

Programowanie w języku C++ (EAR1S03006)

Politechnika Białostocka - Wydział Elektryczny
Automatyka i Robotyka, semestr III, studia stacjonarne I stopnia
Rok akademicki 2021/2022

Zajęcia nr 1 (06.10.2021)

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 - dodatkowe materiały do zajęć
- Konsultacje
 - poniedziałek, 16:00-17:30, WE-204 / Teams
 - wtorek, 14:00-15:30, WE-204 / Teams
 - sobota, 10:00-11:00, 13:45-15:00, WE-204 / Teams (zaoczne)

Program przedmiotu (1/2)

L.p.	Temat	Godz.
1.	Zajęcia organizacyjne. Operacje wejścia-wyjścia w języku C++, strumienie, sterowanie formatem.	2
2.	Klasa, obiekt, pola i metody. Prawa dostępu do składników klasy. Umieszczenie metod klasy.	2
3.	Dynamiczny przydział pamięci w języku C++. Konstruktor i destruktor. Wskaźniki do obiektów klasy. Tworzenie i likwidacja obiektów klasy przy użyciu wskaźników. Wydanie tematów projektów.	2
4.	Przeciążanie metod i operatorów.	2

Program przedmiotu (2/2)

L.p.	Temat	Godz.
5.	Dziedziczenie. Dostęp do pól i metod klasy bazowej i pochodnej. Konstruktory i destruktory przy dziedziczeniu. Dziedziczenie wielokrotne.	2
6.	Funkcje wirtualne i klasy abstrakcyjne.	2
7.	Obsługa plików w języku C++. Wyjątki w C++.	2
8.	Szablony funkcji i klas. Standardowa biblioteka wzorców STL.	4
9.	Budowa programów z wykorzystaniem elementów języka C++.	10
10.	Zaliczenie zajęć - przedstawienie i ocena projektów (programów komputerowych).	2

Literatura

1. B. Stroustrup: Programowanie. Teoria i praktyka z wykorzystaniem C++. Wydanie III. Helion, Gliwice, 2020.
2. N.M. Josuttis: C++. Biblioteka standardowa. Podręcznik programisty. Wydanie II. Helion, Gliwice, 2014.
3. J. Grębosz: Opus magnum C++11. Programowanie w języku C++. Wydanie II poprawione. Helion, Gliwice, 2020.
4. J. Grębosz: Opus magnum C++. Misja w nadprzestrzeń C++14/17. Helion, Gliwice, 2020.
5. S. Prata: Język C++. Szkoła programowania. Wydanie VI. Helion, Gliwice, 2012.

Warunki zaliczenia przedmiotu

- Obecność na zajęciach (więcej niż trzy nieusprawiedliwione nieobecności skutkują niezaliczeniem projektu)
- Realizacja w trakcie zajęć zadań przedstawionych przez prowadzącego
- Zaliczenie wszystkich sprawozdań w postaci prac domowych, co stanowi: 1/3 oceny końcowej. Sprawozdania będą zawierały zadania podzielone na oceny 3.0, 4.0 oraz 5.0. W semestrze przewiduje się 2-3 sprawozdania.
- Zaliczenie projektu końcowego. Ocena z projektu stanowi 2/3 oceny końcowej.
- Prowadzący zajęcia może przyznawać dodatkowe bonusy za aktywność na zajęciach.
- Ocena końcowa wyznaczana jest na podstawie sumy algebraicznej z ocen (projekt, prace domowe oraz bonusy).

Efekty uczenia się i system ich oceniania

Podstawę do zaliczenia przedmiotu (uzyskanie punktów ECTS) stanowi stwierdzenie, że każdy z założonych **efektów uczenia się** został osiągnięty w co najmniej minimalnym akceptowalnym stopniu.

EU3	potrafi projektować i implementować programy z wykorzystaniem technik programowania obiektowego w języku C++
EU4	potrafi stosować właściwe techniki programistyczne do realizacji programu

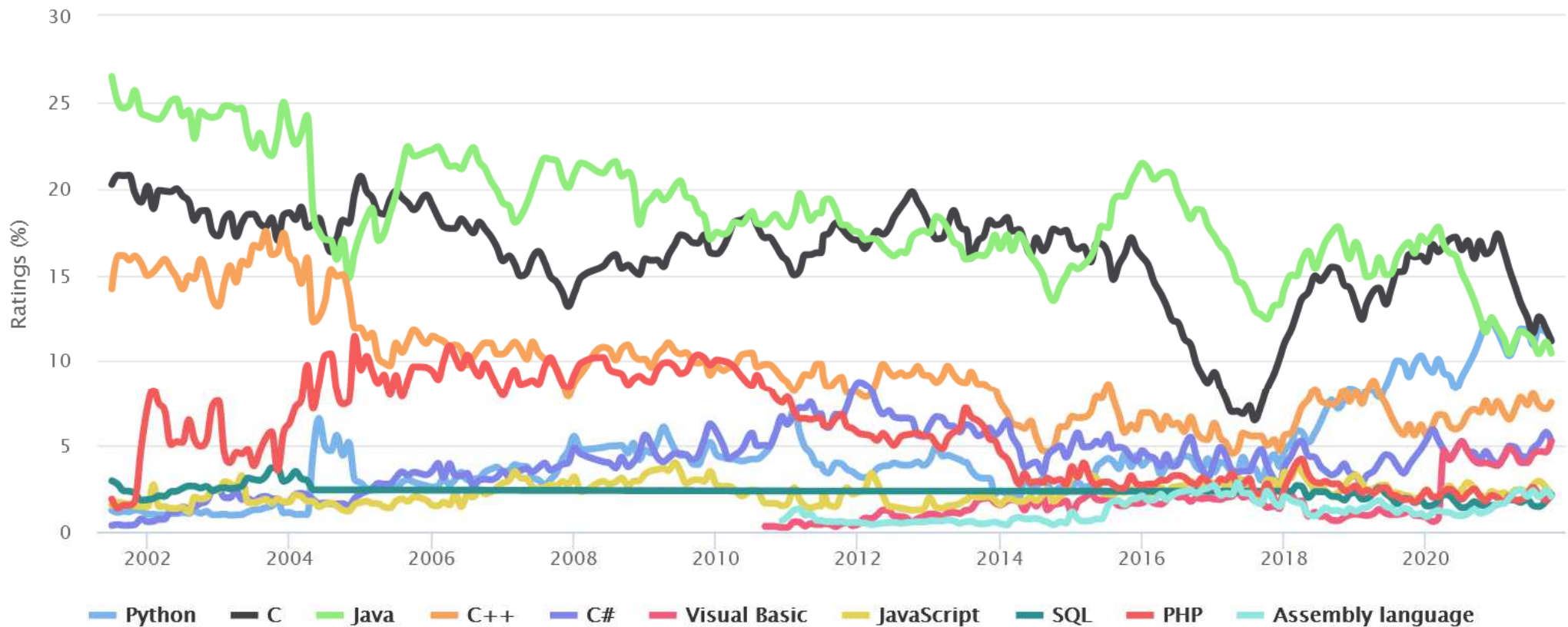
Język C++

- Stworzony na początku lat 80-tych XX wieku (od 1978 r.) przez **Bjarne Stroustupa** jako obiektowe rozszerzenie języka C
- Pierwsza nazwa: **C z klasami**
- Nazwę **C++** zaproponował Rick Mascitti w 1983 r. (w tym roku po raz pierwszy użyto tego języka poza laboratorium naukowym)
- **++** pochodzi od operatora inkrementacji
- Standard języka C++: rok 1998 - ISO/IEC 14882-1998 „Information Technology - Programming Languages - C++”
- Kolejne wersje standardu: 2003, 2011, 2014, 2017, 2020 (15.12.2020, C++20 - ISO/IEC 14882:2020)

TIOBE Programming Community Index

TIOBE Programming Community Index

Source: www.tiobe.com



Operacje wejścia-wyjścia w języku C++

- Operacje wejścia-wyjścia nie są zdefiniowane w języku C++
- Operacje te umożliwiają biblioteki standardowo dołączane przez producenta kompilatora:
 - stdio (język C)
 - stream (stara wersja iostream)
 - iostream

Strumienie:

- Wprowadzanie i wyprowadzanie informacji można potraktować jako strumień bajtów płynących od źródła do ujścia
- Strumienie w C++ realizowane są na zasadzie klas
- Wykorzystanie strumieni wymaga dołączenia pliku nagłówkowego:
`#include <iostream>` zamiast `#include <stdio.h>`

Operacje wejścia-wyjścia w języku C++

■ Predefiniowane strumienie w C++:

cout - związany ze standardowym urządzeniem wyjścia (ekran),
skrót od ang. **C**-onsole **OUT**-put

cin - związany ze standardowym urządzeniem wejścia (klawiatura),
skrót od ang. **C**-onsole **IN**-put

cerr - związany ze standardowym urządzeniem, na które chce się
wypisywać komunikaty o błędach (ekran) - strumień niebuforowany

clog - związany ze standardowym urządzeniem, na które chce się
wypisywać komunikaty o błędach (ekran) - strumień buforowany

■ Za wysyłanie i odbieranie informacji ze strumienia odpowiadają operatory **<<** i **>>**:

<< - operator odpowiadający za wysyłanie informacji do strumienia,
nazywany jest często operatorem **insert** - **wstawienia** (albo **put to**)

>> - operator odpowiadający za wczytywanie informacji, nazywany jest
operatorem **ekstrakcji** (**extract operator**) lub operatorem **get from**

Ogólne zasady dotyczące wyświetlania danych

- Liczby całkowite wyświetlane są w systemie dziesiętnym
- Zmienne typów `char`, `unsigned char` wyświetlane są jako pojedyncze znaki
- Liczby zmiennoprzecinkowe typów `float`, `double` wyświetlane są z dokładnością do 6 cyfr (6 cyfr części całkowitej i ułamkowej, bez zbędnych zer)
- **Wskaźniki** wyświetlane są w systemie szesnastkowym
- Zmienne typów `char *`, `unsigned char *` wyświetlane są jako łańcuchy znaków

Wyświetlanie danych

```
int    x = 10, y = 25;
float  z = 1.1234567;
char   txt[10] = "Napis";

cout << x;
cout << "x = " << x;
cout << x << y;
cout << x << " " << y;
cout << x << " " << y << endl;
cout << z << endl;
cout << txt << endl;
cout << txt << "\n";
```

```
10
x = 10
1025
10 25
10 25
1.12346
Napis
Napis
```

Ogólne zasady dotyczące wczytywania danych

- Białe znaki (spacja, tabulacja, enter) są ignorowane
- Liczby wczytywane są w systemie dziesiętnym
- Nie można umieszczać spacji pomiędzy znakiem liczby a jej wartością
- Wczytywanie liczby całkowitej jest kończone, gdy napotkany znak nie jest cyfrą
- W liczbach zmiennoprzecinkowych nie może występować spacja w środku
- Wczytywanie tekstów jest kończone po napotkaniu pierwszego białego znaku

Wczytywanie danych

```
int x, y;  
float z;  
  
cin >> x;  
cin >> x >> y;  
cin >> x >> z;
```

Program w języku C++

```
#include <iostream>

int main()
{
    std::cout << "Witaj świecie!" << std::endl;
}
```

- **std::** przed nazwami identyfikatorów **cout** i **endl** oznacza, że pochodzą one z biblioteki standardowej (dokładniej - z tzw. **przestrzeni nazw std**)
- **endl** - przejście do nowego wiersza, odpowiada **"\n"** w języku C
- W celu uniknięcia ciągłego pisania **std::** przed nazwami identyfikatorów umieszcza się w programie dyrektywę **using namespace std;**

Program w języku C++

Bez dyrektywy using:

```
#include <iostream>

int main()
{
    std::cout << "Witaj swiecie!" << std::endl;
}
```

Z dyrektywą using:

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Witaj swiecie!" << endl;
}
```

Formatowanie wyjścia

- Metody zmiany sposobu wyświetlania znaków:
 - funkcje `setf`, `unsetf` z klasy `ios` ustawiające odpowiednie flagi
 - funkcje składowe klasy `ios` zmieniające towarzyszące im parametry, np. szerokość, precyzję, itp.
 - manipulatory

- Manipulatory (modyfikatory):
 - specjalne wartości, które można wstawić do strumienia po to, aby wywołać zamierzony efekt polegający na zmianie sposobu formatowania
 - manipulatory działają trwale (nie dotyczy to manipulatora `setw`)

Manipulatory

`flush`

- opróżnia bufor wyjściowy

`endl`

- przejście do nowego wiersza - równoważne: `\n` i `flush`

Manipulatory

hex, dec, oct

- określają system liczbowy, w którym są wyświetlane / wczytywane liczby
- **hex** - system szesnastkowy
- **dec** - system dziesiętny
- **oct** - system ósemkowy

```
int x = 100;  
  
cout << x << " " << hex << x << " " << oct << x << endl;  
cout << x << endl;
```

```
100 64 144  
144
```

Manipulatory

showbase, noshowbase

- włącza / wyłącza wyświetlanie **0x** na początku liczby w systemie szesnastkowym i **0** na początku liczby w systemie ósemkowym
- działa tylko dla liczb całkowitych

```
int x = 10;

cout << dec << x << " " << showbase << x << noshowbase << endl;
cout << hex << x << " " << showbase << x << noshowbase << endl;
cout << oct << x << " " << showbase << x << endl;
```

```
10 10
a 0xa
12 012
```

Manipulatory

showpos, noshowpos

- włącza / wyłącza pokazywanie znaku liczby dodatniej

```
int x = 10;  
float y = 12.34567;  
  
cout << showpos << x << " " << noshowpos << x << endl;  
cout << showpos << y << " " << noshowpos << y << endl;
```

```
+10 10  
+12.3457 12.3457
```

Manipulatory

showpoint, noshowpoint

- włącza / wyłącza pokazywanie nieznaczących zer i kropki dziesiętnej

```
float x = 10;  
cout << showpoint << x << " " << noshowpoint << x << endl;
```

```
10.0000 10
```

Manipulatory

fixed, scientific

- **fixed** - włącza notację dziesiętną (tradycyjną)
- **scientific** - włącza notację wykładniczą (naukową)

```
float x = 100.123456;  
  
cout << x << endl;  
cout << fixed << x << endl;  
cout << scientific << x << endl;
```

```
100.123  
100.123459  
1.001235e+002
```


Manipulatory

setprecision(int n)

- określa dokładność wyświetlania liczb zmiennoprzecinkowych
- dla „trybu krótkiego” - łączna ilość cyfr przed i po kropce dziesiętnej
- dla **fixed** - ilość miejsc po kropce
- dla **scientific** - dokładność mantysy (ale nie wykładnika)
- działa ciągle
- wymaga dołączenia pliku nagłówkowego **iomanip**

```
float x = 12.123456;  
  
cout << setprecision(5) << x << endl;  
cout << fixed << x << endl;  
cout << scientific << x << endl;
```

```
12.123  
12.12346  
1.21235e+001
```

Manipulatory

setw(int n)

- ustawia szerokość wyświetlania liczb lub wczytywania tekstów
- dotyczy tylko najbliższej operacji wejścia-wyjścia
- wymaga dołączenia pliku nagłówkowego **iomanip**

```
int x = 12345;  
  
cout << x << endl;  
cout << setw(10) << x << endl;  
cout << x << endl;
```

12345

12345

12345

Manipulatory

setfill(char znak)

- ustawia znak będący wypełnieniem (domyślnie jest to spacja)
- działa ciągle
- wymaga dołączenia pliku nagłówkowego **iomanip**

```
int x = 12345;  
  
cout << setfill('*');  
cout << setw(10) << x << endl;  
cout << setw(10) << x << endl;
```

```
*****12345  
*****12345
```

Zastosowanie manipulatorów

```
#include <iostream>
#include <iomanip>
#include <ctime>
using namespace std;

#define N 4
#define M 5

int main()
{
    float T[N][M];
    int i, j;

    srand((unsigned int)time(NULL));

    for (i=0; i<N; i++)
        for (j=0; j<M; j++)
            T[i][j] = 100*(float)rand()/RAND_MAX - 50;
```

Zastosowanie manipulatorów

```
cout << fixed << setprecision(3);  
  
for (i=0; i<N; i++)  
{  
    for (j=0; j<M; j++)  
        cout << setw(10) << T[i][j];  
    cout << endl;  
}  
}
```

6.618	-27.541	-35.147	11.235	-28.539
32.482	-45.065	29.177	-42.822	44.375
49.017	46.915	36.261	13.588	12.038
-24.276	6.545	1.527	45.004	6.990