Języki i paradygmaty programowania

Wykład 1

wstęp

Paradygmat

z gr.

παρα´δειγμα (*para´deigma*) = WZORZEC,

SPOSÓB WIDZENIA, PRZYKŁAD…

KOPALINSKI, Słownik wyrazów obcych i zwrotów ocojęzycznych: późn. łac. paradigma, dpn. paradigmatis 'przykład, wzór' z gr.

paradeigma 'j.p' od paradeiknynai 'zestawiać; porównywać';

Paradigm

paradigm |ˈparəˌdīm|

Noun

1 technical a typical example or pattern of something; a model: there

is a new paradigm for public art in this country.

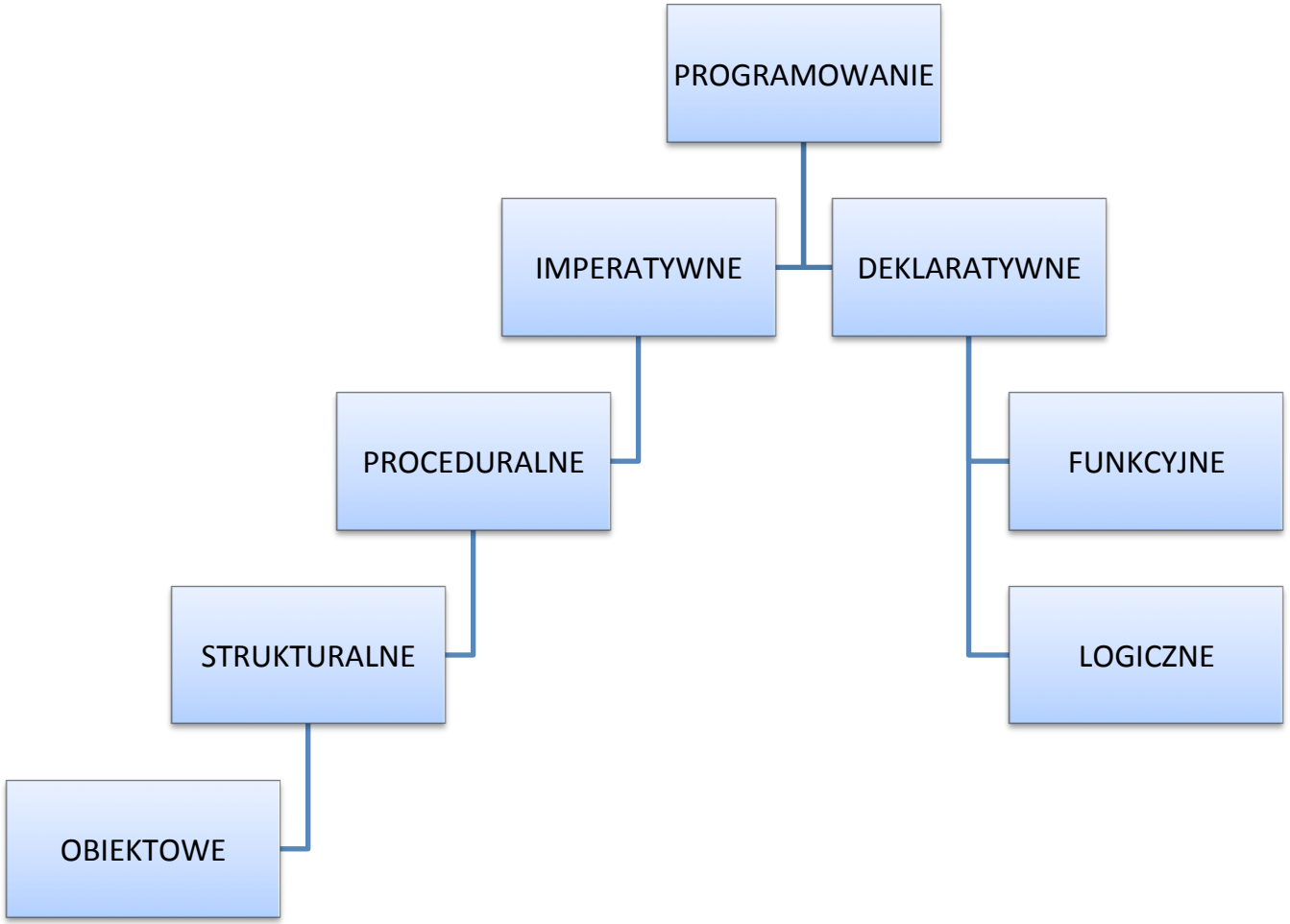
• a worldview underlying the theories and methodology of a particular scientific subject: the discovery of universal gravitation became the paradigm of successful science.

2 a set of linguistic items that form mutually exclusive choices in particular syntactic roles: English determiners form a paradigm: we can say “a book” or “his book” but not “a his book.” Often contrasted with syntagm.

• (in the traditional grammar of Latin, Greek, and other inflected languages) a table of all the inflected forms of a particular verb, noun, or adjective, serving as a model for other words of the same conjugation or declension.

ORIGIN late 15th cent.: via late Latin from Greek paradeigma, from paradeiknunai ‘show side by side,’ from para- ‘beside’ + deiknunai ‘to show.’

Paradygmaty programowania



Paradygmaty, a języki

- liczba paradygmatów: p
- liczba języków: $j = 2^p - 1$

$p \sim 10 \rightarrow j \sim 1000$

- Wszystkich nie omówimy … !
- Prawdę mówiąc, to $p > 40$ (plakat; Van Roy)

TIOBE

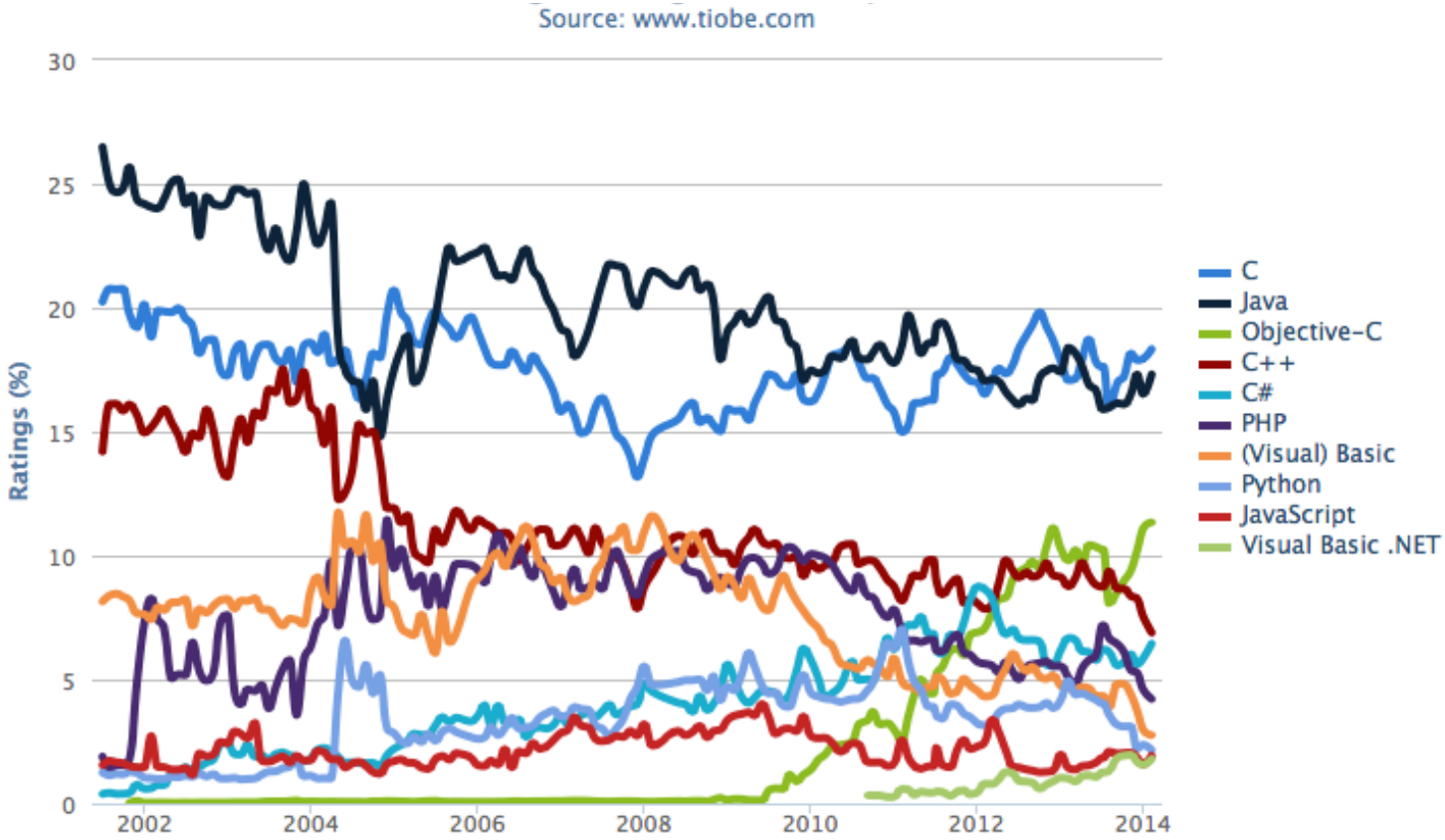
TIOBE Software BV has been founded 1st of October 2000 with the aid of a major investment of Swiss company Synspace and some private investors.

The name TIOBE stands for "The Importance Of Being Earnest". This is also the name of a comedy play written by Oscar Wilde at the end of the Nineteenth Century.

http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

Position Feb 2013	Position Feb 2012	Delta in Position	Programming Language	Ratings Feb 2013	Delta Feb 2012	Status
1	1	=	Java	18.387%	+1.34%	A
2	2	=	C	17.080%	+0.56%	A
3	5	↑↑	Objective-C	9.803%	+2.74%	A
4	4	=	C++	8.758%	+0.91%	A
5	3	↓↓	C#	6.680%	-1.97%	A
6	6	=	PHP	5.074%	-0.57%	A
7	8	↑	Python	4.949%	+1.80%	A
8	7	↓	(Visual) Basic	4.648%	+0.33%	A
9	9	=	Perl	2.252%	-0.68%	A
10	12	↑↑	Ruby	1.752%	+0.19%	A
11	10	↓	JavaScript	1.423%	-1.04%	A
12	16	↑↑↑↑	Visual Basic .NET	1.007%	+0.21%	A
13	13	=	Lisp	0.943%	+0.04%	A
14	15	↑	Pascal	0.932%	+0.12%	A
15	11	↓↓↓↓	Delphi/Object Pascal	0.886%	-1.08%	A
16	14	↓↓	Transact-SQL	0.773%	-0.07%	A--
17	75	↑↑↑↑↑↑↑↑↑↑	Bash	0.741%	+0.61%	A--
18	26	↑↑↑↑↑↑↑↑	MATLAB	0.648%	+0.15%	B
19	24	↑↑↑↑↑	Assembly	0.640%	+0.12%	B
20	19	↓	Ada	0.631%	0.00%	B

Popularność…?



TIOBE: Historia...

To see the bigger picture, please find the positions of the top 10 programming languages of many years back. Please note that these are *average* positions for a period of 12 months.

Programming Language	2014	2009	2004	1999	1994	1989
C	1	2	2	1	1	1
Java	2	1	1	13	-	-
Objective-C	3	38	48	-	-	-
C++	4	3	3	2	2	3
C#	5	8	9	30	-	-
PHP	6	5	6	-	-	-
(Visual) Basic	7	4	5	3	3	7
Python	8	6	11	27	22	-
JavaScript	9	9	8	20	-	-
Perl	10	7	4	5	17	23

TIOBE: Zwycięzcy 2013

The hall of fame listing all "Programming Language of the Year" award winners is shown below. The award is given to the programming language that has the highest rise in ratings in a year.

Year	Winner
2013	Transact-SQL
2012	Objective-C
2011	Objective-C
2010	Python
2009	Go
2008	C
2007	Python
2006	Ruby
2005	Java
2004	PHP

Program wykładu

- Wymagania wstępne
  - Podstawy logiki
  - Podstawy teorii mnogości
  - Podstawy algorytmiki
  - Język C, Pascal, C++, Java lub inny

cd

- A. Pojęcia wstępne:
  - składnia i semantyka języków programowania
  - analiza leksykalna i składniowa
  - nazwy
  - typy i konwersja typów
  - struktury kontrolne
  - przekazywanie parametrów do podprogramów
  - abstrakcyjne typy danych
  - przeciążanie operatorów i metod
  - polimorfizm

cd

- Programowanie **imperatywne**:
  - zmienne
  - wyrażenia
  - struktura blokowa
  - wiązanie statyczne i dynamiczne
  - podprogramy
  - organizacja wywołań podprogramów
  - przydział pamięci na stosie i na stercie
  - przykłady z języka C

cd

- Programowanie **obiektowe**:
  - abstrakcyjne typy danych
  - klasy
  - dziedziczenie
  - polimorfizm
  - dynamiczne wiązanie
  - przykłady z języków C++, Java

cd

- Programowanie **funkcyjne**:
  - funkcje jako model programowania
  - Churcha rachunek lambda
  - dopasowywanie wzorca
  - nadawanie typów
  - rekursja
  - leniwa ewaluacja
  - funkcje wyższego rzędu
  - przykłady z języków Lisp, Clojure, Haskell

cd

- Programowanie **w logice**:
  - rachunek predykatów
  - Listy
  - …
- Programowanie współbieżne
- Obliczenia kwantowe – elementy

Literatura 1

1. R. Sebesta: Concepts of Programming Languages: Addison-Wesley, 2011. **(podręcznik obowiązkowy**)
2. B. Tate: Siedem języków programowania, Helion, 2012. (B. Tate: SevenLanguages in seven weeks. Pragmatic Programmers., 2010;) (**podręcznik obowiązkowy**)
3. J. Bylina, B. Bylina: Przegląd języków i paradygmatów programowania.Skrypt. Lublin 2011.
4. B. Stroustrup: http://www.stroustrup.com + Książki: C++ (fragmenty podręcznika; **obowiązkowe**)
5. J. Gosling, H. McGilton: The Java Language Environment. A White Paper,October 1995, A Sun Microsystems

Literatura 2

1. K. Arnold, J. Gosling: The Java Programming Language. Addison Wesley,2005; Java, Wydawnictwa Naukowo -- Techniczne, Warszawa.
2. R. Bird: Introduction to Functional Programming using Haskell. Prentice Hall, 1988.
3. J. Reynolds: Theories of Programming Languages. Cambridge UniversityPress, 1998.
4. Stanford Engineering Everywhere, Introduction to Computer Science |
5. Programming Paradigms: http://see.stanford.edu/see/courses.aspx -->
6. Programming paradigms --> Lectures (by John Cain)

Literatura uzupełniająca

1. R. Penrose: Nowy umysł cesarza. PWN, Warszawa 1995 - Rozdzialy 1-4.
2. Matematyka współczesna, Dwanaście esejów. Pod red. L. A. Steana. WNT, Warszawa 1983, Rozdział: Czym jest obliczanie, wg. M(artina) Davisa, str. 261.
3. P. Van Roy, S. Haridi: Concepts, Techniques, and Models of Computer Programming. MIT Press, 2004.
4. K. Slonneger, B. L. Kurtz: Formal Syntax and Semantics of Programming Languages, Addison-Wesley, 1995. (Rozdział 5: Lambda calculus)

LABORATORIUM

- Języki (zależnie od prowadzącego i innych okoliczności)
  - ruby, python, perl
  - io
  - scala
  - prolog
  - lisp, clojure
  - erlang
  - haskell
  - …

LABORATORIUM

- Co 2-3 tygodnie zmienia się język programowania.
- Zadaniem studenta jest samodzielne poznanie podstaw nowego języka i sporządzenie właściwej notatki na ten temat. Notatka stanowi jedną z podstaw zaliczenia ćwiczeń
- Zadania programistyczne będą ogłaszane albo przez prowadzącego zajęcia w laboratorium albo w internecie
- W laboratorium Studenci realizują aktualne zadania programistyczne
- Kolokwia 2 lub 3
- Oceny otrzymane w trakcie realizacji ćwiczeń I na kolokwiach są podstawą zaliczenia laboratorium
- Obecność w laboratorium jest obowiązkowa i stanowi jeden z warunków zaliczenia ćwiczeń.

Język (L = language)

Definjcja

A = alfabet, zbiór znaków (np. ASCII)

L = skończony podzbiór zbioru A*

gdzie * oznacza dowolną ilość znaków alfabetu, w praktyce skończoną, wziętych w dowolnej kolejności, z możliwością powtórzeń.

Język

- Języki można definiować, używając gramatyk formalnych – reguł – zapisanych w metajęzyku, np. Extended BNF (Backus Naur Form) (ExtBNF, ISO/IEC 14977)
- Słowa języka posiadają jakieś ustalone znaczenie; ciągi słów – zdania – powinny posiadać jakiś sens (semantyka)
- Składnia języka podlega ustalonym regułom (gramatyce formalnej; syntaktyka)

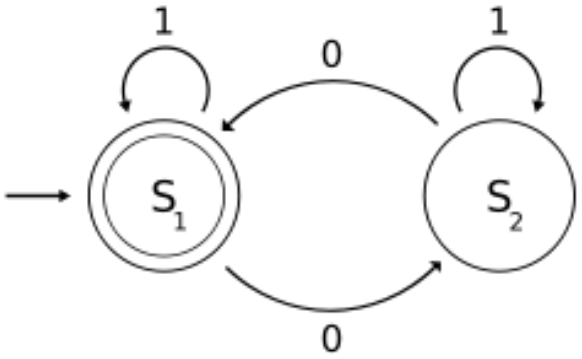
Przykład Ext. BNF

```
letter = "A" | "B" | "C" | "D" | "E" | "F" | "G"
       | "H" | "I" | "J" | "K" | "L" | "M" | "N"
       | "O" | "P" | "Q" | "R" | "S" | "T" | "U"
       | "V" | "W" | "X" | "Y" | "Z" ;
digit = "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9" ;
symbol = "[" | "]" | "{" | "}" | "(" | ")" | "<" | ">"
       | "'" | '"' | "=" | "|" | "." | "," | ";" ;
character = letter | digit | symbol | " " ;

identifier = letter , { letter | digit | " " } ;
terminal = "'" , character , { character } , "'"
         | '"' , character , { character } , '"' ;
```

```
lhs = identifier ;
rhs = identifier
    | terminal
    | "[" , rhs , "]"
    | "{" , rhs , "}"
    | "(" , rhs , ")"
    | rhs , "|" , rhs
    | rhs , "," , rhs ;

rule = lhs , "=" , rhs , ";" ;
grammar = { rule } ;
```

Przykłady generowania słów

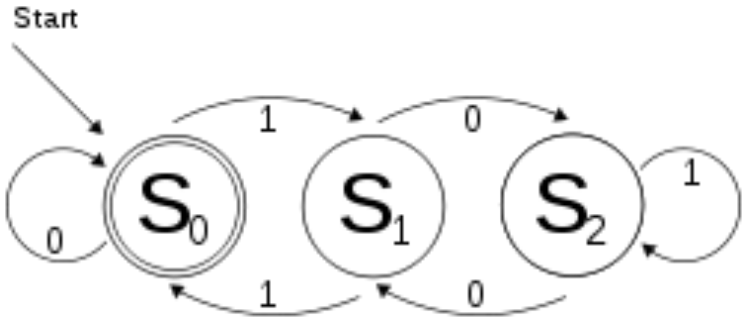
Parzysta liczba zer



- alfabet = (0,1)
- automat DFA:
- L = (100, 1001, 10110, …)

Przykłady generowania słów

Liczby podzielne przez 3:



(Stan końcowy: podwójne kółko)

(przykłady z Wikipedii)

Kryteria oceny języków*

Table 1.1 Language evaluation criteria and the characteristics that affect them

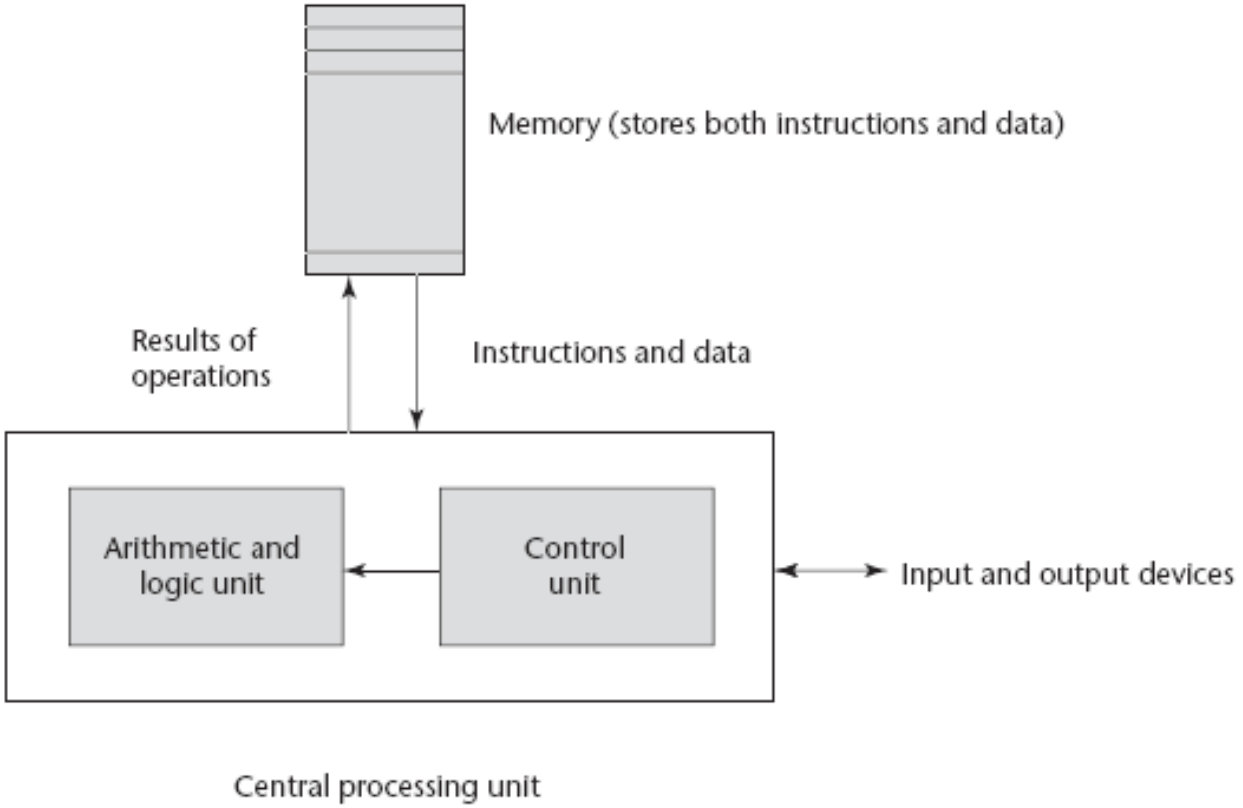
	CRITERIA		
Characteristic	READABILITY	WRITABILITY	RELIABILITY
Simplicity	•	•	•
Orthogonality	•	•	•
Data types	•	•	•
Syntax design	•	•	•
Support for abstraction		•	•
Expressivity		•	•
Type checking			•
Exception handling			•
Restricted aliasing			•

* Sebesta (dalej * oznacza to samo)

Komputer von Neumanna*

Figure 1.1

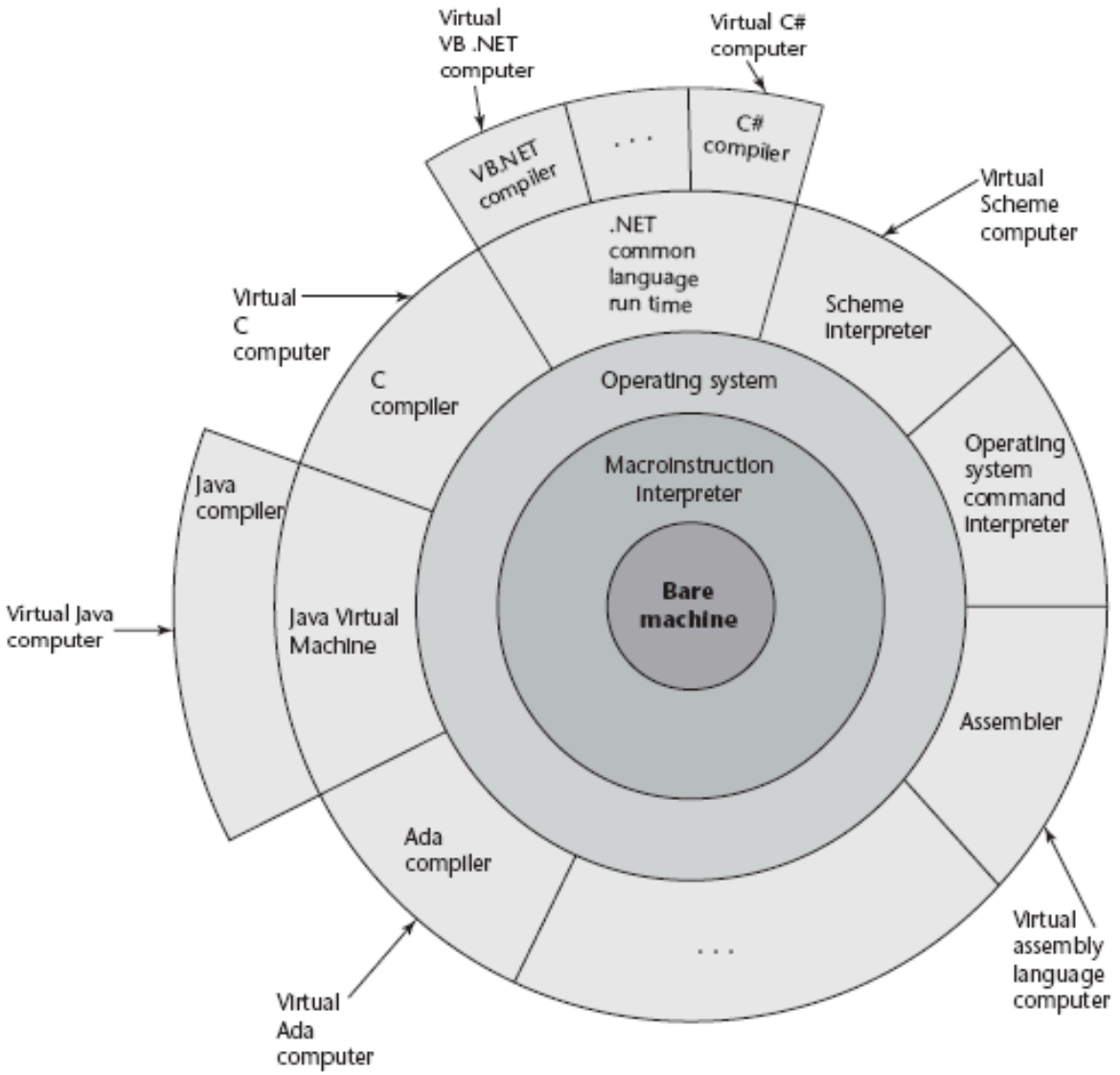
The von Neumann computer architecture



Komputer*

Figure 1.2

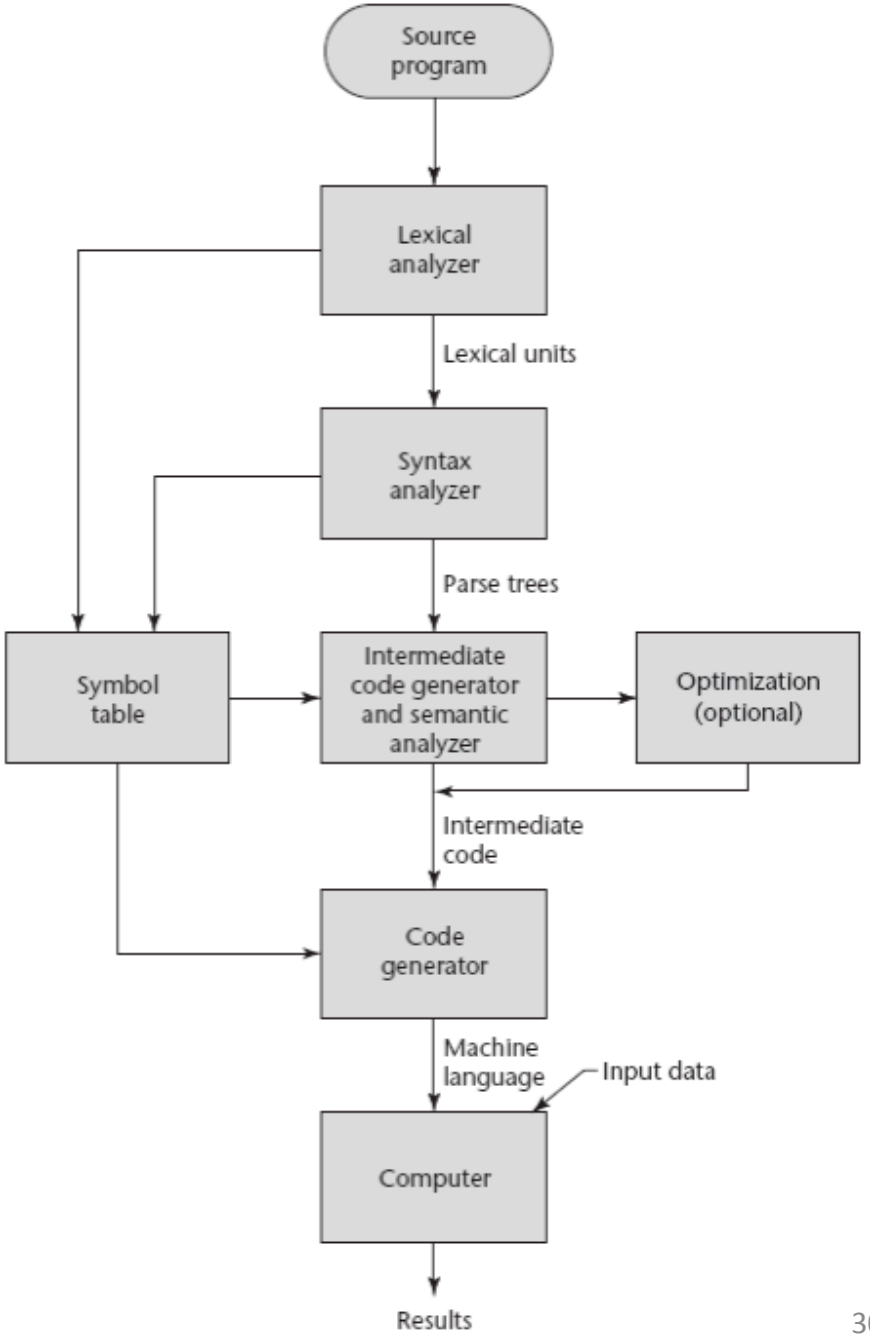
Layered interface of virtual computers, provided by a typical computer system



Kompilacja*

Figure 1.3

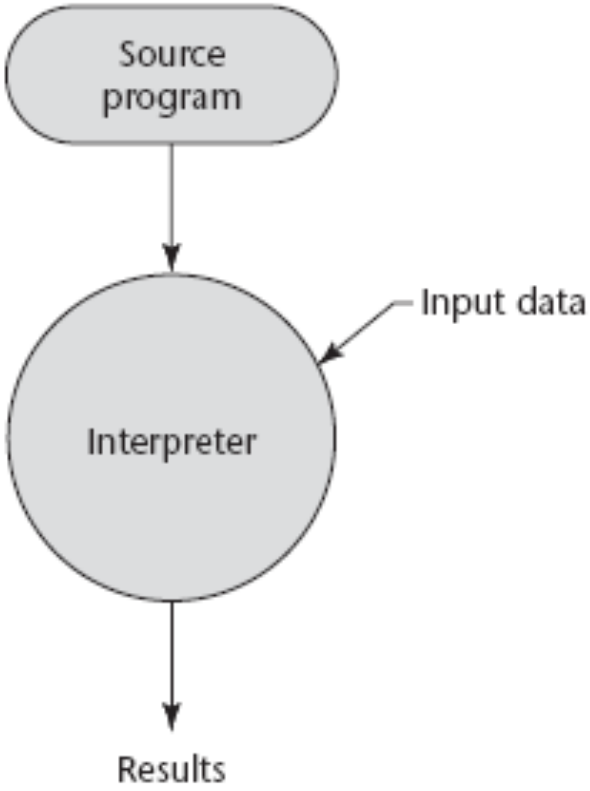
The compilation process



Interpretacja*

Figure 1.4

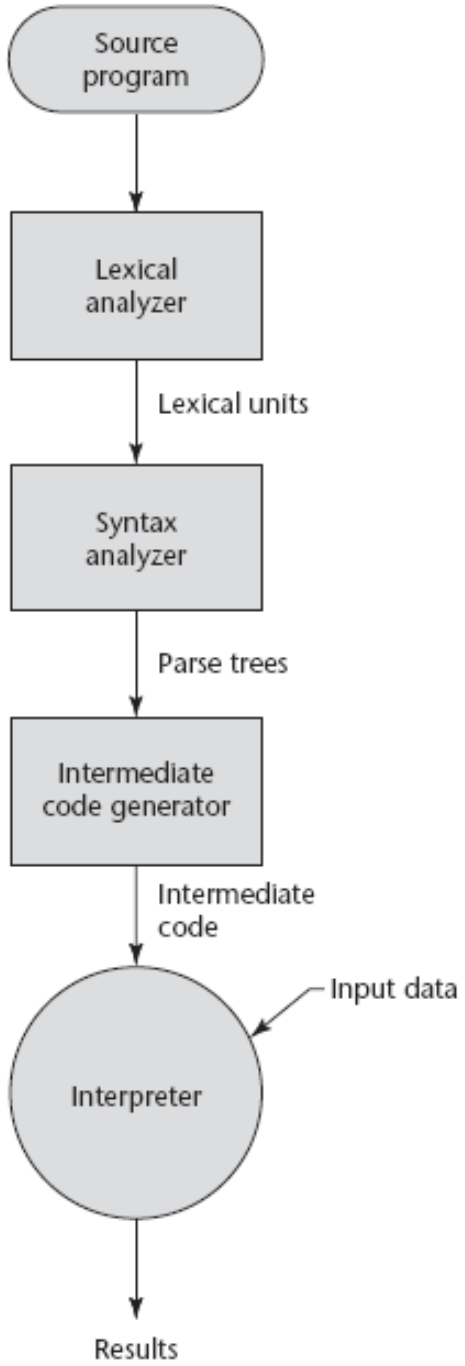
Pure interpretation



Realizacja hybrydowa*

Figure 1.5

Hybrid implementation system



Listy w Java

Napisz program listy.cpp.

Jak to jest w ruby?
Natychmiast!

(przykład z internutu)

```
import java.util.Iterator;
import java.util.List;
import java.util.ArrayList;

public class ListEx {

   public static void main(String[] args) {

      // List Example implement with ArrayList
      List<String> ls=new ArrayList<String>();

      ls.add("one");
      ls.add("Three");
      ls.add("two");
      ls.add("four");

      // remove list object
      ls.remove(2); // remove by index number
      ls.remove("four"); // remove by object

      Iterator it=ls.iterator();

      while(it.hasNext())
      {
        String value=(String)it.next();

        System.out.println("Value :"+value);
      }
   }
}
```