

# Programowanie Obiektowe

Marcin Kamil Bączyk

Wykład 1

3 października 2019

- Prowadzący wykład : mgr inż. Marcin Bączyk  
M.K.Baczyk@elka.pw.edu.pl, pokój 449, konsultacje w  
poniedziałki w godzinach 11-12.
- istnieje możliwość umawiania się na inny termin konsultacji
- strona przedmiotu:  
<http://staff.elka.pw.edu.pl/~mbaczyk1/PROE/index.html>
- strona będzie systematycznie aktualizowana, aż do końca semestru.

- PROE jest przedmiotem zaliczeniowym - ostatni możliwy termin oddawania projektów, poprawiania sprawdzianów itp. przypada na ostatni dzień semestru
- Przedmiot podzielony jest na dwie części : wykład (15) oraz laboratoria (14)
- W trakcie wykładu odbędą się dwa sprawdziany wykładowe po 25 punktów każdy
- W trakcie zajęć laboratoryjnych odbędzie się cztery zajęcia oceniane w skali 0-5 punktów każde
- W trakcie zajęć laboratoryjnych będą realizowane dwa małe projekty oceniane w skali 0-15 punktów
- Prowizoryczny harmonogram laboratorium dostępny na stronie przedmiotu
- Terminy sprawdzianów oraz zajęć laboratoryjnych ustalone zostaną w trakcie kolejnych zajęć.

- Aby zaliczyć przedmiot należy spełnić następujące warunki:
  - uzyskać łącznie minimum 25 punktów ze sprawdzianów wykładowych
  - uzyskać łącznie minimum 25 punktów z laboratorium
  - uzyskać 51 punktów z całego przedmiotu
- Progi ocen
  - $\geq 91$  - 5
  - $\geq 81$  - 4.5
  - $\geq 71$  - 4
  - $\geq 61$  - 3.5
  - $\geq 51$  - 3
  - $< 51$  - 2

- Terminy
  - wtorek 10:15 - 12:00 - 1 grupa
  - wtorek 16:15 - 18:00 - 1 grupa
  - piątek 12:15 - 14:00 - 1 grupa
- limit miejsc w grupie : 13 osób

## Treść wykładu

- Zapoznanie się z elementami projektowania obiektowego
- Zapoznanie się z możliwościami nowoczesnego języka C++
- Zapoznanie z możliwościami biblioteki standardowej C++

## Efekty kształcenia

- Zapoznanie z paradygmatem programowania obiektowego.
- Zdobyć umiejętności rozumienia kodu napisanego w C++.
- Zdobyć umiejętności pisania własnych aplikacji C++.
- Zdobyć umiejętności samodzielnego poszukiwania niezbędnych informacji.

## Projektowanie obiektowe

- Abstrakcja
- Hermetyzacja
- Polimorfizm
- Dziedziczenie

## Składnia języka C++

- Klasy
- Konstruowanie i niszczenie obiektów
- Szablony
- Obsługa wyjątków
- Zarządzanie pamięcią
- ...

- uczestnictwo w zajęciach laboratoryjnych i wykładach
- samodzielna realizacja projektów
- polecana literatura:
  - programowanie obiektowe
    - "Czysty kod" oraz "Czysta architektura" - Robert Martin
    - "Wzorce projektowe. Elementy oprogramowania obiektowego wielokrotnego użytku" - Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides
  - język C++
    - "Język C++. Kompendium wiedzy" - Stroustrup Bjarne
    - "Podstawy języka C++" - Stanley Lippman
    - "Język C++. Standardy kodowania. 101 zasad, wytycznych i zalecanych praktyk" - Herb Sutter, Andrei Alexandrescu
    - <https://en.cppreference.com/w/>
    - <https://stackoverflow.com/>



- Klasy oraz obiekty będące ich instancjami
- Dziedziczenie, dziedziczenie wielobazowe
- Konstruktory oraz destruktory
- Metody wirtualne klas
- Zarządzanie pamięcią
- Dynamiczna kontrola typów

## Klasa:

Jest to definicja wzorca (opis wszystkich danych i dostępnych metod) według którego będą tworzone obiekty danego typu. Innymi słowy klasa to nowy typ zmiennych definiowany przez użytkownika.

## Obiekt:

Obiekty są to instancje konkretnych klas (typów własnych).

Podobnie jak język C pozwala stworzyć pustą strukturę, język C++ pozwala stworzyć pustą klasę.

```
class First;  
class First {};  
  
int main(void)  
{  
    First f1;  
    return 0;  
}
```

- słowo kluczowe
- deklaracja klasy
- definicja klasy
- obiekt

Czy istnieje jakieś wytłumaczenie dla istnienia pustej klasy? Jaki jest jej rozmiar?

## typ prosty

**Typ nazwaZmiennej;**

np. float, double, int, ... oraz

np. klasy, struktury, ...

## typ wskaźnikowy

**Typ \* nazwaZmiennejWskaźnikowej = &nazwaZmiennej**

Typ wskaźnikowy oznacza adres pod jakim zapisana jest dana zmienna

## typ referencyjny

**Typ & nazwaZmiennejReferencyjnej = nazwaZmiennej**

Typ referencyjny należy rozumieć jako odnośnik do zmiennej

## L-referencja

**Typ & nazwaZmiennejReferencyjnej = nazwaZmiennej**

- definicja zmiennej referencyjnej zawsze wymaga podania obiektu (inicjacji)
- związku pomiędzy zmienną referencyjną a obiektem nie można zerwać
- nie można zdefiniować referencji do referencji
- nie można zdefiniować wskazania na referencję (ale referencję do wskazania już tak) - w związku z powyższym nie można używać tablic referencji

## R-referencja

**Typ && nazwaZmiennejReferencyjnej =  
zmiennaNieposiadajaNazwy**

```
int i = 1;           // i = 1
int *pi;
pi = &i;            // i = 1
*pi = 2;           // i = 2
// int & ri - Uwaga Bład
int & ri = i;      // i = 2
ri++;              // i = 3

First f;
First *pf = &f;
// First &rf;    //'rf': references must be initialized
First &rf = f;
```

Referencja wskazuje na dany obiekt, również wtedy gdy obiekt przestanie istnieć (np. pamięć zostanie zwolniona). Język C++ pozwala zwrócić referencję do obiektu tymczasowego - to nigdy nie kończy się dobrze.