


# Podstawy Informatyki

04.10.2019

dr inż. Michał Rad

Plan:

1. Informacje organizacyjne
2. Język C++
    a) Dlaczego C++
    b) Informacje o języku
    c) Struktura programu
    d) Zmienne
    e) Stałe
3. Dane kontaktowe

# Organizacja zajęć

- 7 wykładów
- Strona wiki:
  - home.agh.edu.pl/~rad/wiki
- Ćwiczenia laboratoryjne
  - Obowiązuje wiedza z wykładu
- Zaliczenie na podstawie oceny z laboratorium

# Terminy wykładów

» 4.  10
» 11.10
» 25.10
» 8.  11
» 22.11
» 6.  12
» 13.12

# Ankieta

» Czy kiedyś miałeś/miałaś kontakt z programowaniem?

# Język C++

- Język wysokiego poziomu
- Daje możliwość programowania w różnych paradygmatach programowania (proceduralnym, obiektowym, generycznym) a także programowania współbieżnego
- Język kompilowalny, względnie szybki

# Od kodu do programu

1. Napisz poprawnie kod (w formie tekstowej)
2. Kompiluj
3. Zobacz wyniki,
   a. jeśli są błędy to popraw (idź do p. 1)
   b. jeśli nie ma błędów to .. bardzo dziwne ;)

# Organizacja kodu

- Kod może być zapisany w jednym bądź wielu plikach
- Przed właściwą kompilacją działa tak zwany preprocesor, który umożliwia m.inn. połączenie kodu zapisanego w wielu plikach w jeden kod.

# Przykład

```
#include <iostream>
int main()
{
std::cout<<”Moj pierwszy program”<<std::endl;
}
```

# Podstawowe zasady

(składnia – *syntax*)

- Instrukcja (*statment*) kończy się ’;’
- Komentarz rozpoczynamy od ’//’
- Blok programu ujmowany jest w nawiasy ’{ }’
- Duże i małe litery mają znaczenie (*case sensitive*)

# Słowa kluczowe

| | | | | |
|---|---|---|---|---|
| and | delete | long | signed | virtual |
| and_eq | do | mutable | sizeof | void |
| asm | double | namespace | static | volatile |
| auto | dynamic_cast | new | static_cast | wchar_t |
| bitand | else | not | struct | while |
| bitor | enum | not_eq | switch | xor |
| bool | explicit | operator | template | xor_eq |
| break | export | or | this | |
| case | extern | or_eq | throw | |
| catch | false | private | true | |
| char | float | protected | try | |
| class | for | public | typedef | |
| compl | friend | register | typeid | |
| const | goto | reinterpret_cast | typename | |
| const_cast | if | return | union | |
| continue | inline | short | unsigned | |
| default | int | | using | |

# Zmienne

- Aby używać zmiennych w C++ trzeba je najpierw zadeklarować.
- Deklaracja ustala typ zmiennej i jej nazwę
- Zmienną można od razu zainicjować (nadać wartość)
- Zmienne można deklarować globalnie lub lokalnie

# Przykład

```
#include <iostream>
int y=3;
int main()
{
int x=4;
std::cout<<”x + y to: ”<<x+y<<std::endl;
}
```

# Przykład

```
#include <iostream>
int y=3;
int main()
{
float x=4;
float wynik;
wynik= y/x;
std::cout<<”Wynik:”<<wynik<<std::endl;
}
```

# Deklaracja zmiennej

```
nazwa_typu  nazwa_zmiennej = wartość;
                                (opcjonalnie)
```

# Typy zmiennych

- **int** – liczby całkowite
- **float** – liczby zmiennoprzecinkowe
- **double** – liczby zmiennoprzecinkowe podwójnej precyzji
- **char** – znaki
- **bool** – wartość logiczna

# Operatory

| Operator | Składnia |
|---|---|
| Przypisanie | a = b |
| Dodawanie | a + b |
| Odejmowanie | a – b |
| Mnożenie | a * b |
| Dzielenie | a / b |
| Modulo (reszta z dzielenia) | a % b |
| Inkrementacja | a++ or ++a |
| Dekrementacja | a-- or --a |

| Operator | Składnia |
|---|---|
| Przypisanie z dodaniem | a += b |
| Przypisanie z odejmowaniem | a -= b |
| … mnożeniem | a *= b |
| … dzieleniem | a /= b |
| przypisanie z resztą z dzielenia | a %= b |

# Operatory porównania

| Operator | Składnia |
|---|---|
| Czy równy | a == b |
| Czy różny | a != b |
| Czy większy | a > b |
| Less than | a < b |
| Czy większy lub równy | a >= b |
| Czy mniejszy lub równy | a <= b |
| Logiczne NIE | !a |
| Logiczne ‚i' (AND) | a && b |
| Logical ‚lub' (OR) | a \|\| b |

# Operatory bitowe

| Operator | Składnia |
|---|---|
| Bitowe zaprzeczenie (NOT) | ~a |
| Bitowe ‚i' (AND) | a & b |
| Bitowe ‚lub' (OR) | a \| b |
| Bitowe XOR | a ^ b |
| Bitowe przesunięcie w lewo | a << b |
| Bitowe przesunięcie w prawo | a >> b |

| Operator | Składnia |
|---|---|
| Bitowe AND z przypisaniem | a &= b |
| Bitowe OR z przypisaniem | a \|= b |
| Bitowe XOR z przypisaniem | a ^= b |
| Bitowe przesunięcie w lewo z przypisaniem | a <<= b |
| Bitowe przesunięcie w prawo z przypisaniem | a >>= b |

# Zmienna

» Pewna ilość pamięci która służy do przechowywania wartości

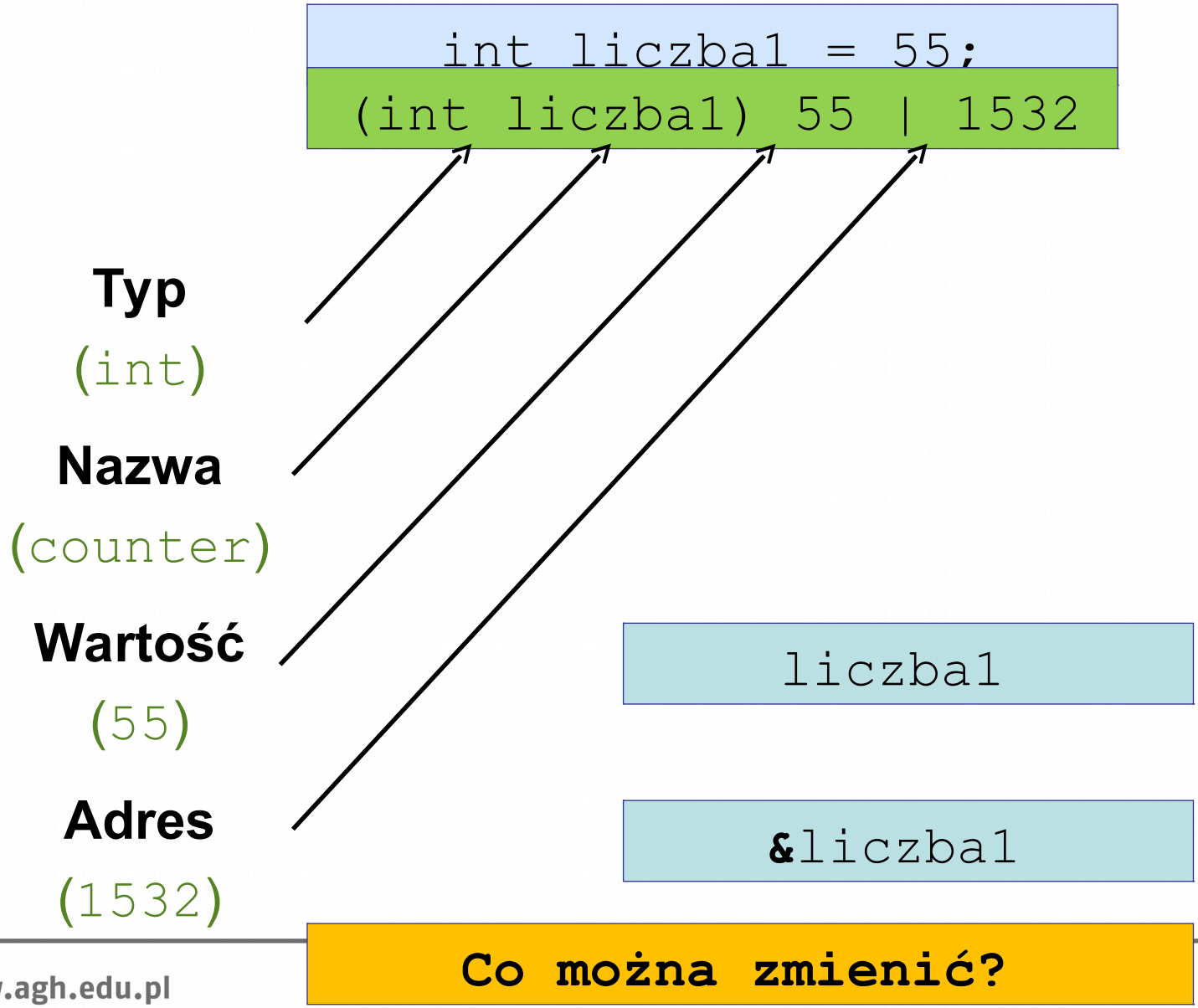
Stos (*stack*)

| | | | |
|---|---|---|---|
| 1500 | 1501 | 1502 | 1503 |
| 1504 | 1505 | 1506 | 1507 |
| 1508 | 1509 | 1510 | 1511 |
| 1512 | 1513 | 1514 | 1515 |
| 1516 | 1517 | 1518 | 1519 |
| (float x) ?? | | | 1518 |
| (int Licznik) 15 | | | 1524 |
| (double PI) 3.141593 | | | 1528 |
| 1532 | 1533 | 1534 | 1535 |

```
double PI = 3.141593;
```

```
int Licznik = 15;
```

```
float x;
```


int liczba1 = 55;
(int liczba1) 55 | 1532
Typ
(int)
Nazwa
(counter)
Wartość
(55)
Adres
(1532)
liczba1
&liczba1
Co można zmienić?

# Stałe

- To wartości których nie można zmienić w programie
- Stałe znakowe (bezpośredni wpis w programie)

```
int a = 5;
float b = 3.54;
char znak = 'r';
```

```
75              // dziesiętne
0113     //  ósemkowe
0x4b     // szesnastkowe
0b1001011 // binarne
     (gcc or C++14)
```

# Stałe

» Stałe preprocesora (definiowane na początku pliku)

```
#define piatka 5
#define t_i_p 3.5
#define znak 'r'
```

AGH

# Stałe

» Zmienna deklarowana ze słowem kluczowym *const*

```
const int a = 5;
const float b = 3.54;
const char znak = 'r';
```

# Zakres zmiennych

- Zmienne globalne
- Zmienne lokalne

# Dane kontaktowe


» dr inż. Michał Rad

» Budynek B1 pokój H20

» email: rad@agh.edu.pl

» www: http://home.agh.edu.pl/~rad/wiki