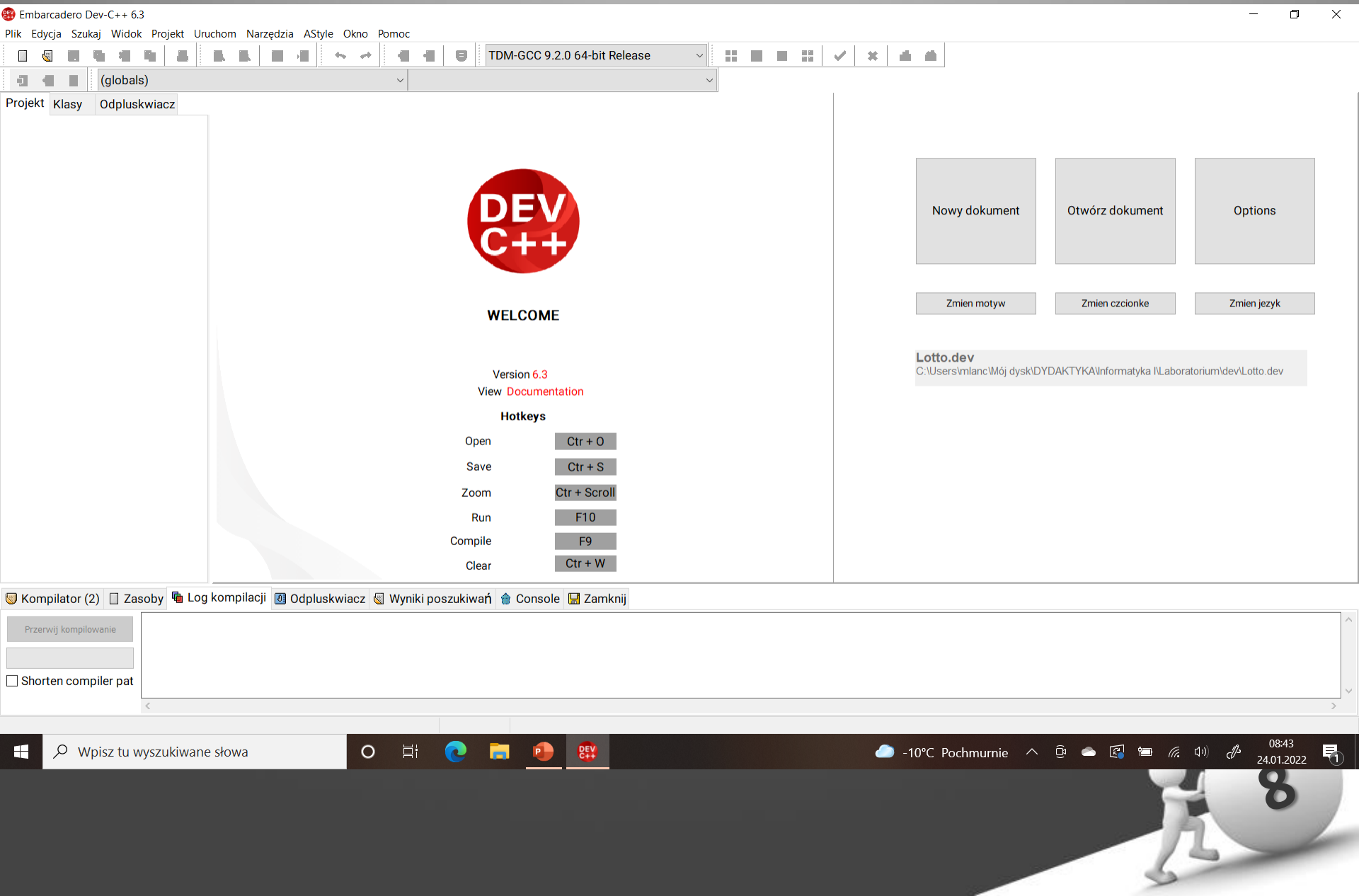
20:44

Kompilator C/C++



20:44

Dev C++


Embarcadero Dev-C++ 6.3
Plik Edycja Szukaj Widok Projekt Uruchom Narzędzia AStyle Okno Pomoc
TDM-GCC 9.2.0 64-bit Release
(globals)
Projekt Klasy Odpluskwiacz
DEV C++
WELCOME
Version 6.3
View Documentation
Hotkeys
Open Ctr + O
Save Ctr + S
Zoom Ctr + Scroll
Run F10
Compile F9
Clear Ctr + W
Nowy dokument
Otwórz dokument
Options
Zmien motyw
Zmien czcionke
Zmien jezyk
Lotto.dev
C:\Users\mlanc\Mój dysk\DYDAKTYKA\Informatyka I\Laboratorium\dev\Lotto.dev
Kompilator (2) Zasoby Log kompilacji Odpluskwiacz Wyniki poszukiwań Console Zamknij
Przerwij kompilowanie
Shorten compiler pat
Wpisz tu wyszukiwane słowa
-10°C Pochmurnie
08:43
24.01.2022

20:44

Microsoft Visual Studio 2020

Plik Edycja Widok Git Projekt Kompilowanie Debuguj Test Analiza Narzędzia Rozszerzenia Okno Pomoc Wyszukaj (Ctrl+Q) wyklad1_1

Debug x64 Lokalny debuger Windows Live Share

wyklad1_1.cpp

wyklad1_1 (Globalny zasięg)

```
1  // wyklad1_1.cpp : Ten plik zawiera funkcję „main”. W nim rozpoczyna się i kończy wykonywanie programu.
2  //
3
4   #include <iostream>
5
6  int main()
7   {
8       std::cout << "Hello World!\n";
9   }
10
11 // Uruchomienie programu: Ctrl + F5 lub menu Debugowanie > Uruchom bez debugowania
12 // Debugowanie programu: F5 lub menu Debugowanie > Rozpocznij debugowanie
13
14 // Porady dotyczące rozpoczynania pracy:
15 //   1. Użyj okna Eksploratora rozwiązań, aby dodać pliki i zarządzać nimi
16 //   2. Użyj okna programu Team Explorer, aby nawiązać połączenie z kontrolą źródła
17 //   3. Użyj okna Dane wyjściowe, aby sprawdzić dane wyjściowe kompilacji i inne komunikaty
18 //   4. Użyj okna Lista błędów, aby zobaczyć błędy
19 //   5. Wybierz pozycję Projekt > Dodaj nowy element, aby utworzyć nowe pliki kodu, lub wybierz pozycję Projekt > Dodaj istniejący elem
20 //   6. Aby w przyszłości ponownie otworzyć ten projekt, przejdź do pozycji Plik > Otwórz > Projekt i wybierz plik sln
21
```

100 % Nie znaleziono żadnych problemów W.: 1 Zn.: 1 SPACJE CRLF

Eksplorator rozwiązań

Przeszukaj: Eksplorator rozwiązań (Ctrl+;)

Rozwiązanie „wyklad1_1” (liczba projektów: 1 z 1)

wyklad1_1

Odwołania

Zależności zewnętrzne

Pliki nagłówkowe

Pliki zasobów

Pliki źródłowe

wyklad1_1.cpp

Eksplorator rozwiązań Zmiany GIT

Właściwości

Dane wyjściowe

Pokaż dane wyjściowe z: Debugowanie

```
Wątek 0x2bd8 zakończył działanie z kodem 0 (0x0).
„wyklad1_1.exe” (Win32): załadowano „C:\Windows\System32\kernel.appcore.dll”.
„wyklad1_1.exe” (Win32): załadowano „C:\Windows\System32\msvcrt.dll”.
„wyklad1_1.exe” (Win32): załadowano „C:\Windows\System32\rpcrt4.dll”.
Wątek 0xa34 zakończył działanie z kodem 0 (0x0).
Wątek 0x2044 zakończył działanie z kodem 0 (0x0).
Program „[13368] wyklad1_1.exe” zakończył działanie z kodem 0 (0x0).
```

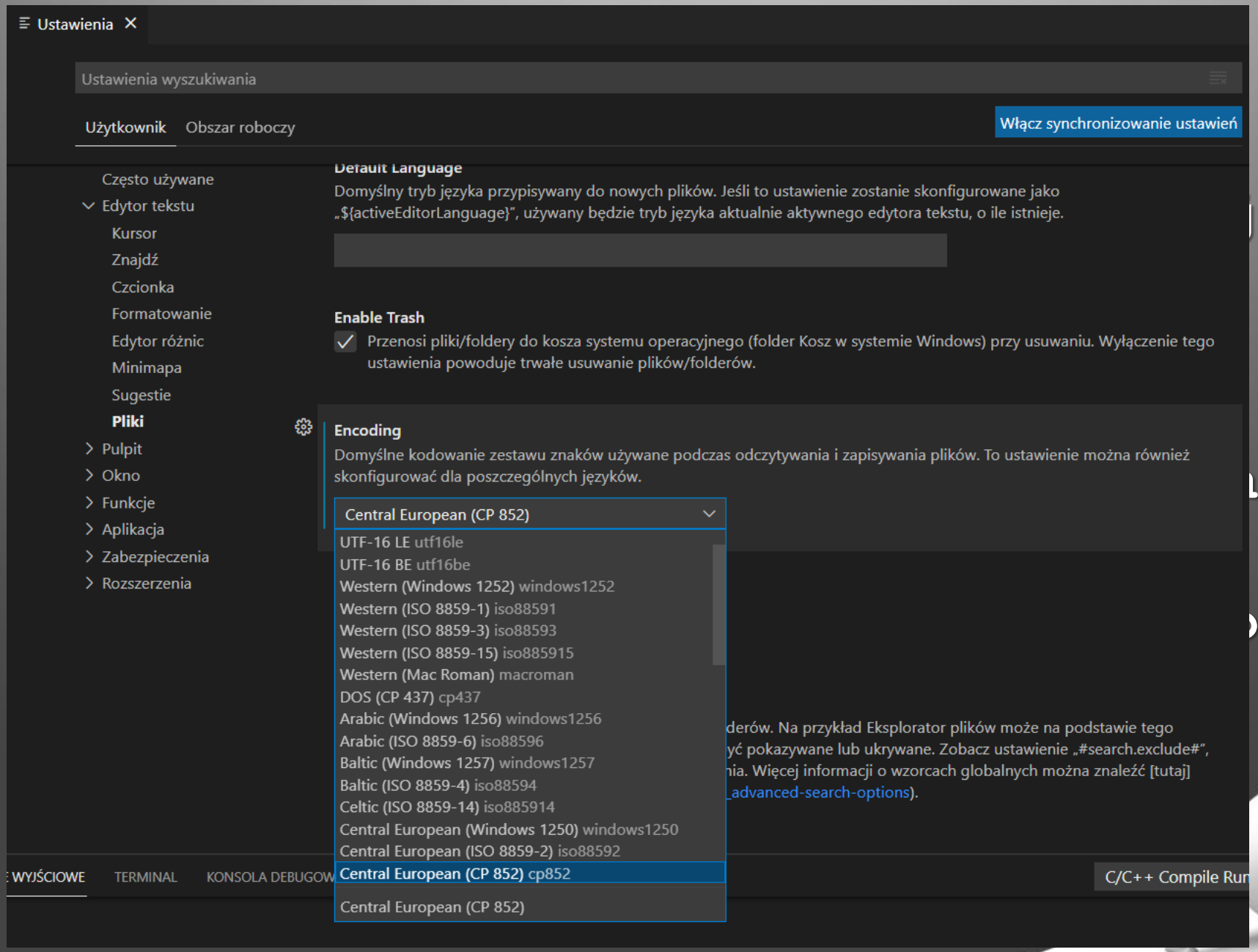
Gotowe Dodaj do kontroli źródła Wybierz repozytorium

Wpisz tu wyszukiwane słowa -5°C Przew. słonecz. 15:20 21.12.2021

na dysku HDD – ponad 8 GB dla programowania w C++

20:44

Polskie znaki w konsoli



10

Zakres materiału

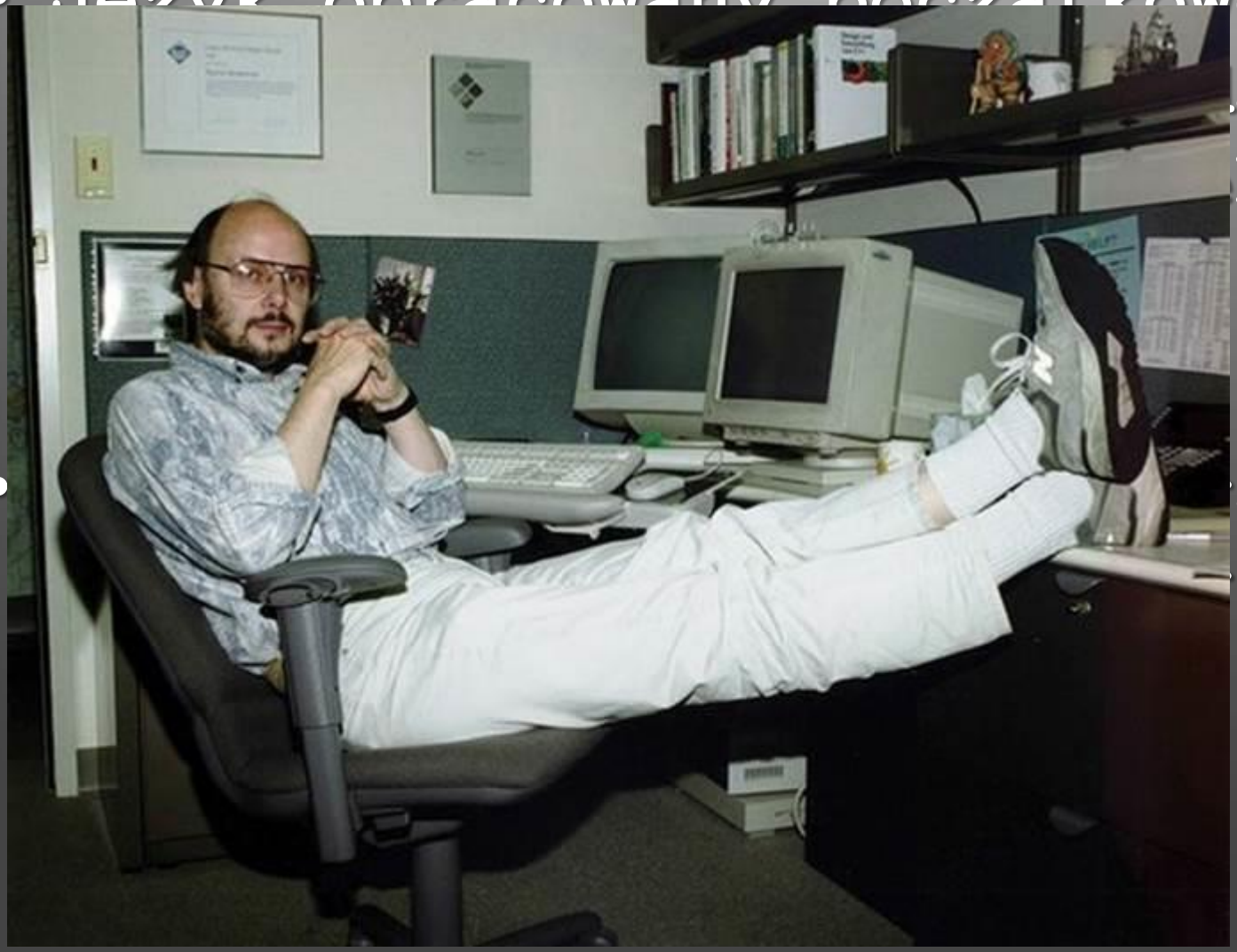
- Paradygmaty programowania.
- Programowanie strukturalne w języku C – przegląd podstawowych struktur danych i instrukcji sterujących.
- Złożone struktury danych: struktury, unie
- Obsługa wejścia-wyjścia, wykorzystanie manipulatorów.
- listy powiązane (linked lists), podwójnie powiązane listy (double linked lists)
- Zapis i odczyt danych z pliku.
- Wyrażenia regularne
- Obsługa wyjątków.

Zakres materiału

- Programowanie obiektowe w języku C++ - podstawy.
- Idea i pojęcia programowania obiektowego.
- Pojęcie klasy i obiektu.
- Przetwarzanie obiektów.
- Właściwości programowania zorientowanego obiektowo: hermetyzacja, dziedziczenie, polimorfizm.
- Funkcje i klasy zaprzyjaźnione.
- Przeciążanie operatorów i funkcji.
- Szablony funkcji i szablony klas.
- Standardowa biblioteka szablonów.
- Klasa string

Język C++

- Język opracowany początkowo w
  ający
  ktowego
- atoriach
  zez



Język C++

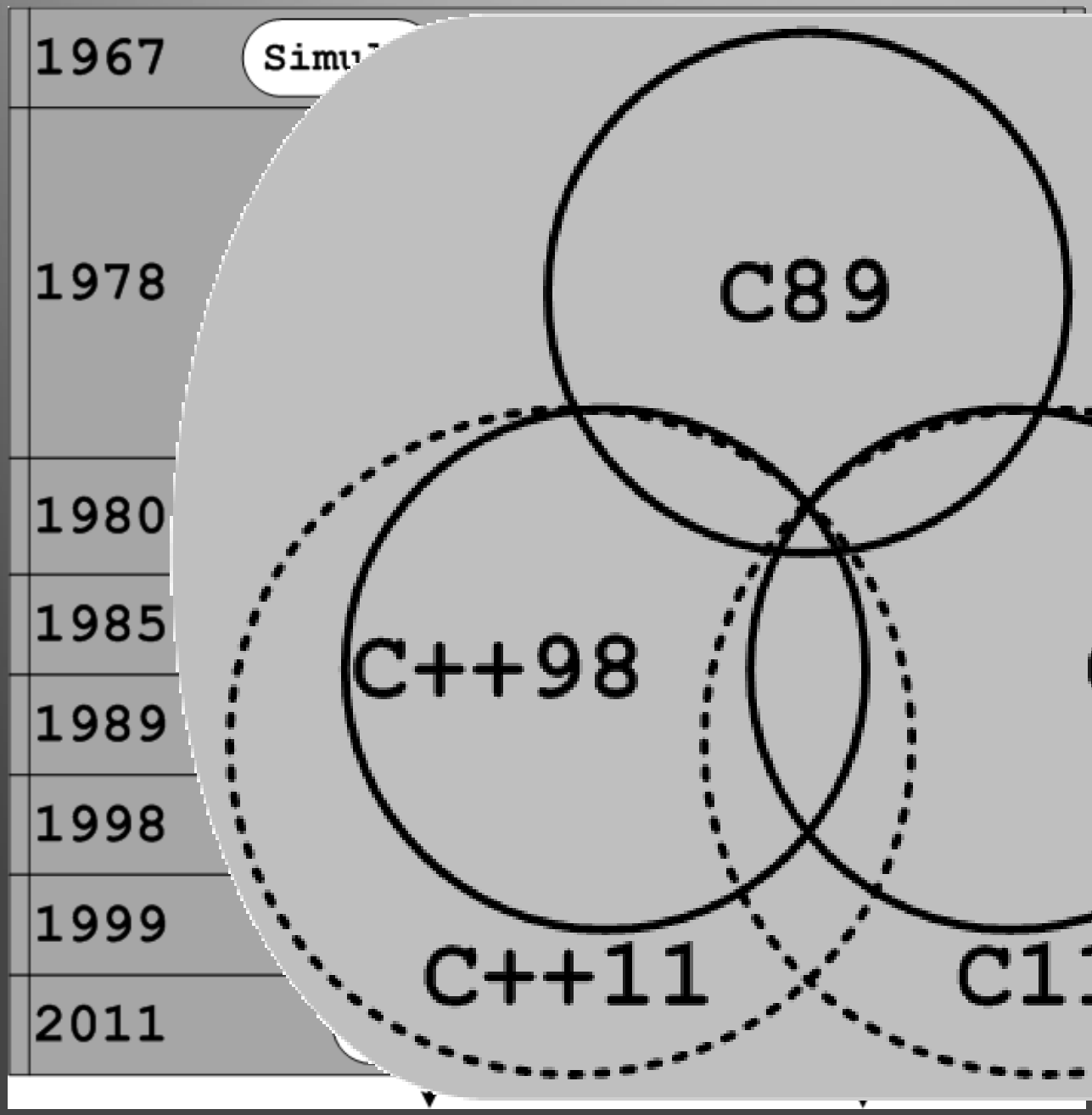
- Początkowo określany był jako „język C z klasami”
- Pod nazwą C++ zaczął funkcjonować od 1983 r. gdy po raz pierwszy został zastosowany poza BELL LABS
- „++” został zaczerpnięty z operacji inkrementacji i++, oznaczał lepszą wersję C
- Do 1998 r. zdefiniowany był w nieformalnym standardzie ARM (Annotated Reference Manual)

Język C++

C++

ges

do



Język C++

- Język C++ jest stale rozwijany, nowe wersje standardu publikowane są co około 3 lata
- Aktualna wersja C++20
- Znaczna część dostępnych pozycji literaturowych dotyczy wersji C++11 (była to duża aktualizacja standardu)
- Kolejna wersja planowana C++23

Język C++

Rok	Standard C++	Nazwa nieoficjalna	
1998	ISO/IEC 14882:1998	C++98	
2003	ISO/IEC 14882:2003	C++03	
2011	ISO/IEC 14882:2011	C++11	C++0x
2014	ISO/IEC 14882:2014	C++14	C++1y
2017	ISO/IEC 14882:2017	C++17	C++1z
2020	ISO/IEC 14882:2020	C++20	C++2a

Język C++

Styczeń 2020	Styczeń 2019	Język programowania	Pozycja	Zmiana
1	1	Java	16.896%	-0.01%
2	2	C	15.773%	+2.44%
3	3	Python	9.704%	+1.41%
4	4	C++	5.574%	-2.58%
5	7	C#	5.349%	+2.07%
6	5	Visual Basic .NET	5.287%	-1.17%
7	6	JavaScript	2.451%	-0.85%
8	8	PHP	2.405%	-0.28%
9	15	Swift	1.795%	+0.61%
10	9	SQL	1.504%	-0.77%

TIOBE Index for January 2020

Język C++

Feb 2021	Feb 2020	Programming Language	Ratings	Change
1	2	C	16.34%	-0.43%
2	1	Java	11.29%	-6.07%
3	3	Python	10.86%	+1.52%
4	4	C++	6.88%	+0.71%
5	5	C#	4.44%	-1.48%
6	6	Visual Basic	4.33%	-1.53%
7	7	JavaScript	2.27%	+0.21%
8	8	PHP	1.75%	-0.27%
9	9	SQL	1.72%	+0.20%
10	12	Assembly language	1.65%	+0.54%

TIOBE Index for February 2021

Kod w C++

- Kod programu w języku C++ pisze się w analogicznym układzie jak w języku C
  - Nagłówek – deklaracje bibliotek, instrukcje preprocesora, deklaracje funkcji, klas, zmiennych globalnych itd.
  - Funkcja `main()` i kod pozostałych funkcji programu
- Plik z kodem źródłowym ma rozszerzenie `.cpp` a plik nagłówkowy `.hpp`

Biblioteki

- W języku C++ podobnie jak w języku C do kodu programu podpina się biblioteki dostarczające:
  - Typy danych
  - Funkcje
  - Klasy
- W języku C++ można korzystać z bibliotek przeznaczonych dla języka C
- Ze względu na przyjętą nomenklaturę nazwy tych biblioteki rozpoczynają się od litery ’c’ i nie posiadają rozszerzenia ’.h’

Biblioteki C

C	C++	C	C++	C	C++
assert.h	cassert	ctype.h	*cctype*	errno.h	cerrno
float.h	cfloat	iso646.h	*ciso646*	limits.h	climits
locale.h	clocale	math.h	*cmath*	setjmp.h	csetjmp
signal.h	csignal	stdarg.h	*cstdarg*	stddef.h	cstddef
stdio.h	cstdio	stdlib.h	*cstdlib*	string.h	cstring
time.h	ctime	wchar.h	cwchar	cwtype.h	cwctype

Podstawowa biblioteka

- W języku C++ podstawową biblioteką wykorzystywaną w niemal każdym programie jest biblioteka wejścia-wyjścia <iostream>
- Biblioteka dostarcza funkcji obsługujących strumień wejścia i wyjścia
- Większość bibliotek w C++ dostarcza klasy umożliwiających tworzenie obiektów danego typu oraz dostarczające metody do ich obsługi

Klasa iostream

Klasa ios:
Ogólne właściwości strumienia, zawiera wskaźniki do obiektu klasy streambuf

Klasa streambuf:
zarządza pamięcią buforów wejściowych i wyjściowych

Klasa istream
metody do obsługi strumienia wejściowego

Klasa ostream
metody do obsługi strumienia wyjściowego

Klasa iostream dziedziczy metody do obsługi wejścia i wyjścia z klas istream i ostream

Klasa iostream

- Komunikacja z użytkownikiem w programach konsolowych realizowana jest za pomocą metod odwołania do właściwego strumienia oraz operatorów przekierowania danych
- Strumień wyjściowy realizowany jest za pośrednictwem metody cout
- Strumień wyjściowy obsługiwany jest za pomocą metody cin
- Operatory przekierowania << i >> określają kierunek przesyłania danych

Strumień wyjsciowy

```
1  #include <iostream>
2
3  int main()
4  {
5      std::cout << "Hello world!" << std::endl;
6  }
```

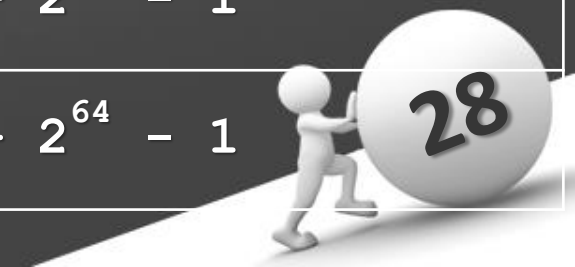
- **std** - oznacza odwołanie do standardowych elementów języka C++
- **::** - jest operatorem zasięgu, za pomocą którego odwołujemy się do metod z przestrzeni podanej przez dwukropkami
- **endl** – realizuje analogiczne zadanie do "**\n**" w języku C

Typy danych

- W języku C++ zmienne deklaruje się w sposób analogiczny jak w C
- Podaje się typ danych a następnie nazwę zmiennej
- Można inicjować zmienną przypisując po deklaracji wartość danego typu do zmiennej
- Podstawowe typy danych można modyfikować podobnie jak w języku C

Typ całkowity

Nazwa	*Wielkość (bajty)*	*Zakres*
short	*2*	$-2^{15} \div 2^{15} - 1$
int	*4*	$-2^{31} \div 2^{31} - 1$
long	*4*	$-2^{31} \div 2^{31} - 1$
long long	*8*	$-2^{63} \div 2^{63} - 1$
unsigned short	*2*	$0 \div 2^{16} - 1$
unsigned int	*4*	$0 \div 2^{32} - 1$
unsigned long	*4*	$0 \div 2^{32} - 1$
unsigned long long	*8*	$0 \div 2^{64} - 1$



Typ rzeczywisty

Nazwa	*Wielkość (bajty)*	*Zakres*
float	*4*	**pojedyncza precyzja** - dokładność 6 - 7 cyfr po przecinku - $3.4 \cdot 10^{\pm 38}$
double	*8*	podwójna precyzja - dokładność 15 - 16 cyfr po przecinku - $1.7 \cdot 10^{\pm 308}$
long double	*12*	liczby z ogromną dokładnością - 19 - 20 cyfr po przecinku - $1.7 \cdot 10^{\pm 308}$

Typ znakowy i logiczny

Nazwa	Wielkość (bajty)	Zakres
char	1	-128 ÷ 127
unsigned char	1	0 ÷ 255

Nazwa	Wielkość (bajty)	Zakres
bool	1	true (1) false (0)

Typy danych

```
1  #include <iostream>
```

C:\WINDOWS\system32\cmd.exe

```
Typy zmiennych:
int x=25
float y=3.1415
char z=S
bool w=1
1
1
1Press any key to continue . . .
1
14  }
```

Operatory

- `x+y` //dodawanie
- `x-y` //odejmowanie
- `x*y` //mnożenie
- `x/y` //dzielenie
- `x%y` //reszta z dzielenia (int)

Modyfikatory

```
#include
int main(
{
    int x
    for(i
        s
}
```

```
C:\Windows\system32\cmd.exe
x=0 y=1
x=1 y=2
x=2 y=3
x=3 y=4
x=4 y=5
x=5 y=6
x=6 y=7
x=7 y=8
x=8 y=9
x=9 y=10

Press any key to continue . . .
```

```
<< std::endl;
```

```
x -=y;
x/=y;
x%=y;
```

Inicjator auto

- Inicjując zmienne można skorzystać z inicjacji automatycznej
- Kompilator dobiera typ dla zmiennej na podstawie przypisanej do niej wartości

```
auto x = 10;          //typ int
auto y = false;       //typ bool
auto z = 'a';         //typ char
auto v = sqrt(x);     //typ double
auto w = 1.34;        //typ float
```

stałe

- W języku C++ pojęcie stałej ma szersze znaczenie jak W C
- W C++ wyróżnia się wartość stałej zwracanej przez wyrażenie, funkcję lub metodę
- **const** – przedrostek deklaracji stałej „zmiennej”
- **constexpr** – przedrostek wyrażenia, funkcji lub metody zwracającej wartość stałą

stałe

```
1  #include <iostream>
2  #define w 15;
3  constexpr int iloczyn(const int a, int b)
4      {
5          return a*b;
6      }
7  int main()
8  {
9      const int x = 5;
10     int y = 10;
11     constexpr int z1 = x*x;
12     constexpr int z2 = x*y;//x*w
13     const int z3 = x*y;
14     const int z4 = iloczyn(x,y);
15     std::cout << z1 << "," << z2 << "," << z3 << "," << z4;
16 }
```

Złożone typy danych

```
#include <iostream>
using namespace std;
struct test{
    int A,B;
};
union test2
{
    int A;
    float B;
};

int main()
{
    test alfa;
    test2 beta;
    cout << "Podaj dane:" << endl;
    cout << "A=";
    cin >> beta.A;
    cout << "B=";
    cin >> beta.B;
    cout << "Podano A=" << beta.A << " i  B=" << beta.B;
}
```

ować

e z C) –

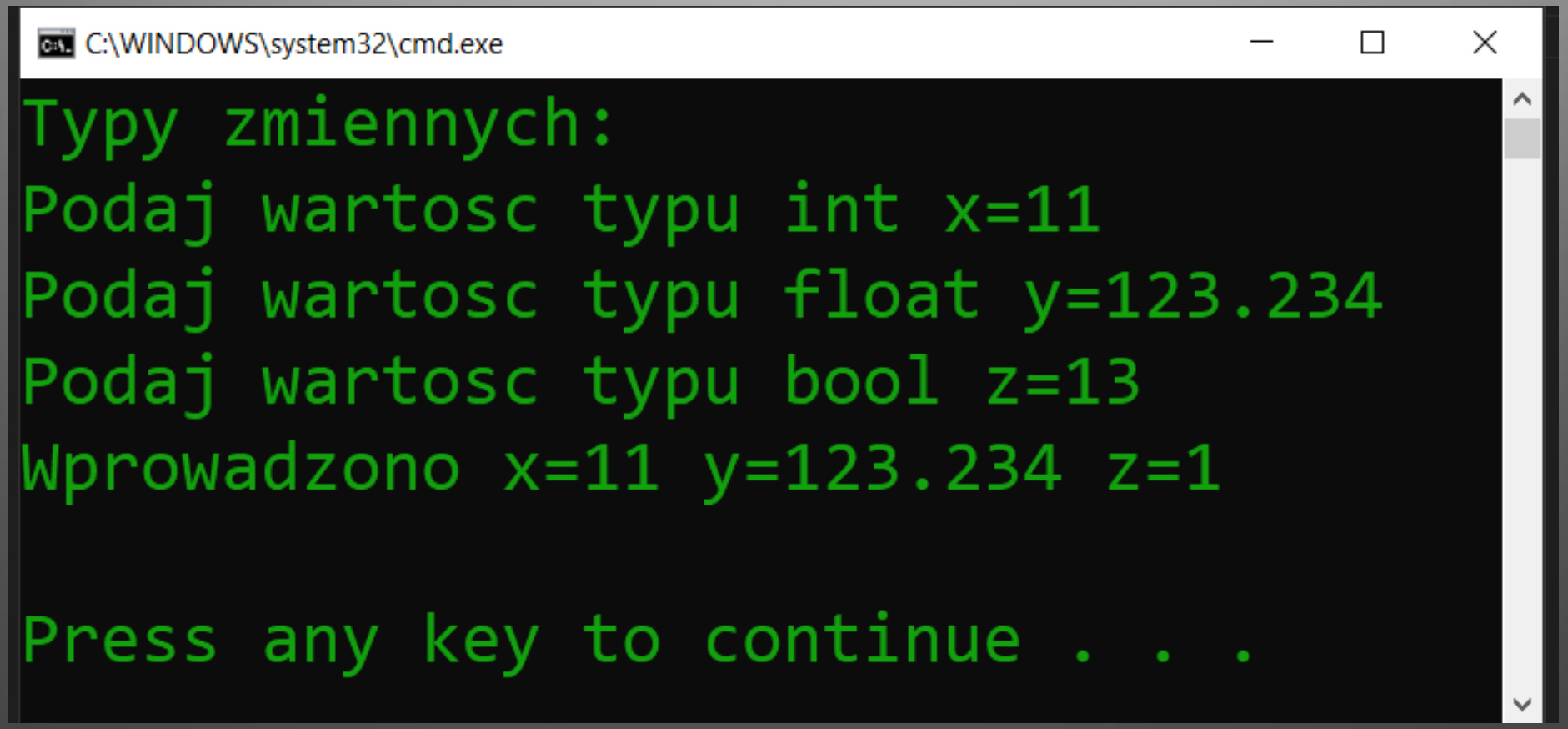
ymaga

słowa

on

znie przy

Strumień wejściowy



Przestrzenie nazw

- W języku C++ wprowadzono mechanizm ułatwiający pisanie dużych programów przez wielu programistów
- Możliwe jest grupowanie zmiennych, funkcji, klas, … w tak zwane przestrzenie nazw
- W ramach różnych przestrzeni nazw w jednym programie mogą funkcjonować elementy o tych samych nazwach
- Przestrzenie nazw deklaruje się w obszarze nagłówkowym

Przestrzenie nazw

```
1  #include <iostream>
2  namespace calk
3  {
4   int x=10;
5   void write(){std::cout << "Zmienna x=" << x << std::endl;}
6  }
7  namespace rzecz
8  {
9      float x=3.1415;
10     void write(){std::cout << "Zmienna x=" << x << std::endl;}
11 }
12 namespace inne
13 {
14     char x='x';
15     bool y=true;
16 }
17 int main()
18 {
19     using rzecz::write;
20     using namespace inne;
21     calk::write();
22     write();
23     std::cout << "x=" << x << " y=" << y << std::endl;
24 }
```

```
C:\WINDOWS\system32\cmd.exe
Zmienna x=10
Zmienna x=3.1415
x=x y=1

Press any key to continue . . .
```

Przestrzeń standardowa

- Większość podstawowych typów i funkcji w języku C++ jest zgrupowana w standardowej przestrzeni nazw
- W małych programach można ustawić domyślną przestrzeń

```
using namespace std;
```

- Dzięki temu nie trzeba przed funkcjami i typami z przestrzeni `std` umieszczać odwołania `std::`
- Nie jest to jednak zalecane

Przestrzeń standardowa

```
1  #include <iostream>
2  using namespace std;
3  int main()
4  {
5      int x;
6      float y;
7      bool z;
8      cout << "Typy zmiennych:" << endl;
9      cout << "Podaj wartosc typu int x=";
10     cin >> x;
11     cout << "Podaj wartosc typu float y=";
12     cin >> y;
13     cout << "Podaj wartosc typu bool z=";
14     cin >> z;
15     cout << "Wprowadzono x=" << x << " y=" << y << " z=" << z << endl;
16 }
```

Tablice

```
1  #include <iostream>
2
3
4
5
6
7
8
9
10
11
12
13
14 }
```

```
C:\WINDOWS\system32\cmd.exe
Hello World!
tab[5]=5
Twoje imie to: ☺
Podaj imie: Michał
Twoje imie to: Michał

Press any key to continue . . .
```

- Tablice można deklarować i inicjować

Decyzje

- Podejmowanie decyzji w języku C++ realizowane może być za pomocą instrukcji warunkowej `if()else` i wyboru `switch()case`

- Zasady ich użycia są identyczne z tymi poznanymi w języku C

Decyzje

Operatory porównania i logiczne

```
x==y        //równy
x!=y        //różny
x<y         //mniejszy
x>y         //większy
x<=y        //mniejszy lub równy
x>=y        //większy lub równy
&&          //iloczyn i (AND)
||          //suma  lub (OR)
!           //negacja (NOT)
```

Decyzje

```cpp
#include <iostream>
#include <cmath>
int main()
{
```

C:\WINDOWS\system32\cmd.exe

```
Podaj liczbę całkowita: -45
Moduł x=45
Wybierz działanie:
a) kwadrat liczby
b) pierwiastek kwadratowy liczby
:.> a
x^2=2025

Press any key to continue . . .
```

```cpp
        default:
        std::cout << "Błędny wybór opcji programu." << std::endl;
    }
}
```

Powtarzanie kodu

- Pętla iteracyjna **for()**
- Pętla warunkowa **while()**
- Pętla warunkowa **do..while()**
- Instrukcje przerwania pętli:
  - **continue;**
  - **break;**
  - **return;**

Pętla for() po tablicy

Pętla iteracyjna for() w języku C++ może zostać zdefiniowana do iteracji po elementach pętli bez konieczności indeksowania

```
1  #include <iostream>
2
3  int main
4  {
5      floa
6      size
7      for(
8
9      std::cout << std::endl;
10     for(auto i:tab)
11         std::cout << i << " ";
12     std::cout << std::endl;
13 }
```

```
C:\Windows\system32\cmd.exe
1.23 3 12.34 23 45 123.45
1.23 3 12.34 23 45 123.45

Press any key to continue . . .
```

funkcje

- Każdy program w języku C/C++ składa się z szeregu powiązanych ze sobą funkcji
- Główną funkcją każdego programu jest `main()`
- Pisząc kod programu poszczególne jego zadania zazwyczaj wydziela się do funkcji które są następne wywoływane zgodnie z przewidzianym algorytmem działania

funkcje

- Każda funkcja (poza `main()`) powinna zostać najpierw zadeklarowana
- Deklaracja zawiera informacje o:
  - typie danych zwracanym przez funkcję,
  - nazwie funkcji,
  - liście typów parametrów wejściowych.
- Język C++ pozwala na przeciążanie funkcji

funkcje

```
1  #include <iostream>
2  int dodaj(int,int);
3  float dodaj(float,float);
4  fl
5  fl
6  in
7  {
8
9
10
11
12
13
14 }
15 float dodaj(float x, float y){ return x+y;}
16 int dodaj(int x, int y){ return x+y;}
17 float dodaj(int x, float y){ return x+y;}
18 float dodaj(float x, int y){ return x+y;}
```

```
C:\WINDOWS\system32\cmd.exe
a+b=15
x+y=-210.994
x+b=33.456
a+y=-229.45

Press any key to continue . . .
```

Funkcja main()

- Parametry wejściowe tak jak każda funkcja deklarowana w C++
- Funkcja main() może posiadać dwa parametry
- Pierwszy typu int przechowuje informacje o liczbie parametrów
- Drugi jest wskaźnikiem na tablicę znakową i przechowuje ciągi tekstowe poszczególnych parametrów
- Pierwszym parametrem jest nazwa programu – indeks 0

Funkcja main()

```
1  #include <iostream>
2
3  int main(int numer, char *param[])
4  {
5      for(int i=0;i<numer;i++)
6          std::cout << "parametr " << i << ": " << param[i] << std::endl;
7
8  }
```

```
C:\Windows\system32\cmd.exe
parametr 0: .\kod7.exe
parametr 1: Alfa
parametr 2: 23
parametr 3: Ala ma kota
parametr 4: 23.123

Press any key to continue . . .
```

Plik nagłówkowy

- Podobnie jak w języku C kod programu można podzielić na kilka osobnych plików
- Podstawowym podział jest realizowany na plik nagłówkowy (.hpp) i źródłowy (.cpp)
- Kod programu można podzielić zadaniowo i zapisać w osobnych parach (.hpp i .cpp)
- Plik nagłówkowy zawiera instrukcje preprocesora (podpięcie bibliotek, plików klas), deklaracji zmiennych i stałych globalnych oraz deklaracje funkcji

Plik nagłówkowy

kod6.cpp ● kod6.hpp

wyklad_1 > kod6.cpp > ...

```cpp
#include "kod6.hpp"
int main()
{
    dane();
    std::cout << "Suma: " << suma();
}
void dane()
{
    srand(time(0));
    for(int i=0;i<20;i++)
        x[i] = rand()%21;
}
int suma()
{
    int n=0;
    for(int i=0;i<20;i++)
        n+=x[i];
    return n;
}
```

kod6.hpp

wyklad_1 > kod6.hpp > ...

```cpp
#ifndef KOD6_HPP
#define KOD6_HPP

#include <iostream>
#include <ctime>
#include <cstdlib>
int suma();
void dane();
int x[20];

#endif
```

Złożone typy danych

- Znane z języka C złożone typy danych:
  - **struct** - struktura
  - **union** - unia
  - **enum** - typ wyliczeniowy

- **klasa** - deklaratywny typ obiektowy

Metody wejścia i wyjścia

```
1   #include <iostream>
2
3   int main()
4   {
5       int x[] = {2,3,4,5,12,43,121};
6       char tekst[]="Ala ma kota";
7       std::cout.put(230);
8       std::cout.put('\t').put('A').put('\n');
9       for(auto i:x)
10          std::cout.write((char*)&i,sizeof(i));
```

```
C:\Windows\system32\cmd.exe
Š       A
☻   ♥   ♦   ♣   ♀   +   y
Ala ma kota
Press any key to continue . . .
```

Wyjściowy system liczbowy

```cpp
#include <iostream>

int main()
{
    float x[] = {24.34,12.3,4.34,54.5,1.22,43.43,.121};
    for(auto i:x){
        std::dec(std::cout);
        std::cout << i << "\t";
```

```
C:\Windows\system32\cmd.exe
24.34   24.34   24.34
12.3    12.3    12.3
4.34    4.34    4.34
54.5    54.5    54.5
1.22    1.22    1.22
43.43   43.43   43.43
0.121   0.121   0.121

Press any key to continue . . .
```

te do
można
pomocą
liczba

Szerokość pola danych

```
1  #include <iostream>
2
3  int main()
4  {
5      int x[] = {24,123,434,545,122,4343,121};
6      for(auto i:x){
7          std::dec(std::cout); std::cout.width(6); std::cout << i;
8          std::cout << std::oct; std::cout.width(8); std::cout << i;
9          std::cout << std::hex; std::cout.width(12); std::cout << i <<std::endl;
10     }
11 }
```

```
C:\Windows\system32\cmd.exe
    24      30          18
   123     173          7b
   434     662         1b2
   545    1041         221
   122     172          7a
  4343   10367        10f7
   121     171          79

Press any key to continue . . .
```

tylko na
nych wysyłany do
wego, następny
ny w sposób

Wypełnienie

```cpp
#include <iostream>

int main()
{
    int x[] = {24,123,434,545,122,4343,121};
    for(auto i:x){
        std::dec(std::cout); std::cout.fill(26); std::cout.width(6); std::cout << i;
        std::cout << std::oct; std::cout.width(8); std::cout.fill('.'); std::cout << i;
        std::cout.fill('_'); std::cout << std::hex; std::cout.width(12); std::cout << i <<std::endl;
    }
}
```

```
C:\Windows\system32\cmd.exe
→→→→24......30_____________18
→→→123.....173_____________7b
→→→434.....662_____________1b2
→→→545....1041_____________221
→→→122.....172_____________7a
→→4343...10367_____________10f7
→→→121.....171_____________79

Press any key to continue . . .
```

ładowej

nienić znak

pól

float - precyzja

C:\Windows\system32\cmd.exe

```
precyzja(6)                    3.14159                    3.14159
precyzja(7)                   3.141593                   3.141593
precyzja(8)                  3.1415927                  3.1415926
precyzja(9)                 3.14159265                 3.14159264
precyzja(10)               3.141592654                3.141592644
precyzja(11)              3.1415926536               3.1415926436
precyzja(12)             3.14159265359              3.14159264359
precyzja(13)             3.14159265359              3.14159264359
precyzja(14)            3.1415926535898            3.1415926435898
precyzja(15)           3.14159265358979           3.14159264358979
precyzja(16)          3.141592653589793          3.141592643589794
precyzja(17)         3.1415926535897931         3.1415926435897941
precyzja(18)        3.14159265358979312        3.14159264358979408
precyzja(19)       3.141592653589793116       3.141592643589794078
precyzja(20)       3.141592653589793116       3.141592643589794078

Press any key to continue . . .
```

```
17          std::cout << pi2 << std::endl;
18      }
```

$\pi \approx 3{,}14159265358979323846264338327950288419716 9\ldots$

Sterowanie wyświetlaniem

- Precyzyjne sterowanie sposobu wyświetlania różnych wartości służy funkcja składowa `.setf()`
- Usuwanie nadanego formatowania realizuje metoda `.unsetf()`
- Funkcja ta korzysta ze stałych zdefiniowanych w klasie `ios_base`
- Konieczne jest odwołanie standardowego wejścia do klasy `ios_base`

```
using std::ios_base
```

lub

```
using namespace std;
```

Sterowanie wyświetlaniem

- ios_base::boolalpha – wartości logiczne wyświetlane są jako **true** i **false**
- ios_base::showbase – wyświetla przedrostek systemu liczbowego
- ios_base::showpoint – wyświetla „.” oddzielającą część ułamkową (wraz z zerami)
- ios_base::uppercase – przy zapisie hex korzysta z dużych liter
- ios_base::showpos – przy liczbach dodatnich wyświetla „+”

Sterowanie wyświetlaniem

```
#include <iostream>

C:\WINDOWS\system32\cmd.exe
Test zmiennej bool: 1
Test zmiennej bool: true
Test zmiennej int(hex): 8a7
Test zmiennej int(HEX): 8A7
Test zmiennej int(HEX): 0X8A7
Test zmiennej int(hex): 0x8a7

Press any key to continue . . .
                                                    dl;
                                                    dl;
                                                    dl;
    std::cout.unsetf(ios_base::uppercase);
    std::cout << "Test zmiennej int(hex): " << i << std::endl;
}
```

Sterowanie wyświetlaniem

Funkcja setf() dla pewnych stałych może przyjmować drugi parametr (stałą) – uszczegółowiający

```
cout.setf(st.szczegółowa, st.głowna);
```

ios_base::floatfield	ios_base::fixed	Notacja stałoprzecinkowa
	ios_base::scientific	Notacja naukowa
ios_base::adjustfield	ios_base::left	Wyrównanie do lewej
	ios_base::right	Wyrównanie do prawej
	ios_base::internal	Wyrównanie do lewej dla znaku lub przedrostka systemu liczbowego, wyrównanie do prawej dla wartości

Sterowanie wyświetlaniem

```cpp
#include <iostream>

int main()
{
    using std::ios_base;
    int X = 23687;
    ios_base::fmtflags styl = ios_base::uppercase;
    std::cout.setf(styl);
    std::cout << "X (HEX) = ";
    std::cout << std::hex;
    std::cout << X << std::endl;
```

```
C:\Windows\system32\cmd.exe
X (HEX) = 5C87
X (hex) = 5c87

Press any key to continue . . .
```

Manipulatory

- W zależności od stosowanego kompilatora dostępne są ***manipulatory*** realizujące wybrane zadania ustawiane za pomocą funkcji `setf()`

```
cout << manipulator;
```

- Jeżeli dany manipulator jest niedostępny konieczne jest skorzystanie z funkcji `setf()`

Manipulatory

boolalpha	setf(ios_base::boolalpha)
noboolalpha	unsetf(ios_base::boolalpha)
showbase	setf(ios_base::showbase)
noshowbase	unsetf(ios_base::showbase)
showpoint	setf(ios_base::showpoint)
noshowpoint	unsetf(ios_base::showpoint)
showpos	setf(ios_base::showpos)
noshowpos	unsetf(ios_base::showpos)
uppercase	setf(ios_base::uppercase)
nouppercase	unsetf(ios_base::uppercase)

Manipulatory

internal	setf(ios_base::internal, ios_base::adjustfield)
left	setf(ios_base::left, ios_base::adjustfield)
right	setf(ios_base::right, ios_base::adjustfield)
dec	setf(ios_base::dec, ios_base::basefield)
hex	setf(ios_base::hex, ios_base::basefield)
oct	setf(ios_base::oct, ios_base::basefield)
fixed	setf(ios_base::fixed, ios_base::floatfield)
scientific	setf(ios_base::scientific, ios_base::floatfield)

Manipulatory

W bibliotece <iomanip> dostępne są manipulatory realizujące tożsame zadanie z funkcjami:

- `cout.fill(char);`
  `cout << setfill(char) << …`

- `cout << fixed;`
  `cout.precision(int);`
  `cout << setprecision(int) << …`

- `cout.width(int);`
  `cout << setw(int) << …`

Manipulatory

```
#include <iostream>
#include <iomanip>
using namespace std;
int main()
{
    const char * osoby[] =
    const int premia[]={10
    for(int i=0;i<3;i++)
    {
        cout.fill('_');
        cout.width(18);
        cout << osoby[i] <
        cout.fill('.');
        cout.width(8);
        cout << premia[i]
    }
    cout << endl;
    for(int i=0;i<3;i++)
    {
        cout << setfill('_
        << setfill('.') <<
    }
}
```

```
C:\Windows\system32\cmd.exe
_______Jan Kowalski:....1000 zł
___Adam Wiśniewski:.....500 zł
________Ola Kwiatek:....1245 zł

_______Jan Kowalski:....1000 zł
___Adam Wiśniewski:.....500 zł
________Ola Kwiatek:....1245 zł

Press any key to continue . . .
```

Strumień wyjściowy

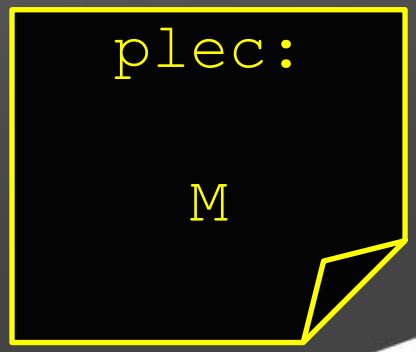
- Obiekt cin reprezentuje wejściowy strumień bitowy
- Standardowy strumień wejściowy generowany jest przez klawiaturę
- Dane ze strumienia wejściowego operatorem przypisania są konwertowane na typ żądany przez zmienną do której jest skierowany

Strumień wejściowy

```
char nazwisko[20];
int wiek;
char plec;
cin >> nazwisko >> wiek >> plec;
```

Pomijane są odstępy, znaki nowego wiersza i tabulacji

Nowak 56 M



Strumień wejściowy

```
1  #include <iostream>
2  int suma()
3  {
4      int sum=0,x;
5      while(std::cin >> x)
6      {
7          sum+=x;
8      }
9      return sum;
10 }
11 int main()
12 {
13     std::cout << "Dane wejściowe: ";
14     int sum = suma();
15     std::cout << "Suma=" << sum << std::endl;
16     std::cout << "Nowe dane wejściowe: ";
17     sum = suma();
18     std::cout << "Suma=" << sum << std::endl;
19 }
```

```
C:\Windows\system32\cmd.exe
Dane wejściowe: 1 2 3 4 5 a 23
Suma=15
Nowe dane wejściowe: Suma=0

Press any key to continue . . .
```

Liczby losowe

- Generator liczb pseudolosowych z języka C
  - Wymaga bibliotek <cstdlib> i <ctime>
  - Ustawienie ziarna generatora – srand(time(0)) srand(time(NULL))
  - Generowanie w zakresie 0-RAND_MAX rand() – RAND_MAX = 32767
  - Generowanie w zakresie (a,b) a+rand()%(b-a+1)

Liczby losowe

- **Generator niedetermistyczny w języku C++**
  - **Wymaga biblioteki <random>**
  - **deklaracja generatora:**

```
random_device nazwa;
```

  - **Wybór silnika generatora:**

```
typ nazwaSilnika(nazwa());
```

    - default_random_engine - **silnik domyślny**
    - minstd_rand - **minimalny standardowy** generatorminstd_rand0 - **minimalny standardowy generator**
    - mt19937 - **generator Mersenne Twister 19937**
    - mt19937_64 - **generator Mersenne Twister 19937**
    - ranlux24_base - **generator Ranlux o podstawie 24**
    - ranlux48_base - **generator Ranlux o podstawie 48**
    - ranlux24 - **generator Ranlux 24**
    - ranlux48 - **generator Ranlux 48**
    - knuth_b - **generator Knuth-B**

Liczby losowe

- Generator niedeterministyczny w języku C++
  - Możliwy wybór rozkładu
  - Rozkład całkowity <int>

```
uniform_int_distribution<int> nazwaR(a,b);
```

  - Rozkład rzeczywisty <float><double>

```
uniform_real_distribution<float> nazwaR(a,b);
```

- Generowanie liczby losowej

```
x = nazwaSilnika();
x = nazwaR(nazwaSilnika);
```

Przykład

```
C:\Windows\system32\cmd.exe
Liczba losowa: 4836268
Liczba z rozkładu całkowitego: 4
Liczba z rozkładu rzeczywistego: 6.73813
----------------------------------------
Liczba losowa: 15580392
Liczba z rozkładu całkowitego: 1
Liczba z rozkładu rzeczywistego: -9.06782
----------------------------------------
Liczba losowa: 1237236
Liczba z rozkładu całkowitego: 8
Liczba z rozkładu rzeczywistego: -4.22967
----------------------------------------
Liczba losowa: 12404067
Liczba z rozkładu całkowitego: -2
Liczba z rozkładu rzeczywistego: 1.59025
----------------------------------------
Liczba losowa: 8281623
Liczba z rozkładu całkowitego: 5
Liczba z rozkładu rzeczywistego: 7.89632
----------------------------------------

Press any key to continue . . .
```

```
1  #i
2  #i
3  us
4  ir
5  {
6
7
8
9
10
11
12
13
14
15
16 }
```

```
.0);

adInt(generator) << endl;
kladFloat(generator) << endl;
 << endl;
```