



Technical University of Lodz
Institute of Electronics

Algorytmy i struktury danych

*Wprowadzenie
dla studentów kierunku Inżynieria Biomedyczna*

Łódź 2017





Informacje ogólne

Wykładowcy



Andrzej Materka

materka@p.lodz.pl

<http://www.materka.p.lodz.pl>

tel.: 42 631 26 27

pokój 217, budynek B9

konsultacje:

<http://www.materka.p.lodz.pl/dydaktyka.html>



Marek Kociński

marek.kocinski@p.lodz.pl

<http://www.eletel.p.lodz.pl/kocinski/>

tel.: 42 631 26 17

pokój 218, budynek B9

konsultacje w kalendarzu:

https://poczta2011.p.lodz.pl/service/user/marek.kocinski@p.lodz.pl/studenci_2012.html



Informacje ogólne

- **Laboratorium:** 30 godz. (15 tyg. x 2 godz.)
- **Miejsce:** sala 413, budynek B9, Wólczańska 211/215
- **Punkty:** 3 ECTS (1 punkt ECTS = 25 - 30 godz.)
- **Praca własna:** 75 - 90 godzin
- **Język programowania:** Python
- **Narzędzia:** Środowisko Enthought Canopy



Najpopularniejsze języki (GitHub 2011)

GitHub is a [web-based hosting service](#) for software development projects that use the [Git revision control](#) system. GitHub offers both paid plans for private repositories, and free accounts for open source projects. As of May 2011, GitHub was the most popular open source code repository site.^[3]

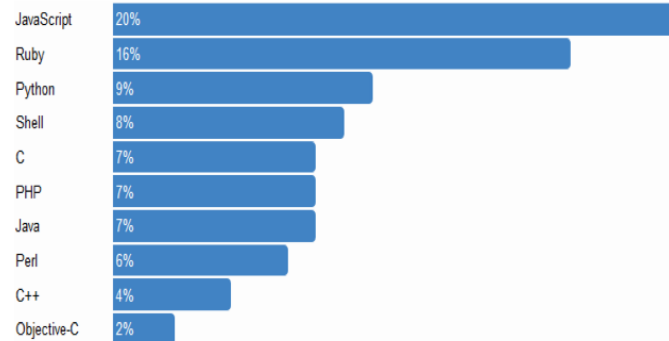
GitHub Inc. was founded in 2008 and is based in San Francisco, California.^[4]

In July 2012, the company received US\$100 Million in [Series A](#) funding, primarily from [Andreessen Horowitz](#).^{[5][6][7]}

Contents [\[hide\]](#)

- [1 Description](#)
- [2 Statistics](#)
- [3 Limitations and Constraints](#)
- [4 See also](#)
- [5 Notes](#)
- [6 References](#)
- [7 External links](#)

GitHub Top Languages 2011



GitHub

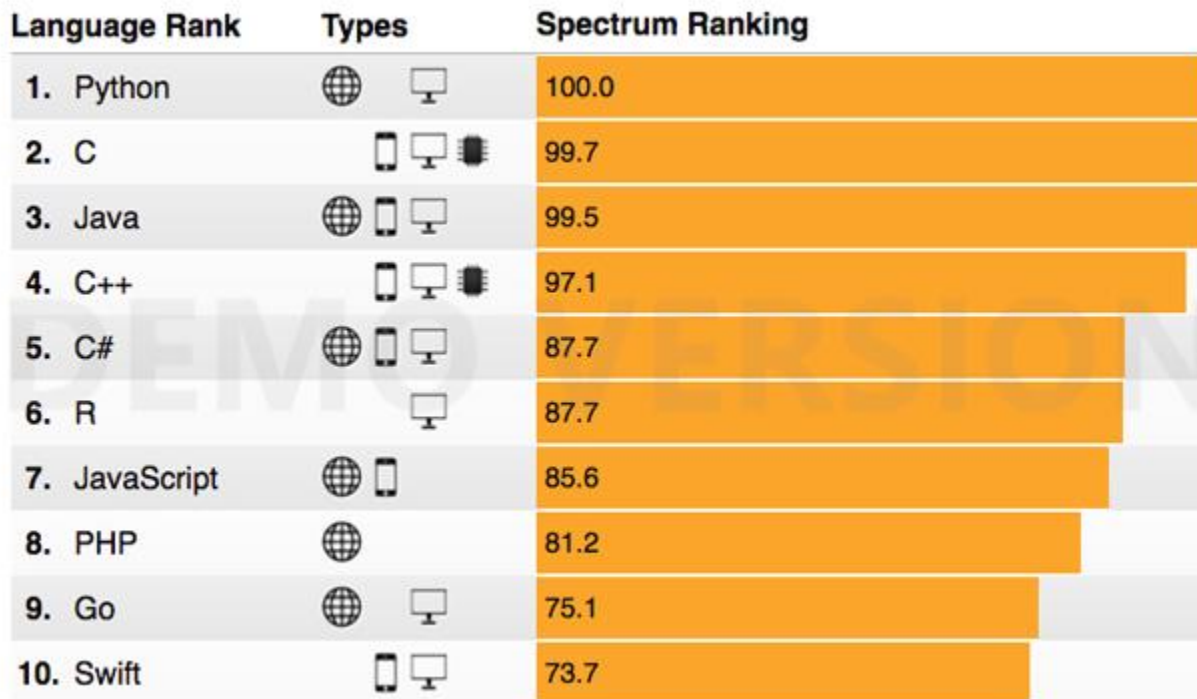
github
SOCIAL CODING

URL	GitHub.com [link]
Slogan	Social Coding (for all)
Commercial?	Yes
Type of site	collaborative revision control
Registration	Required
Available language(s)	English
Owner	GitHub, Inc. [link]
Launched	April 2008 ^[1]
Alexa rank	▼ 317 (September 2012) ^[2]



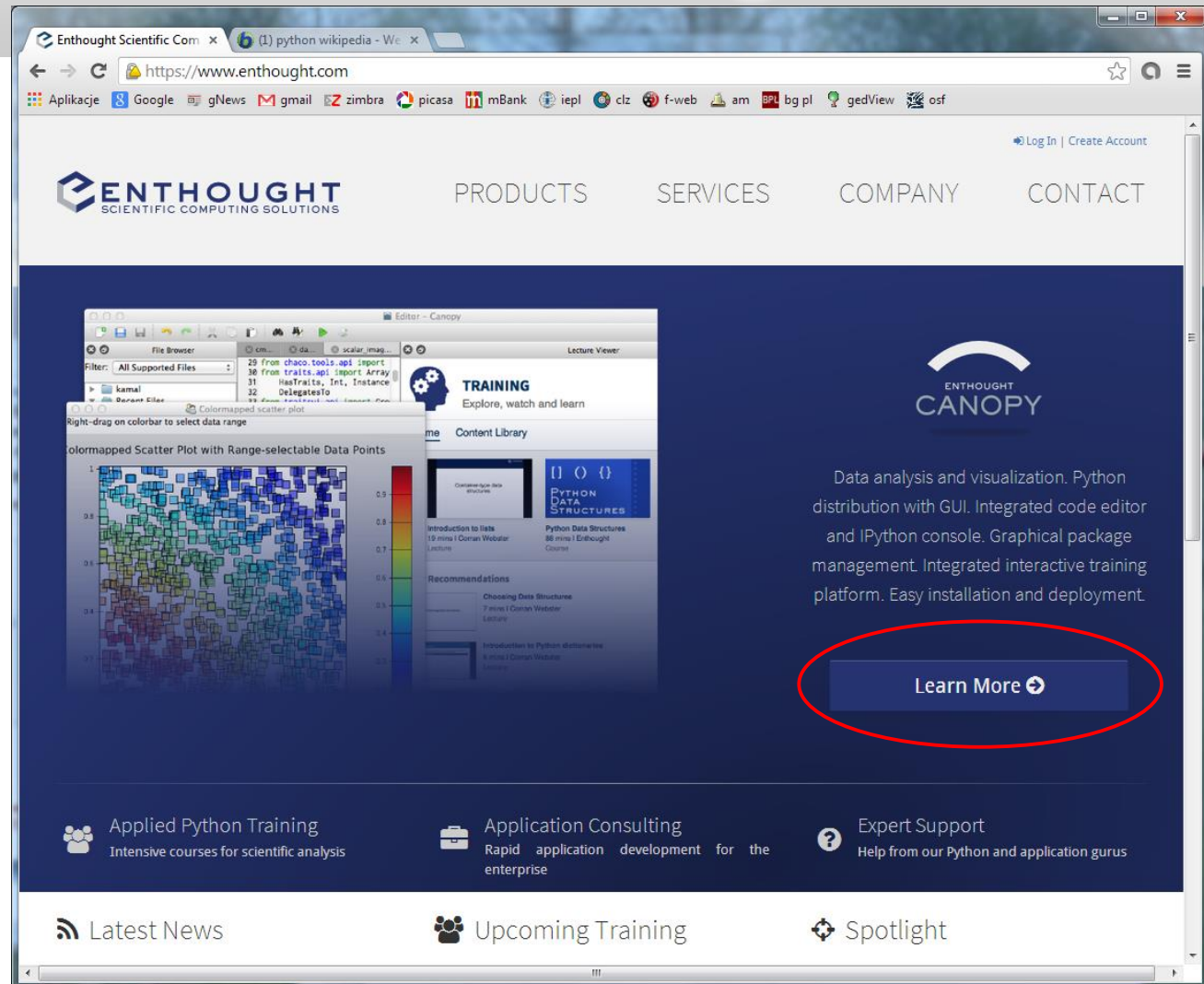
Najpopularniejsze języki (IEEE Spectrum 2017)

<https://spectrum.ieee.org/computing/software/the-2017-top-programming-languages>



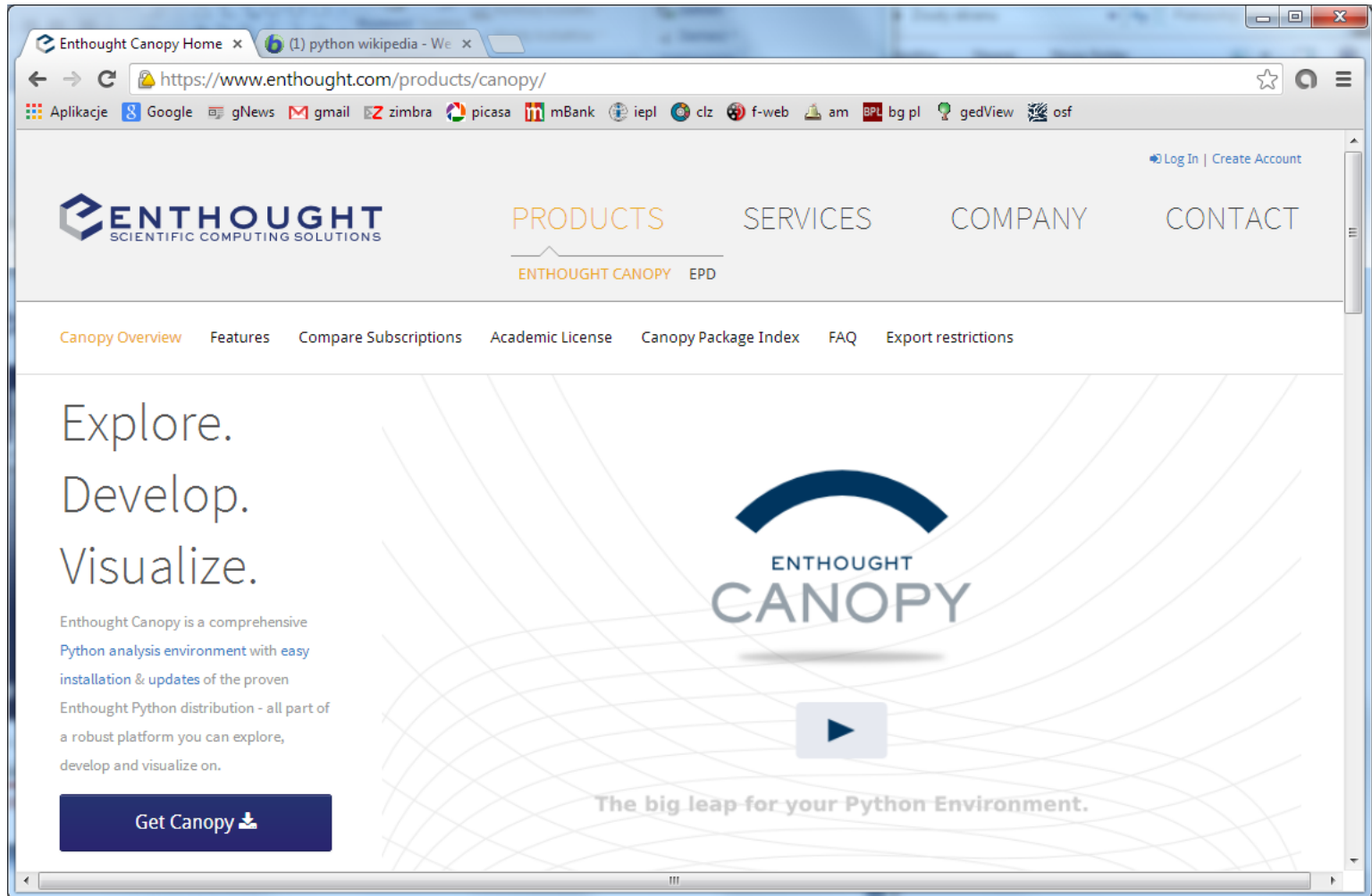


www.enthought.com





www.enthought.com/products/canopy/

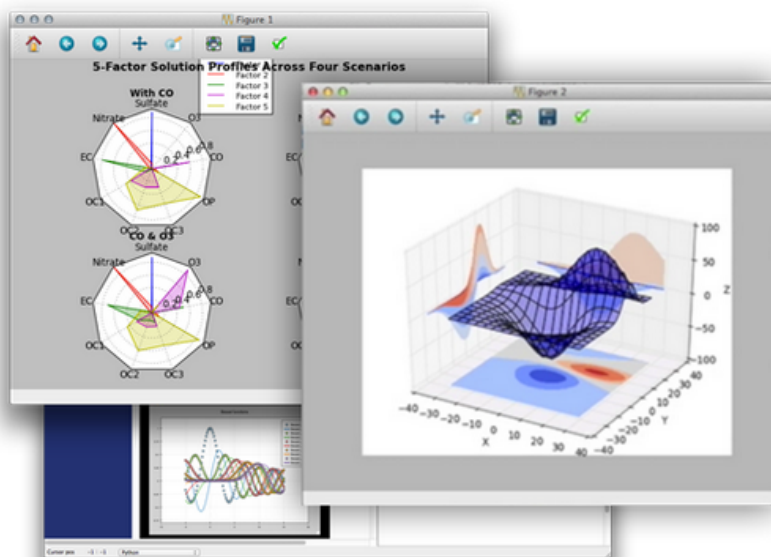




www.enthought.com/products/canopy/

[Canopy Overview](#) [Features](#) [Compare Subscriptions](#) [Academic License](#) [Canopy Package Index](#) [FAQ](#) [Export restrictions](#)

What's in it for you?



Scientists and Engineers

A comprehensive, Python-based analysis desktop & Python distribution, Canopy provides an open and intuitive environment for scientific and analytic computing. Since it's Python, your algorithms, scripts and programs will never be locked into a proprietary language. And with the analysis desktop, data analysis, scripting and plotting are more straightforward.

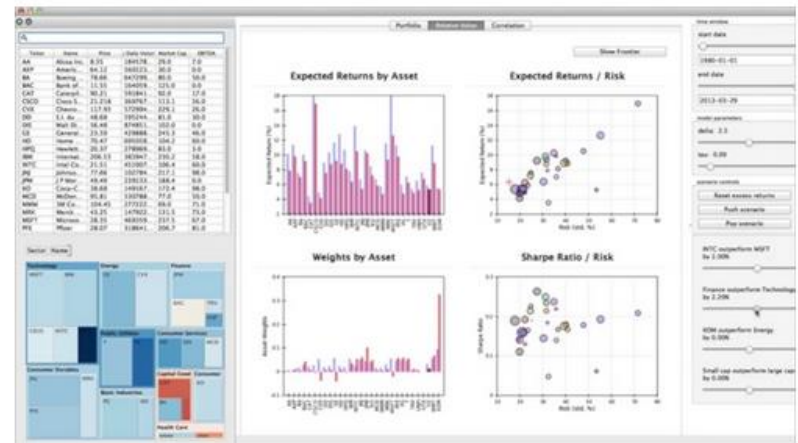


www.enthought.com/products/canopy/

Canopy Overview Features Compare Subscriptions Academic License Canopy Package Index FAQ Export restrictions

Quantitative and Data Analysts

Python spans from algorithm development and testing on the desktop to teams developing web-server applications. With the Canopy desktop and Python distribution, data ingestion, manipulation and analysis are simplified, and algorithm prototyping and testing are streamlined.





www.enthought.com/products/canopy/

[Canopy Overview](#) [Features](#) [Compare Subscriptions](#) [Academic License](#) [Canopy Package Index](#) [FAQ](#) [Export restrictions](#)



Portable Power

NORTHROP GRUMMAN



Enterprise

As a comprehensive Python-based analysis environment, Canopy puts powerful, yet very cost-effective, tools in the hands of analysts, scientists and engineers. As a robust application platform, it streamlines technical computing application development and deployment for your organization and for your customers.

With the popular, intuitive Python language and the comprehensive Canopy application platform, your organization can deploy new applications, algorithms and analysis tools much faster than with standard software languages and platforms. Users, especially "power" users, can extend and innovate with scripting and open platform APIs, driving the creation and sharing of innovative techniques and tools.



Kto używa języka Python?

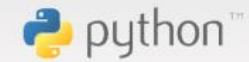


Jan Koprowski <jan.koprowski@gmail.com> Politechnika Gdańska, FTIMS – Informatyka Stosowana

5



Kto używa języka Python?



NOKIA
CONNECTING PEOPLE



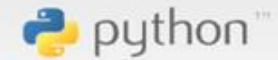


Kto używa języka Python?

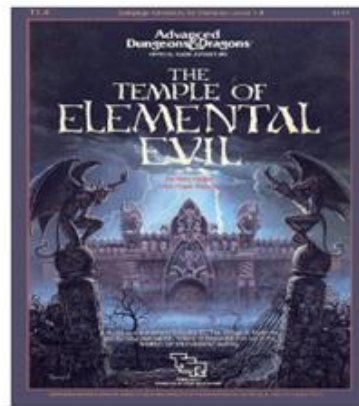
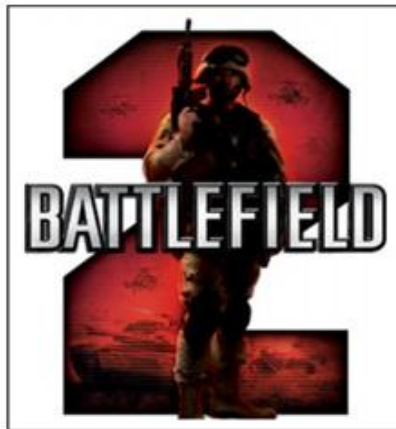




Kto używa języka Python?



SID MEIER'S CIVILIZATION IV





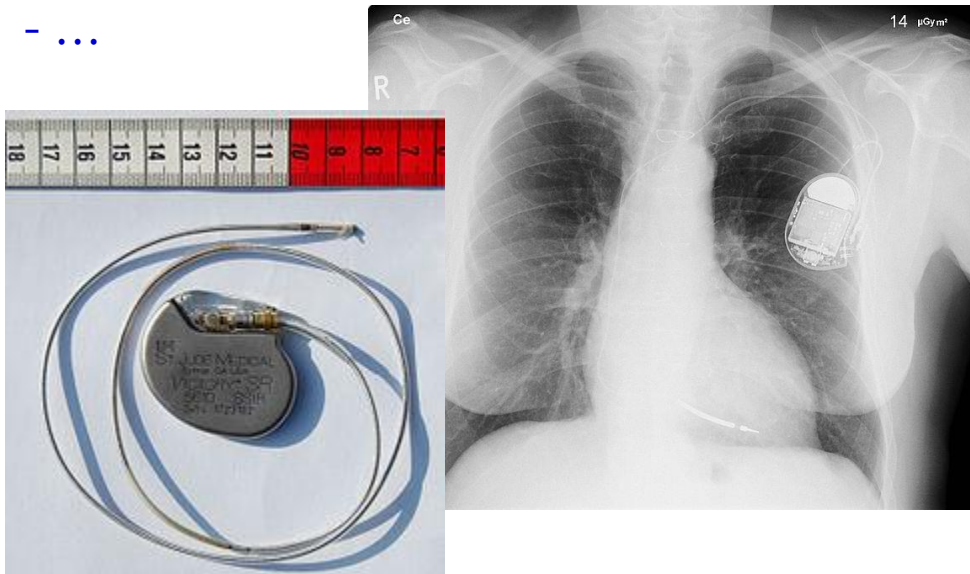
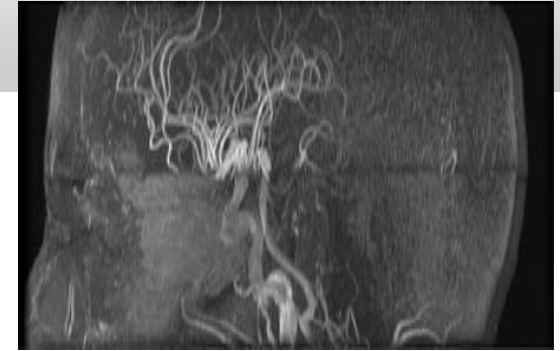
Python i inne języki





Komputery w medycynie

- Systemy informatyczne w ochronie zdrowia
- Obrazowanie medyczne (CT, MRI,...)
- Ilościowa analiza obrazów
- Analiza sygnałów medycznych
- Systemy nawigacji chirurgicznej
- Stymulatory serca
- ...



http://en.wikipedia.org/wiki/Artificial_cardiac_pacemaker



Komputery w medycynie



Laptops, Netbooks and Tablets



Desktops and Workstations



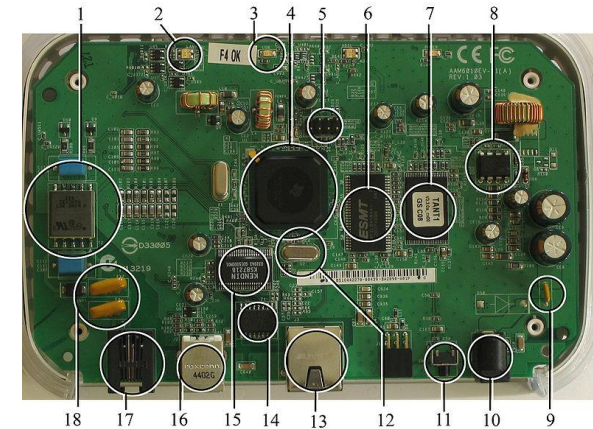
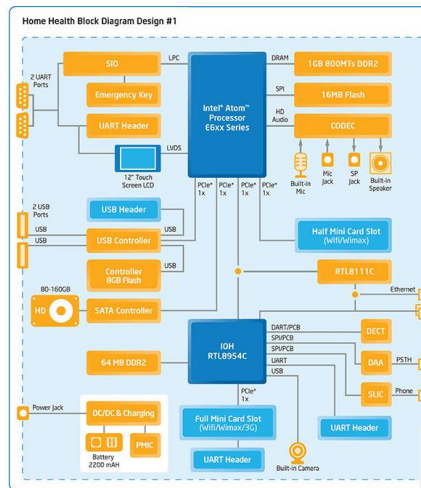
Servers, Storage and Networking



Software and Peripherals

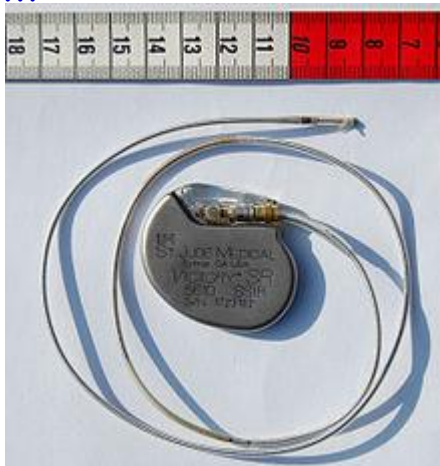
- Systemy informatyczne w ochronie zdrowia
- Obrazowanie medyczne (CT, MRI,...)
- Ilościowa analiza obrazów
- Analiza sygnałów medycznych
- Systemy nawigacji chirurgicznej
- Stymulatory serca

<http://content.dell.com/us/en/healthcare/healthcare-solutions>



Embedded systems

- Czym jest komputer w tych zastosowaniach?
- Jakie funkcje pełni?
- Dlaczego używamy komputerów?



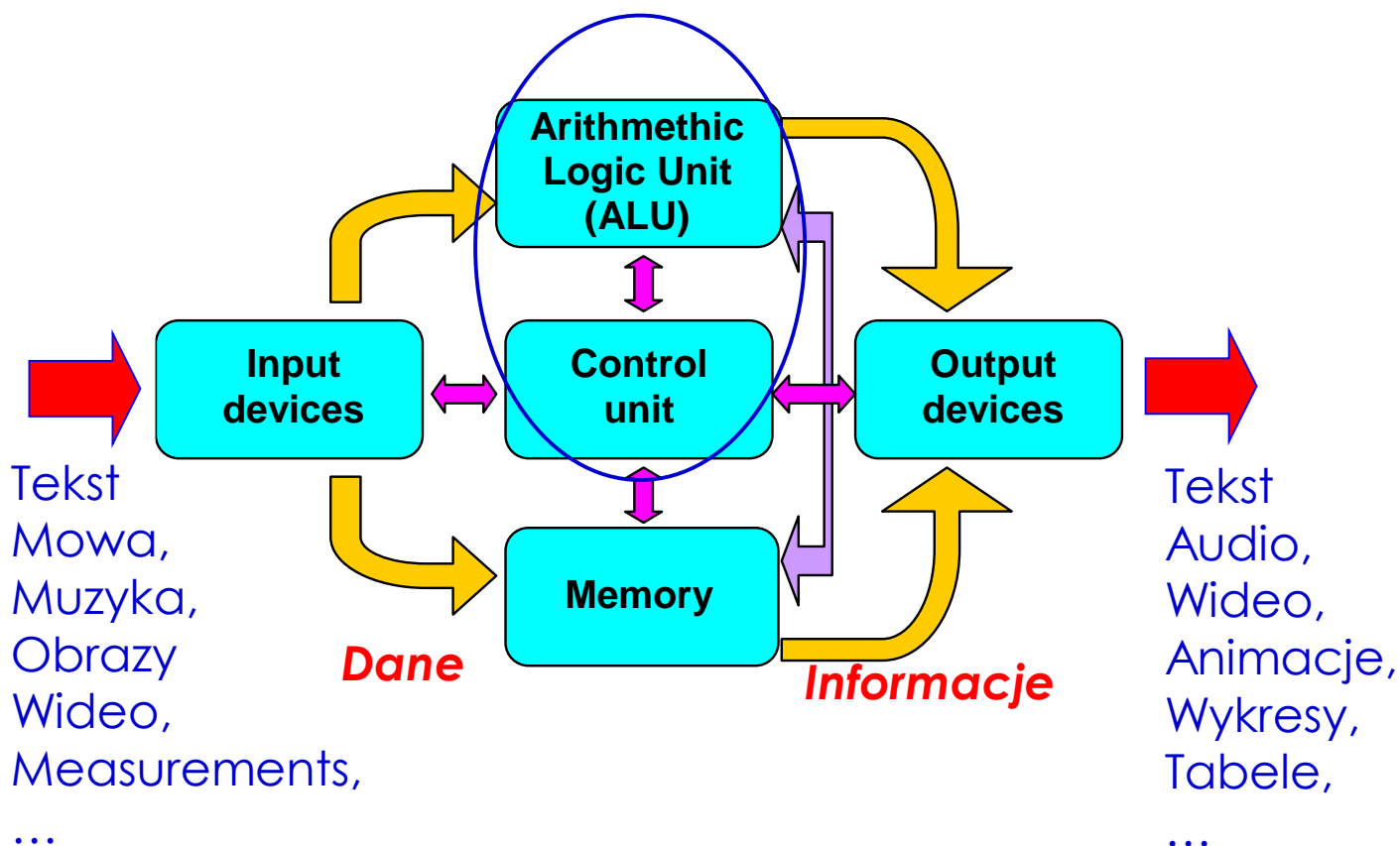


The diagram illustrates the internal components of a computer and their interactions. At the top left, a box labeled 'Zegar' (Clock) points to a sequence of bits '0 1 0 1 0 1 0 1' with a 'time' axis below it. A callout box explains: 'Krótszy okres zegara (większa częstotliwość) → szybszy komputer' (Shorter clock period (higher frequency) → faster computer). The main components are: 'Jednostka arytmetyczno-logiczna (ALU)' (Arithmetic Logic Unit) at the top; 'Układy wejściowe' (Input Units) on the left; 'Układy sterujące' (Control Units) in the center; 'Układy wyjściowe' (Output Units) on the right; and 'Pamięć' (Memory) at the bottom. A 'Program' box at the bottom left points to the Memory. Data flow is shown with yellow arrows: from Input to ALU and Memory; from Memory to Output; and from ALU to Output. Control flow is shown with purple arrows: bidirectional connections between ALU and Control Units, and between Control Units and Memory. Red arrows indicate the overall input and output of the system.



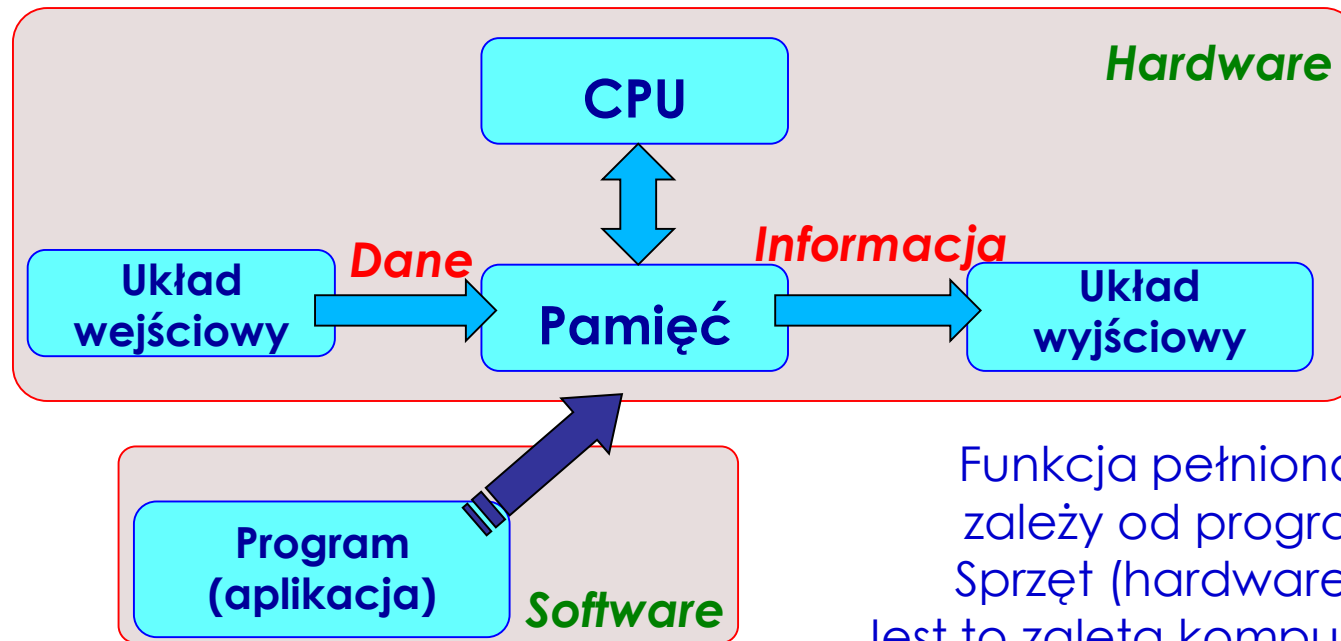
Podstawy komputerów

(Arithmetic Logic Unit + Control Unit) = Central Processing Unit (CPU)





Potrzeba programowania

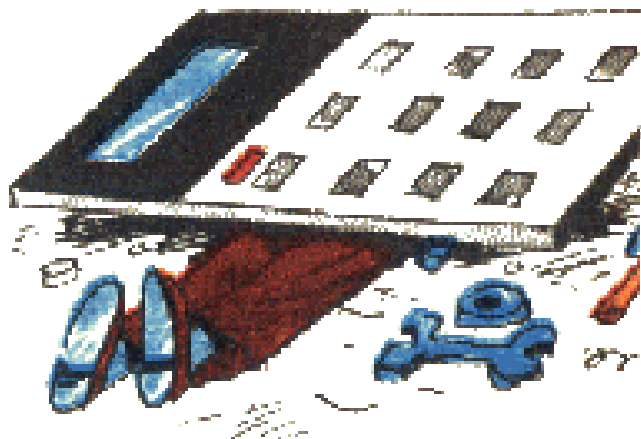


Funkcja pełniona przez komputer zależy od programu (software'u). Sprzęt (hardware) się nie zmienia. Jest to zaleta komputerów cyfrowych (elastyczność funkcjonalna).



Podstawowe założenia

- Dane są zapisane w postaci liczb binarnych.
- Czas wykonania elementarnych instrukcji jest bardzo krótki.
- Elementy składowe układów komputera mają bardzo małe rozmiary.



Reprezentacja liczb

dziesiętna	dwójkowa
10 1	8 4 2 1
0	0
1	1
2	1 0
3	1 1
4	1 0 0
...	...
1 5	1 1 1 1

- dekompozycja na proste operacje
- odporność na zakłócenia



Podstawowe operacje na liczbach dwójkowych

Reguły dodawania

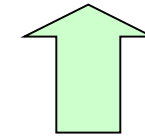
0	0	1	1
+0	+1	+0	+1
<hr/>	<hr/>	<hr/>	<hr/>
0	1	1	10

6×5:

			1	1	0
		×	1	0	1
			<hr/>		
			1	1	0
		0	0	0	
+	1	1	0		
	<hr/>				
	1	1	1	1	0

Kodowanie tekstu

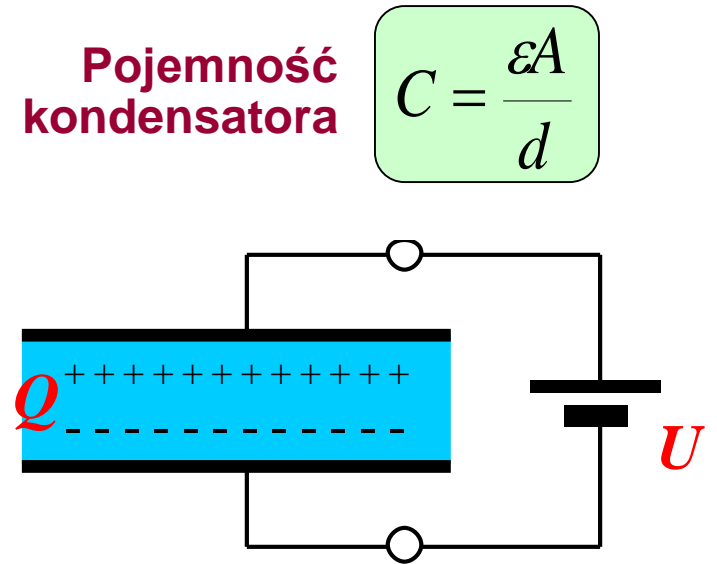
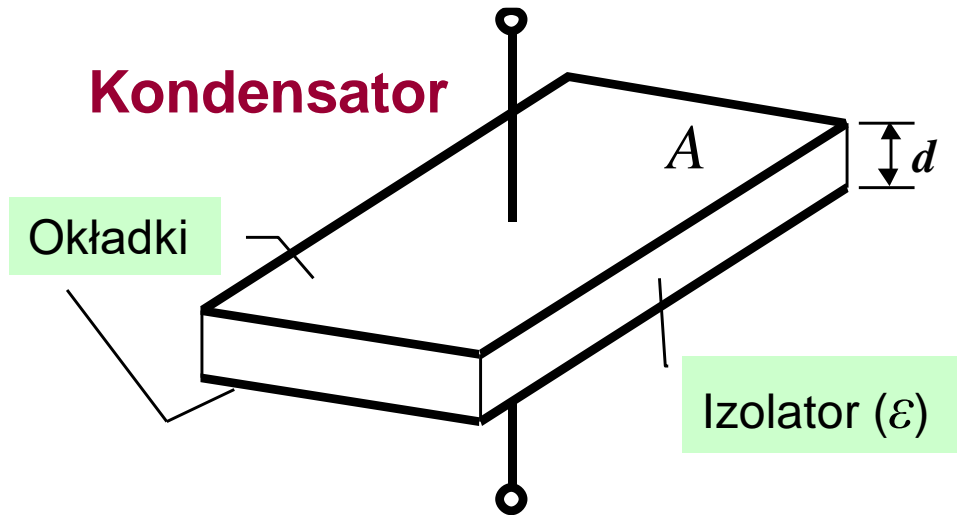
!	-	00100001
\$	-	00100100
A	-	01000001
B	-	01000010
Z	-	01011010
a	-	01100001
m	-	01101110



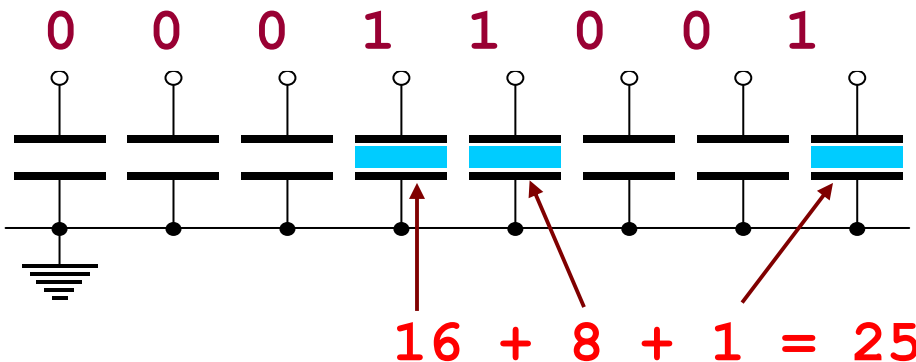
Możliwość operacji na symbolach



Pamięć komputera



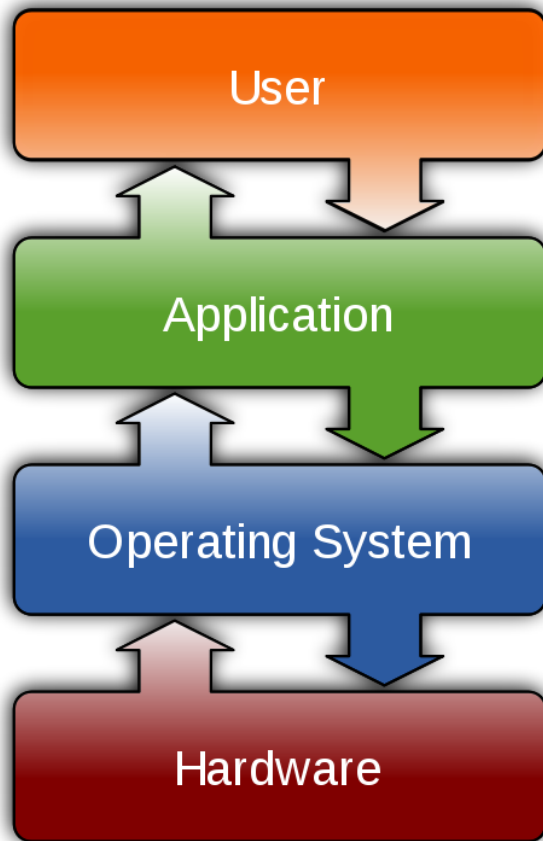
1 bajt pamięci (8 bitów-)



$$Q = CU$$

Ładunek Pojemność Napięcie

System operacyjny (Operating System)



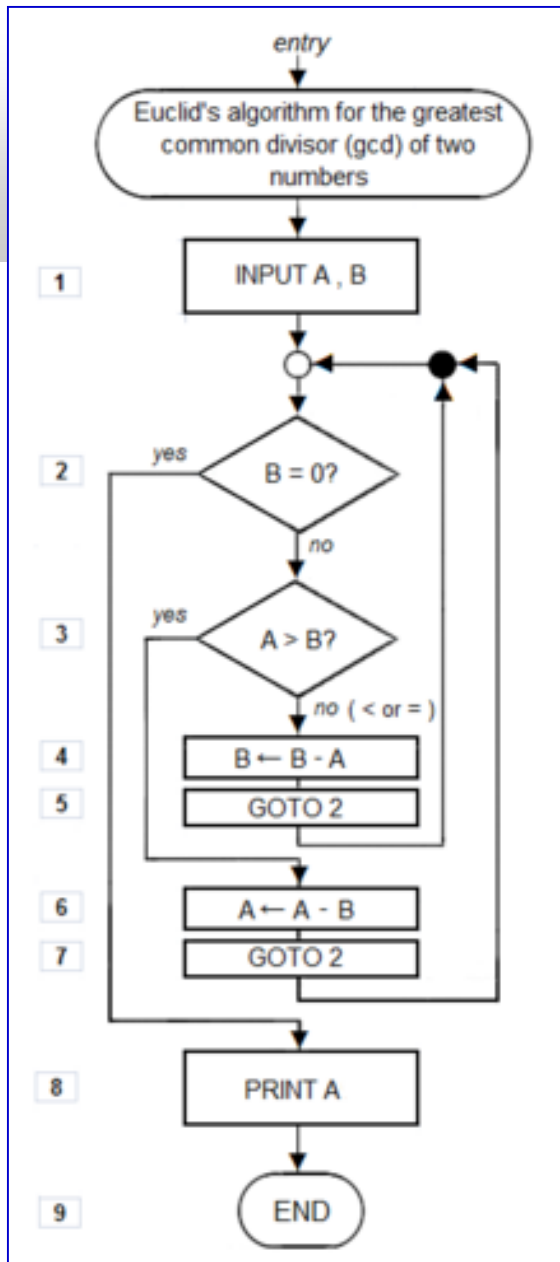
Zestaw programów do zarządzania sprzętem (hardware'm) i do zapewniania usług (services) dla programów aplikacyjnych.

Przykłady: Microsoft Windows, Apple Mac OSX, Android, Linux.



Algorytm

Procedura obliczeń, krok po kroku



Muḥammad ibn Mūsā al-Khwārizmī (Persian)



Języki programowania

Generacje języków

1. Kod maszynowy (100011 00011 01000...)

2. Asembler



3. Zbliżone do języków ludzkich,
kompilowane lub asemblowane przed
wykonaniem (np. Java, C, Pascal)

4. Języki dziedzinowe (domain-specific)
(np. COBOL)

5. Rozwiązywanie problemów przez użycie ograniczeń i warunków
zamiast algorytmów napisanych przez programistę
(sztuczna inteligencja, uczenia na przykładach).

<u>Adres</u>	<u>Mnemonic instrukcji</u>	<u>Argumenty</u>
00000000	push	ebp
00000001	mov	ebp, esp
00000003	movzx	ecx, [ebp+arg_0]
00000007	pop	ebp
00000008	movzx	dx, cl
0000000C	lea	eax, [edx+edx]
0000000F	add	eax, edx
00000011	shl	eax, 2
00000014	add	eax, edx
00000016	shr	eax, 8
00000019	sub	cl, al
0000001B	shr	cl, 1
0000001D	add	al, cl
0000001F	shr	al, 5
00000022	movzx	eax, al
00000025	ret	



Przykład: Python a assembler

Program do wyświetlania komunikatu „Hello World”

Asembler X86 – x86-64 Linux, syntaks AT&T

```
.section          .rodata
string:
    .ascii "Hello, World!\n\0"

length:
    .quad . -string          #Dot = 'here'

.section          .text
.globl _start          #Make entry point visible to linker
_start:
    movq $4, %rax          #4=write
    movq $1, %rbx          #1=stdout
    movq $string, %rcx
    movq length, %rdx
    int $0x80              #Call Operating System
    movq %rax, %rbx        #Make program return syscall exit status
    movq $1, %rax          #1=exit
    int $0x80              #Call System Again
```



Przykład: Python a assembler

Program do wyświetlania komunikatu „Hello World”

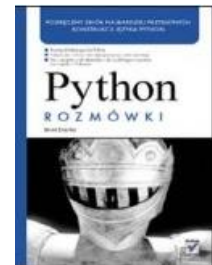
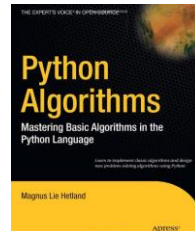
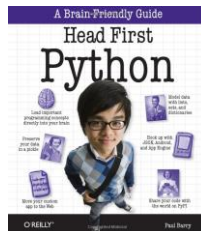
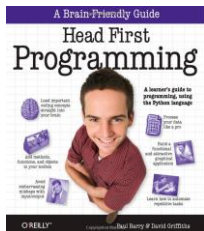
Kod w języku Python

```
print "Hello World"
```



Literatura

- **Head First Programming: A Learner's Guide to Programming Using the Python Language** by David Griffiths
- **Head First Python** by Paul Barry
- **Python Algorithms: Mastering Basic Algorithms in the Python Language** by Magnus Lie hetland
- **Python. Rozmówki**, Brad Dyley
- **Python od Podstaw** – zespół autorów
- **Zanurkuj w pythonie** (http://pl.wikibooks.org/wiki/Zanurkuj_w_Pythonie)
- **Programowanie z Pythonem** podręcznik stworzony dla studentów I roku neuroinformatyki I fizyki medycznej na Wydziale Fizyki Uniwersytetu Warszawskiego (http://brain.fuw.edu.pl/edu/TI:Programowanie_z_Pythonem/Wersja_do_druku)
- I wiele innych książek oraz materiałów szkoleniowych dostępnych w Internecie





Ocena końcowa

- 5 testów w czasie semestru (75 %)
- Aktywność na zajęciach (obecność, prezentacje, programowanie) (25 %)

Uczestnicy zajęć są zobowiązani do przynoszenia ich elektronicznych legitymacji studenckich.



Dobre rady

- Slajdy prezentowane na zajęciach nie obejmują całej wiedzy potrzebnej do zaliczenia testów.
- Wiedzę i umiejętności zdobywa się drogą studiów i ćwiczeń.
- Pomocne jest zapisywanie notatek w czasie zajęć.
- Warto przeglądać materiały szkoleniowe (ang. tutorials) dotyczące modułów języka Python używanych na zajęciach.



Wykorzystane materiały

1. Python for Scientist and Engineers – slajdy szkoleniowe Enthought, Inc. www.enthought.com
2. www.pl.python.org
 1. <http://pl.python.org/docs/>
 2. <http://pl.python.org/kursy,jezyka.html>
 3. <http://pl.python.org/wyklady.html>