Programowanie komputerów I

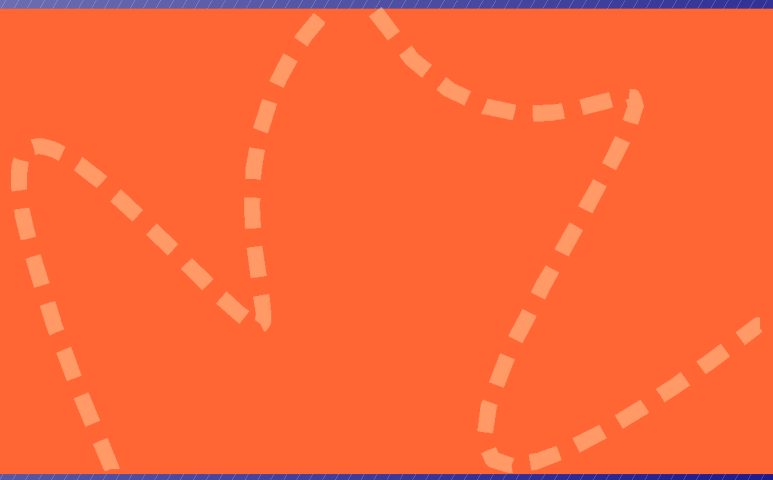
Programowanie obiektowe

Willy Picard
Katedra Technologii Informacyjnych
Akademia Ekonomiczna w Poznaniu
<*picard@kti.ae.poznan.pl*>

Agenda

- Cel(e) wykładu
- Od Włoch do Indonezji
- Interfejsy, Klasy i Obiekty
- Atrybuty i metody
- Kapsułkowanie
- Dziedziczenie i polimorfizm
- Podsumowanie

Cel(e) wykładu



Przegląd wykładu

Podstawowe pojęcia

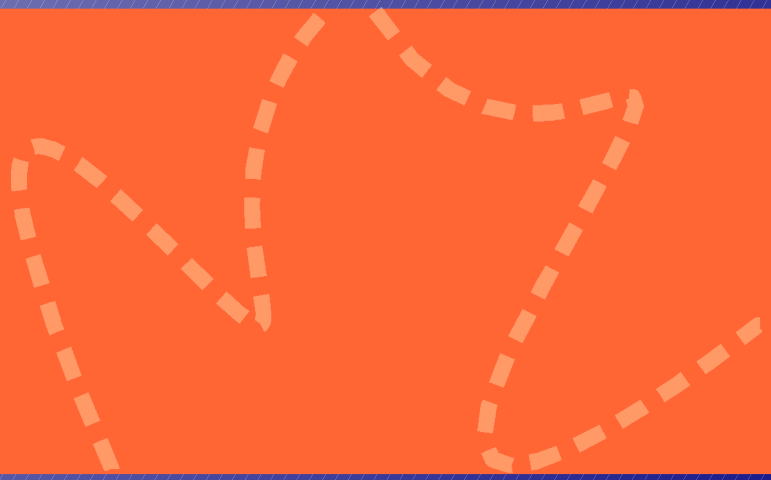
- 1: Wprowadzenie
- 2: Podstawowe struktury danych & instrukcje
- 3: Programowanie obiektowe I
- 4: Programowanie obiektowe II
- 5: Programowanie obiektowe III
- 6: Zaawansowane struktury danych
- 7: Wątki & Wyjątki

CELE WYKŁADU

Cel na dziś

Wprowadzić
programowanie obiektowe
(*object-oriented programming*)

Od Włoch do Indonezji



Włochy

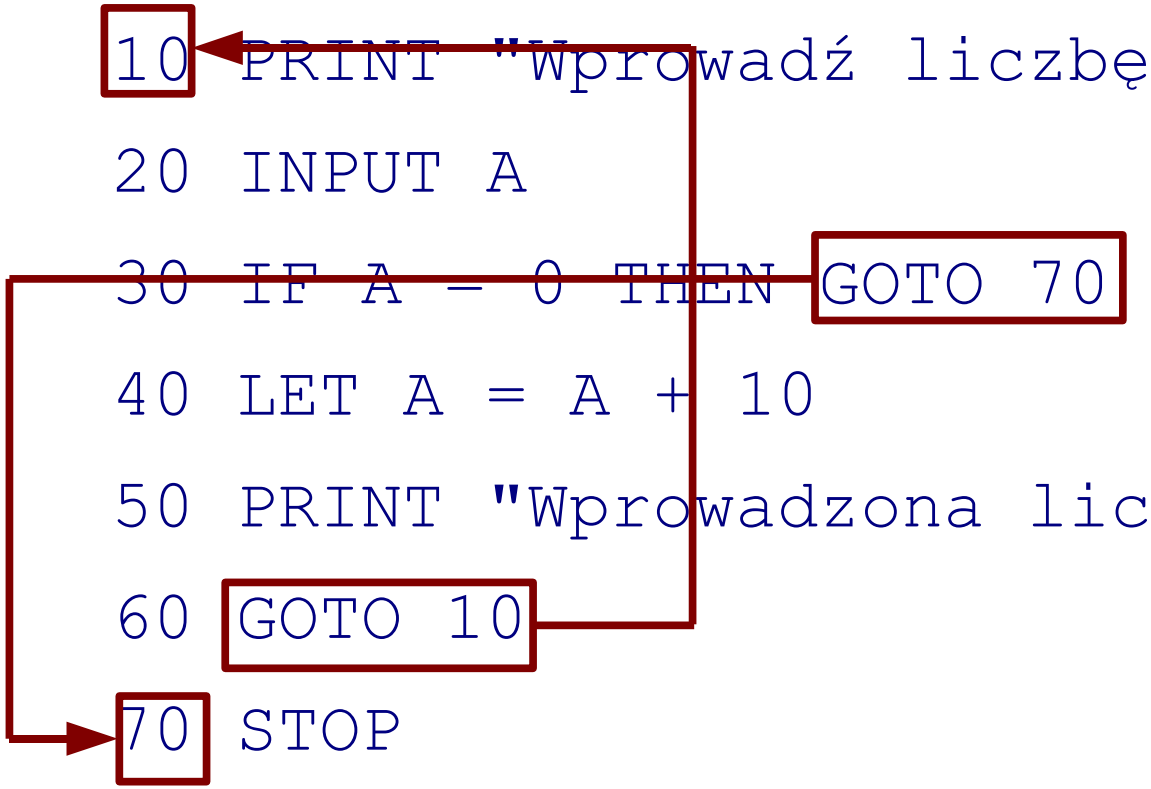
Programowanie Spaghetti

- Assembler, BASIC
- Instrukcje rozgałęzienia
- Dla miłośników `goto`
- Dane niestrukturyzowane

Łatwość utrzymania: *

Przykład w BASIC

```
10 PRINT "Wprowadź liczbę, zero żeby wyjść:";
20 INPUT A
30 IF A = 0 THEN GOTO 70
40 LET A = A + 10
50 PRINT "Wprowadzona liczba + 10 = "; A
60 GOTO 10
70 STOP
```



PASCAL

- Modularność
  - Procedury
  - Funkcje
- Dany ustrukturyzowane
- Funkcje i procedury operują na ustrukturyzowanych danych

Łatwość utrzymania: **

Przykład rekordu w PASCAL

```
program RECORD_INTRO (output);
      type  data = record
                     miesiąc, dzień, rok : integer
                   end;
      var   dziś : data;
      begin
            dziś.dzień    :=   25;
            dziś.miesiąc  :=   09;
            dziś.rok      := 1983;
            writeln('Dzisiaj: ',
                    dziś.dzień,':',
                    dziś.miesiąc,':',
                    dziś.rok);
      end.
```

Przykład procedury w PASCAL

```
program DODAJ LICZBY (input, output);
  procedure DODAJ ( pierwsza, druga : integer );
    var   wynik : integer;
    begin
      wynik := pierwsza + druga;
      writeln('Wynik = ', wynik);
    end;

    var   liczba1, liczba2 : integer;
    begin
      writeln('Wprowadź dwie liczby');
      readln( liczba1, liczba2 );
      DODAJ( liczba1, liczba2);
    end.
```

Przykład funkcji w PASCAL

```
program DODAJ_LICZBY (input, output);
  function DODAJ ( pierwsza, druga : integer ): integer;
    begin
      DODAJ := pierwsza + druga;
    end;

    var   suma, liczba1, liczba2 : integer;
    begin
      writeln('Wprowadź dwie liczby');
      readln( liczba1, liczba2 );
      suma := DODAJ( liczba1, liczba2)
      writeln('Suma = ', suma)
    end.
```

C

- Rozdzielenie
  - deklaracje
  - definicje
- Nagłówki
- Biblioteki

Łatwość utrzymania: ***

Przykład w C

W pliku „myMath.h”

```
int dodaj(int i, int j);
```

W pliku „myMath.c”

```
#include “myMath.h”
int dodaj(int i, inj) { return i+j };
```

W pliku myProg.c

```
#include "myMath.h"
#include <iostream.h>
#include <cstdlib>

int main( int argc, char* argv[]){
  int a = atoi(argv[1]);
  int b = atoi(argv[2]);
  int suma = dodaj(a, b);
  cout << a << "+" << b << "=" << suma;
}
```

Ograniczenia języka C

- Powiązanie między
  - Procedurami/funkcjami
  - Strukturami danych
- Kod rozproszony

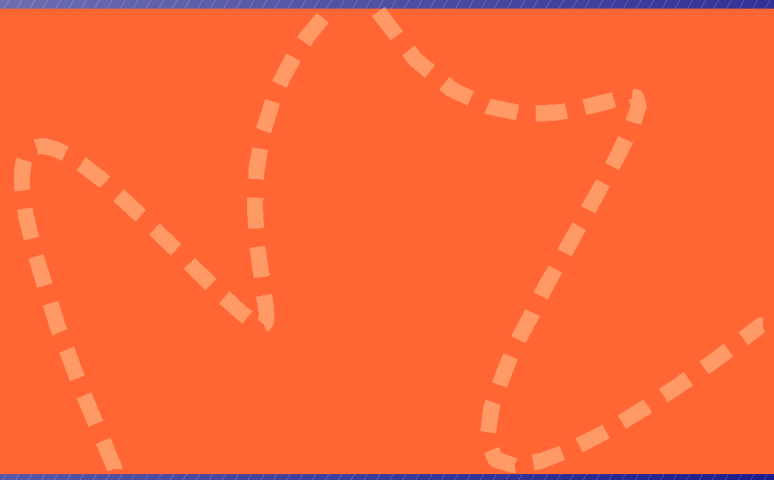
Obiektowe języki programowania *(OOPLs)*

- Historia
  - Nygaard i Dahl, Norwegian Computer Center
  - Simula 67
- Obecne obiektowe języki programowania
  - C++
  - Objective C
  - Smalltalk
  - Eiffel
  - Common LISP Object System (CLOS)
  - Object Pascal
  - Ada
  - ...

Indonezja


SOUTH CHINA SEA
BRUNAI
MALAYSIA
SUMATRA
KALIMANTAN
SULAWESI
MALUKU
PAPUA NEW GUINEA
IRIAN JAYA
JAVA
BALI
TIMOR
ARAFURA SEA
INDIAN OCEAN
TIMOR SEA

Interfejsy, Klasy i Obiekty



Definicja interfejsu

Interfejs jest definicją zachowania jako zbiór funkcji

Przykład interfejsu

- Interfejs *Samochód*
  - Wejdź do samochodu
  - Uruchom samochód
  - Przyspiesz
  - Zahamuj
  - Skręć
  - Zaparkuj samochód
  - Zatrzymaj samochód
  - Wyjdź z samochodu

Interfejsy w Javie

- Składnia

```
interface <nazwa>{
    ...
}
```

- Przykład

```
interface ISamochód {
    ...
}
```

Definicja klasy

Klasa jest definicją implementacji zachowania

Przykład klasy

- Klasa *Samochód*
  - np. Citroen C3
  - Wejdź do samochodu
  - Uruchom samochód
  - Przyspiesz
  - Zahamuj
  - Skręć
  - Zaparkuj samochód
  - Zatrzymaj samochód
  - Wyjdź z samochodu

Przykład implementacji samochodu

- Pedał gazu
- Kierownica
- Obecna prędkość

- Przyspiesz
  - Wciśnij pedał gazu

Klasy w Javie

- Składnia

```
class <nazwa>{
    ...
}
```

- Przykład

```
class CitroenC3 {
    ...
}
```

Klasyczna definicja klasy

Klasa jest zbiorem zmiennych i metod do ich obsługi

Definicja obiektu

Obiekt jest
instancją klasy

Przykład obiektu

- Obiekt typu *Samochód*
  - Konkretny samochód
  - np. Citroen C3 z rejestracją “PO TATO”
  - np. kierownica skórzana
  - np. sportowy pedał gazu
  - np. 30km/h

Klasy i interfejsy

Klasa, która implementuje (*implements*) interfejs musi zdefiniować wszystkie metody zadeklarowane w interfejsie

Klasy i interfejsy w Javie

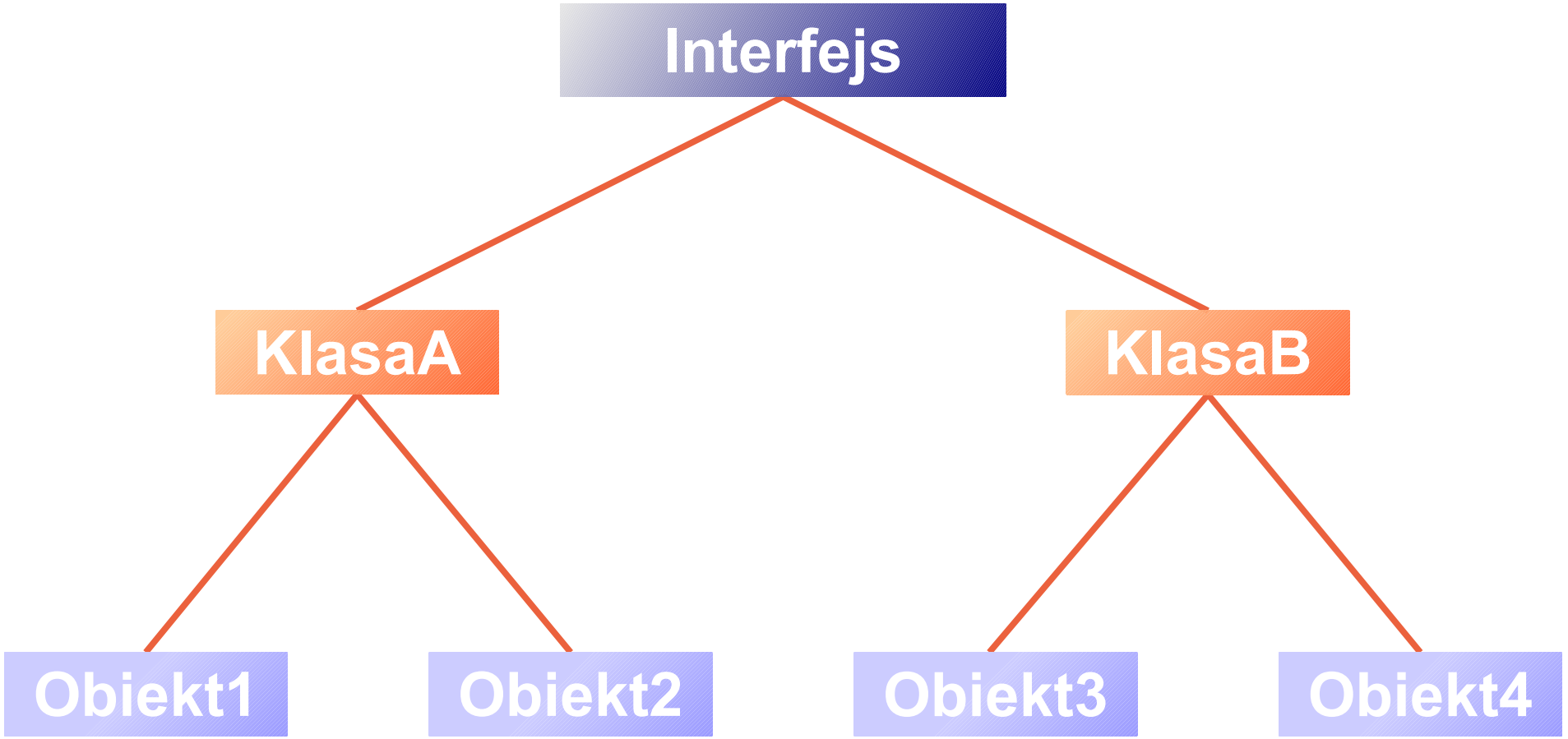
- Składnia

```
class <nazwaKlasy> implements <nazwaInterfejsu>{
    ...
}
```

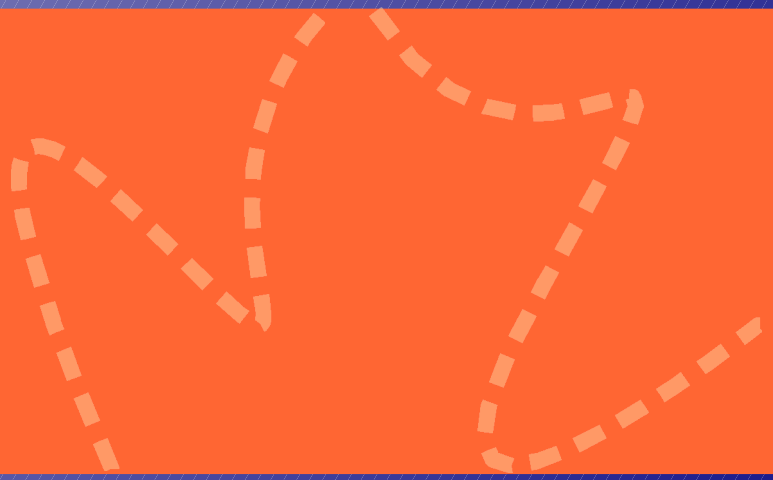
- Przykład

```
class CitroenC3 implements ISamochód{
    ...
}
```

Podsumowanie



Podsumowanie



Podsumowanie

- Reguła 1
  - Używaj interfejsy
- Reguła 2
  - Używaj interfejsy
- Reguła 3
  - Używaj interfejsy

Język C *vs.* OOPLs

- Powiązanie między — *klasy*
  - Procedurami/funkcjami
  - Strukturami danych
- Ponowne wykorzystanie kodu — *dziedziczenie*
- Kod rozproszony — *klasy, dziedziczenie*
- Deklaracja *vs.* definicja — *interfejsy, kapsułkowanie*

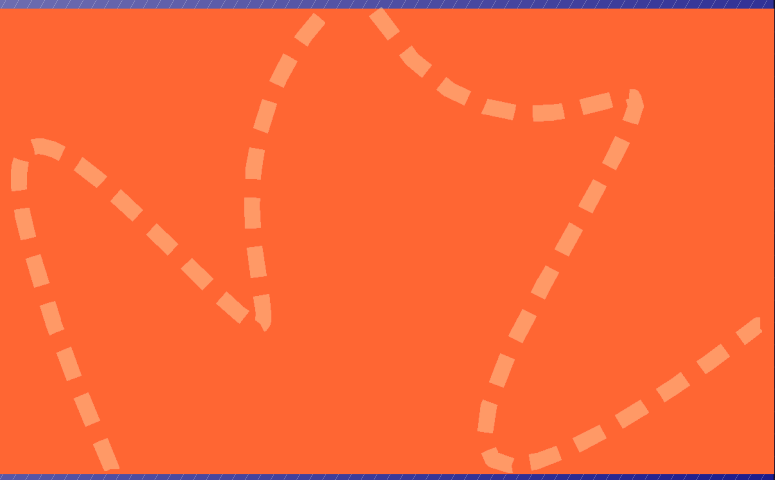
Przykład

```
package pl.poznan.ae.compProg;
import java.util.*;

public class Sorter {
  private List _words;

  public void sort(String[] words){
     _words = Arrays.asList(words);
     Collections.sort(_words);
  }
  public String getSortedWords(){
     String sortedString = "";
     for (int i = 0; i< _words.size(); i++){
       sortedString += _words.get(i);
     }
     return sortedString;
  }
 public static void main(String[] args){
     Sorter sorter = new Sorter();
     sorter.sort(args);
     System.out.println(sorter.getSortedWords());
 }
}
```

Do zobaczenia za tydzień



Programowanie obiektowe II

Powrót