Wstęp do informatyki

Informatics or computer science?
History

Piotr Fulmański

Wydział Matematyki i Informatyki,
Uniwersytet Łódzki, Polska

7 października 2010

Spis treści

1. **Informatics or computer science?**
   - Computer science
   - Informatics

2. **History**
   - Prehistory
   - Starożytność
   - Pierwsze tysiąclecie naszej ery
   - Wiek XX

Informatics or computer science?

Computer Science is no more about computers than astronomy is about telescopes

Edsger W. Dijkstra

The computer was born to solve problems that did not exist before

Bill Gates

Informatics or computer science?

Computer Science is no more about computers than astronomy is about telescopes

Edsger W. Dijkstra

The computer was born to solve problems that did not exist before

Bill Gates

Informatics or computer science?

Informatics or computer science?

Odpowiedzi na postawione pytanie nie ułatwia też wcale obowiązująca terminologia. W języku polskim mamy termin *informatyka*, który najczęściej na język angielski tłumaczony jest jako „Computer science", co sugeruje bardzo bliskie powiązanie informatyki z komputerami. Istnieje jednak także drugi termin „Informatics" także tłumaczony na język polski jako *informatyka*.

Informatics or computer science?

Computer science

New Jersey Institute of Technology

Computer science is the study of information: its structure, its representation, and its utilization. This includes the **theory, analysis, design, efficiency, implementation, application of computer programs (software) and computer equipment (hardware)** for developing computerized information processing systems in response to users' needs.

Computer Science Department, College of Saint Benedict, Saint John's University (I)

Computer science is the study of computation. The computation we study is rarely arithmetic, though – often the computation is more symbolic. We might ask about how to compute a good move in chess. Or we might ask how to draw a picture of a three-dimensional scene. As computer scientists, we look for models of computation. And we ask what we can – or cannot – do with these models.
Computer scientists learn to program computers, because a program is an excellent way of precisely describing a particular computational technique. [...]

Computer Science Department, College of Saint Benedict, Saint John's University (II)

Thus, a computer science student can expect to study all of the following questions.

- How do you design an algorithm, (a step by step plan), to solve a problem?
- How do you write a computer program to implement your plan?
- How can you analyze a program's speed?
- How does a computer work to execute a program?
- What social responsibilities do programmers have?
- How can we develop reliable software systems?
- What are the mathematical properties of computation?

You can expect a large emphasis on computer programming and on mathematics.

Computer Science Department, College of Saint Benedict, Saint John's University (III)

Equally important is what computer science **IS NOT**. **It does not emphasize the use of computers** in a corporate environment. You can liken it to the difference between an aerospace engineer and an airplane pilot; computer science is more like aerospace engineering. Computer science students do not learn how to use spreadsheets, word processors and other application programs as part of their study of computer science, but may develop those skills on their own or through workshops offered on campus. When we study operating systems or networks, we emphasize the internals, not how they should be configured for use. If you want to study these topics, you want to look for an information technology or information systems program. [...]
We also are not computer engineering. Computer engineering emphasizes how computer hardware works. **Computer science students learn about the fundamentals, but only as much as needed to understand how computer software works.** [...][a]

[a] http://www.csbsju.edu/computerscience/curriculum/, Computer Science

Informatics or computer science?

Computer Science Department, University of Cambridge

There is unfortunately a lot of confusion over these terms in the general population and in schools [...]. **Computer Science is the study of information and computation.** The other terms are more vocational and mostly describe training courses in how to use particular pieces of software. Think of it this way: someone with an ICT [Information Communications Tecnology] qualification will know how to use a program like Word. Someone with a Computer Science qualification will know how to create a program like Word, and will also know how to make it easier to use, how to make it work on a variety of machines, how to make it easy to add additional functionality, how to fix bugs in it, how to make it communicate with other pieces of hardware or software, how to market it and how to deal with any legal or copyright problems with it. They will understand the theoretical basis underlying the program. They will also know how to do a million other things besides.[a]

[a] http://www.cl.cam.ac.uk/admissions/undergraduate/myths/#CSIT, Computer Science Department, University of Cambridge, dostęp 2009-09-05

Informatyka w sensie odpowiednika terminu *Computer science*

Informatyka jest studiowaniem teoretycznych podstaw informacji (powstawania, przepływu, składowania), obliczeń i praktycznych technik pozwalających na ich implementację i wykorzystanie w systemach komputerowych. Często opisywana jest jako studiowanie algorytmicznych procesów wytwarzających, opisujących, przekształcających i składujących informację. Powołując się na Petera Denninga[a], zasadnicze pytanie na jakie daje odpowiedź informatyka to „Co można efektywnie zautomatyzować".

[a] Denning, P.J., *Computer Science: The Discipline*, Encyclopedia of Computer Science, 2000

Informatics or computer science?

School of Informatics, The University of Edinburgh

Informatics is the study of the structure, behaviour, and interactions of natural and engineered computational systems.

Informatics studies the representation, processing, and communication of information **in natural and engineered systems. It has computational, cognitive and social aspects.** The central notion is the transformation of information – whether by computation or communication, whether by organisms or artifacts.

Informatics has many aspects, and encompasses a number of existing academic disciplines – Artificial Intelligence, Cognitive Science and Computer Science, [...] Mathematics, Electronics, Biology, Linguistics and Psychology.[a]

[a] http://www.inf.ed.ac.uk/about/vision.html, School of Informatics, The University of Edinburgh, dostęp 2009-10-05

Informatics or computer science?

School of Informatics, The University of Edinburgh

Informatics is the study of the structure, behaviour, and interactions of natural and engineered computational systems.

Informatics studies the representation, processing, and communication of information **in natural and engineered systems. It has computational, cognitive and social aspects.** The central notion is the transformation of information – whether by computation or communication, whether by organisms or artifacts.

Informatics has many aspects, and encompasses a number of existing academic disciplines – Artificial Intelligence, Cognitive Science and Computer Science, [...] Mathematics, Electronics, Biology, Linguistics and Psychology.[a]

[a] http://www.inf.ed.ac.uk/about/vision.html, School of Informatics, The University of Edinburgh, dostęp 2009-10-05

Informatics or computer science?

School of Informatics, The University of Edinburgh

Informatics is the study of the structure, behaviour, and interactions of natural and engineered computational systems.

Informatics studies the representation, processing, and communication of information **in natural and engineered systems. It has computational, cognitive and social aspects.** The central notion is the transformation of information – whether by computation or communication, whether by organisms or artifacts.

Informatics has many aspects, and encompasses a number of existing academic disciplines – Artificial Intelligence, Cognitive Science and Computer Science, [...] Mathematics, Electronics, Biology, Linguistics and Psychology.[a]

[a] http://www.inf.ed.ac.uk/about/vision.html, School of Informatics, The University of Edinburgh, dostęp 2009-10-05

Department of Informatics, Donald Bren School of Information and Computer Science

What is informatics?

- Informatics combines aspects of software engineering, human-computer interaction, and the study of organizations and information technology.
- In European universities, informatics is the term most often used for computer science.
- Computer science studies computers; informatics studies computers **and people**.

Informatyka w sensie odpowiednika terminu *Informatics*

Informatyka jest studiowaniem systemów pozyskujących, reprezentujących, przetwarzających i wytwarzających informację włączając w to wszystkie obliczeniowe, kognitywne i społeczne aspekty.

Zasadniczym przedmiotem zainteresowania jest przetwarzanie (przekształcanie) informacji czy to przez procesy obliczeniowe czy komunikacyjne, czy to przez organizmy żywe czy urządzenia. W tym sensie informatykę należy postrzegać jako dziedzinę znacznie szerszą niż informatyka w sensie *Computer Science*.
Można powiedzieć, że informatyka (w sensie *Informatics*) ogólnie pojęty aspekt pozyskiwania, przetwarzania, składowania itd. informacji rozciąga zarówno nad maszynami (komputery) jak i istotami żywymi a ogólnie, wszystkim tym co ma jakikolwiek związek z informacją.

Za pierwsze „urządzenie" liczące (a raczej zliczające) uważa się kawałki kości z wykonaną odpowiednią ilością nacięć (ang. *tally stick*)

- 35000 p.n.e., kość udowa pawian z 29 nacięciami (Góry Lebombo, Centralna Afryka Równikowa).
- 30000 p.n.e., kość wilka z 57 nacięciami pogrupowanymi po 5, (Dolni Vestonice, Morawy).
- 20000 p.n.e., kość strzałkowa pawiana z licznymi nacięciami w kilku grupach i trzech kolumnach (Ishango, terenach Parku Narodowego Virunga, Demokratyczna Republika Konga). Pierwotnie znalezisko uważano za przykład, potwierdzający wykorzystanie tego typu narzędzi do zliczania, ale obecnie niektórzy badacze sugerują, że informacje zapisane na kości są czymś więcej niż prostym przykładem zliczania i dowodzą znacznie większej świadomości matematycznej (mówi się np. o liczbach pierwszych).

Abakus (3000 lat p.n.e.)

- Pierwsze abakusy będące dziełem ludów sumeryjskich pochodzą mniej więcej z 3000 r. p.n.e.
- Istota działania i problem dziesiątkowania.
- Rok 200 p.n.e. – chiński abakus nazywany *suanpan* i system bi-quinarny wykorzystywany w takich historycznych już komputerach jak IBM 650 (1953), UNIVAC 60 (1952) czy UNIVAC LARC (1960).

bi-quinary coded decimal (IBM 650)

0	01	10000
1	01	01000
2	01	00100
3	01	00010
4	01	00001
5	10	10000
6	10	01000
7	10	00100
8	10	00010
9	10	00001

Zero

500 r. p.n.e. – pierwsze znane przykłady użycia zera przez matematyków indyjskich.

Opis gramatyki (Panini, V w.)

500 r. p.n.e. – Panini, za pomocą 3959 reguł podał wysoce usystematyzowany, opis gramatyki Sanskrytu znany jako Ashtadhyayi („Osiem rozdziałów"). W swoim upisie używając m.in. **metareguł**, **transformacji** i **rekursji** spowodował, że gramatyka ta uważana jest za pierwszy **system formalny**. Występująca obecnie w powszechnym użyciu notacja BNF (Backus-Naur Form) opisu gramatyk bezkontekstowych wykorzystywana jako formalny sposób opisu gramatyki języków programowania, zbioru instrukcji czy protokołów jest do tego stopnia podobna do gramatyki Paniniego, że bywa nazywana też Panini-Backus form.

Zero

500 r. p.n.e. – pierwsze znane przykłady użycia zera przez matematyków indyjskich.

Opis gramatyki (Panini, V w.)

500 r. p.n.e. – Panini, za pomocą 3959 reguł podał wysoce usystematyzowany, opis gramatyki Sanskrytu znany jako Ashtadhyayi („Osiem rozdziałów"). W swoim upisie używając m.in. **metareguł**, **transformacji** i **rekursji** spowodował, że gramatyka ta uważana jest za pierwszy **system formalny**. Występująca obecnie w powszechnym użyciu notacja BNF (Backus-Naur Form) opisu gramatyk bezkontekstowych wykorzystywana jako formalny sposób opisu gramatyki języków programowania, zbioru instrukcji czy protokołów jest do tego stopnia podobna do gramatyki Paniniego, że bywa nazywana też Panini-Backus form.

Przykład notacji BNF

Dla przykładu, używając notacji BNF, określimy liczby całkowite przy pomocy następujących reguł

1. `<znak minus>::= -`
   Przykład wartości: -
2. `<zero>::= 0`
   Przykład wartości: 0
3. `<cyfra niezerowa>::= 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9`
   Przykład wartości: 1, 2, 3
4. `<ciąg cyfr> ::= <cyfra> | <cyfra><ciąg cyfr>`
   Przykład wartości: 0, 1, 01, 001, 23, 45, 99, 10023, 000001
5. `<liczba calkowita dodatnia>::= <cyfra niezerowa> | <cyfra niezerowa><ciąg cyfr>`
   Przykład wartości: 1, 2, 34, 56, 406, 556066
6. `<liczba calkowita ujemna>::= <znak minus><cyfra niezerowa> | <znak minus><cyfra niezerowa><ciąg cyfr>`
   Przykład wartości: -1, -2, -34, -56, -406, -556066

Opis pierwszego algorytmu (Euklides, IV w)

400 r. p.n.e. – w „Elementach” Euklides przedstawia algorytm znajdowania największego wspólnego dzielnika. I choć algorytm nazywa się algorytmem Euklidesa to faktycznie wymyślił go Eudoksos z Knidos (IV wiek p.n.e.), a Euklides jedynie zawarł go w swoim dziele.

Pojęcie nieskończoności (IV-IIIw.)

W indyjskich (dżinijskich) tekstach z okresu IV i III w p.n.e. pojawia się pojęcie nieskończoności i to nie tylko w sensie filozoficznym (bo w takim znane było jeszcze wcześniej), ale jako koncept matematyczny związany z liczbami.

That is whole, this is whole
From the whole, the whole arises
When the whole is taken from the whole
The whole still will remain

Binarny system liczbowy (Pingala, III w.)

W trzecim stuleciu indyjski pisarz Pingala wykorzystał zaawansowane koncepcje matematyczne opisując wzorce prozodyczne (metryczne) a więc określające rytmiczną strukturę wersu. Wtedy też zanotowano pierwsze wykorzystanie binarnego systemu liczbowego.

Pierwszy „komputer analogowy” (150–100 r. p.n.e.)

W roku 1901 naszej ery we wraku obok greckiej wyspy Antykithiry (Antikythera), leżacej pomiędzy Kithirą i Kretą, odkryto datowany na lata 150–100 p.n.e. starożytny mechaniczny przyrząd przeznaczony do obliczania pozycji ciał niebieskich. Do czasu XVIII-wiecznych zegarów nie jest znany żaden mechanizm o podobnym stopniu złożoności. Mechanizm ten można uważać za pierwszy analogowy komputer.

Liczby ujemne (Chiny, I w.)

Na ostatnie stulecie datuje się pierwsze wykorzystanie liczb ujemnych, którego dokonali Chińczycy.

Logarytm (Indie, II w.)

W II w. umiejscawia się wynalezienie logarytmu przez dżinijskich matematyków.

Zero, liczby ujemne, pozycyjny zapis liczb (Brahmagupta, 600 r.)

W roku 600 indyjski matematyk Brahmagupta **zdefiniował pojęcia zera i liczb ujemnych** a także **opisał pozycyjny sposób zapisu liczb** (ang. *place-value numeral system*). Jak zauważyliśmy zero pojawiło się już znacznie wcześniej, ale dopiero Brahmagupta nadał jemu indywidualny charakter wyróżniając tą liczbę spośród innych. Oto jak w rodziale ósmym swojego dzieła „Brahmasphutasiddhanta" Brahmagupta opisuje działania na liczbach ujemnych

- The sum of two positives is positive, of two negatives negative; of a positive and a negative [the sum] is their difference; if they are equal it is zero. The sum of a negative and zero is negative, [that] of a positive and zero positive, [and that] of two zeros zero.
- A negative minus zero is negative, a positive [minus zero] positive; zero [minus zero] is zero. When a positive is to be subtracted from a negative or a negative from a positive, then it is to be added.

Pojęcie *algorytmu* (Muhammad ibn Musa al-Chorezmi (al-Khawarizmy), 825 r.)

Słowo **algorytm** pochodzi od nazwiska perskiego astronoma i matematyka żyjącego na przełomie VIII i IX w n.e. W 825 roku Muhammad ibn Musa al-Chorezmi (al-Khawarizmy) napisał traktat zatytułowany *O obliczeniach na liczbach indyjskich*, w którym podał wiele precyzyjnych opisów dotyczących różnych matematycznych reguł (np. dodawania czy mnożenia liczb dziesiętnych). W XII wieku dzieło to zostało przetłumaczone na łacinę jako *Algoritmi de numero Indorum*, co należało rozumieć następująco: *Algoritmi o liczbach Indyjskich*. Pojawiające się tutaj po raz pierwszy słowo *Algoritmi* było oczywiście inaczej zapisanym nazwiskiem matematyka. Większość ludzi rozumiała jednak tytuł bardziej jako *Algorytmy o liczbach Indyjskich* a stąd już blisko do *Algorytmy na liczbach indyjskich (arabskich)*. W ten oto sposób precyzyjnie opisaną metodę obliczeniową zaczęto nazywć algorytmem (łac. *algorismus*).

Kryptografia, kryptoanaliza (Alkindus, 801 –873.)

Żyjący w latach 801–873 Abu Yusuf Ya'qub ibn Ishaq al-Kindi znany także jako Alkindus uważany jest za pioniera kryptografii i kryptoanalizy. Wprowadził metodę analizy częstotliwościowej (ang. *frequency analysis method*) pozwalającej w oparciu o statystyczny rozkład liter w tekście dokonać jego deszyfracji.

Mechaniczny kalkulator (Leonardo da Vinci, 1492 r.)

W roku 1492 Leonardo da Vinci (1452–1519) sporządza szkice urządzenia składającego się z zachodzących na siebie kół zębatych. I choć konstrukcja nigdy nie powstała, uważa się, że mogł to być projekt mechanicznego kalkulatora pozwalającego na dodawanie i odejmowanie liczb. Da Vinci stworzył także plany mechanicznego człowieka, czyli pierwowzoru współczesnych robotów.

Pałeczki Napiera (Napier, 1617 r.)

W 1588 Joost Buerghi (1552–1632) odkrywa logarytm naturalny a w 1614 logarytmy dziesiętne wprowadza Henry Briggs (1561 – 1630). W roku 1617 John Napier (1550–1617) publikuje *Rabdologiae* w którym opisuje urządzenie wspomagające proces mnożenia, dzielenia a także wyciągania pierwiastków kwadratowych za pomocą specjalnych pałeczek (tzw. pałeczki Napiera). Wzorując się na ideii logarytmu, specjalnie zaprojektowane pałeczki pozwalały sprowadzić np. mnożenie do serii dodawań a dzielenie do serii odejmowań. Pokażemy ideę działania pałeczek na dwóch prostych przykładach.

Pierwsza mechaniczna maszyna licząca (Wilhelm Schickard, 1623 r.)

W tym samym czasie żył Wilhelm Schickard (1592–1635), który jest uznawany za twórcę pierwszej mechanicznej maszyny liczącej. W 1623 zbudował on maszynę (nazywaną zegarem liczącym) czterodziałaniową wykorzystującą pałeczki Napiera. Maszyna była zdolna do dodawania i odejmowania liczb 6-cio cyforwych sygnalizując przy tym błędy przepełnienia (czyli błędy powstające gdy wynik nie daje się wyrazić za pomocą zaplanowanej ilości cyfr).

Sumator arytmetyczny (Blaise Pascal, 1642 r.)

W 1642 r. francuski matematyk Blaise Pascal (1623–1662) buduje sumator arytmetyczny.

Stepped Reckoner (Gottfried Wilhelm Leibnitz, 1671 r.)

W 1671 niemiecki matematyk Gottfried Wilhelm Leibniz (1646–1716) zbudował maszynę nazywaną *Stepped Reckoner*. Jej 16-cyfrowa wersja mogła

- dodać (odjąć) 8-cyfrową liczbę do (od) 16-cyfrowej;
- wykonać mnożenie dwóch 8-cyfrowych liczb;
- podzielić liczbę 16-cyfrową przez 8-cyfrową.

Dodawanie i odejmowanie odbywało się w jednym cyklu (tj. przy jednym obrocie korby). Mnożenie i dzielenie odbywało się „cyfra po cyfrze". Operacje te mogły odbywać się także na uprzednio otrzymanym wyniku przechowywanym w akumulatorze, dzięki czemu można było policzyć pierwiastek jako serię dzieleń i dodawań.
Z punktu widzenia historii informatyki istotna jest informacja, iż Leibniz opisał system binarny, będący podstawą repreznetacji danych we współczesnych komputerach.

Maszyna różnicowa (Babbage, 1822 r.)

W 1786 Johann Helfrich von Müller (1746 – 1830) podał ideę maszyny różnicowej służącej do automatycznego wylicznia tabelaryzowanych wartości wielomianów. Zaproponowane przez niego rozwiązanie wykorzystał Charles Babbage (1791–1871) probując skonstruować swoje wersje maszyny różnicowej. Projekt pierwszej maszyny zaproponowany został Królewskiemu Stowarzyszeniu Astronomicznemu (ang. *Royal Astronomical Society*) w roku 1822 a jej ulepszona wersja powstawała w latach 1847–1849. Niestety pomimo znacznych nakładów finansowych poniesionych na obie maszyny, żadna z nich nie została zbudowana. Spowodowane to było zapewne znacznym stopniem komplikacji[a] i trudnościami technologicznymi związanymi z wykonaniem precyzyjnych mechanizmów. Istotną różnicą pomiędzy wcześniej konstruowanymi maszynami a maszyną różnicową, było to, że po nastawieniu danych początkowych wszelkie dalsze obliczenia odbywały się automatycznie, bez udziału człowieka, za wyjątkiem samego faktu napędzania maszyny.

[a] 25000 częci, jej łączna waga to 13500kg, wysokość 8 stóp, tj. 2,4m. Udoskonalona wersja składać się miała z 4000 części, jej waga to 3000kg i 6 stóp wysokości.

Maszyna analityczna (Babbage, 1834 r.)

Praktycznie już na samym początku prac nad maszyną różnicową, bo w 1834, Babbage rozpoczął projektowanie, rewolucyjnego jak na tamte czasy, urządzenia nazywanego **maszyną analityczną**. Maszyna ta miała składać się z następujacych bloków funkcjonalnych

- magazynu (odpowiednik pamięci w dzisiejszych komputerach), miał służyć do przechowywania danych oraz wyników z przeprowadzanych na nich operacji;
- młyna (jednostka licząca), odpowiednik dzisiejszej jednostki arytmetyczno-logicznej, miał wykonywać proste działania arytmetyczne;
- mechanizmu sterującego (jednostka sterująca), kierującego działaniem całego urządzenia i w założeniach programowalnego.

Wymieńmy najważniejsze parametry maszyny analitycznej

- Siłą napędową maszyny miał być silnik parowy.
- Maszyna miała mieć ok. 30 metrów długości i 10 metrów szerokości.
- Dane wejściowe (program oraz dane) wprowadzane miały być za pomocą **kart dziurkowanych**, powszechnie wykorzystywanych w tamtym czasie przez krosna mechaniczne[a]
- Do sygnalizowania i udostępniania danych wyjściowe maszyna pisiadała drukarkę i dzwonek. Dodatkowo wyniki mogły być przedstawione na kartach dziurkowanych w celu dalszego ich wykorzystania.

[a] Pomysł „programowania" za pomocą kart dziurkowanych wzoru tkanego przez krosna pochodzi od Joseph-Marie Jacquard'a i datowany jest na rok 1801.

Najważniejsze parametry maszyny analitycznej (c.d.)

- Maszyna wykorzystywała stałoprzecinkową arytmetykę oraz system dziesiętny.
- Magazyn mógł pomieścić 1000 liczb 50-cio cyfrowych.
- Młyn mógł wykonać wszystkie cztery operacje arytmetyczne, porównanie oraz obliczyć pierwiastek kwadratowy.
- Dostępna była „instrukcja" skoku warunkowego.
- Operacja dodawania (odejmowania) trwać miała ok. 3 sekund natomiast mnożenie (dzielenie) zajmować miało 2-4 minut[a].

[a] Jest to dosyć ciekawa „własność" wszystkich maszyn liczących: zwykle mnożenie zajmuje znacznie więcej (o rząd lub dwa) czasu niż dodawanie.

Ada

O maszynie Babbage'a, mimo iż nigdy nie zbudowanej, warto też pamiętać z innego powodu. To na nią bowiem powstały, w oparciu o dostępną „specyfikacje", pierwsze programy, kórych autorką jest Ada Augusta, hrabiny Lovelace, córka George'a Byrona. Tym samym stała się ona pierwszym pierwszym programistą w historii a potwierdzeniem jej zasług jest nadanie jej imienia jednemu z najbardziej uniwersalnych i zaawansowanych obecnie języków programowania, jakim jest **Ada**.

Algebra Boole'a (George Bool, 1854 r..)

W 1854 George Boole opublikował swoją najważniejszą pracę, *An Investiagtion into The Laws of Thought on Which Are Founded The Mathematical Theories of Logic and Probabilities* (Badanie praw myślenia, na którym oparte są matematyczne teorie logiki i prawdopodobieństwa), w której wykazał, jak prawa logiki podane przez Arystotelesa mogą stanowić przedmiot rachunków. Przedstawione idee stanowią podstawę działania współczesnych komputerów.

Karty perforowane jako nośnik danych (Herman Hollerith, 1890 r.)

Spis powszechny w 1890r.

Trioda (Lee De Forest, 1906 r.)

Od okresu międzywojennego możemy już mówić o rozwoju technologii elektronicznej, dzięki wynalezieniu w 1906 przez Lee De Forest'a lampy elektronowej (triody). Lampy elektronowe w ogólności służą do wzmacniania, generacji, przekształcania itp. sygnałów elektrycznych. Trioda składa się z trzech elektrod – anody, katody i siatki. Umożliwia sterowanie przepływem elektronów z katody do anody przez zmianę napięcia na siatce a zatem **umożliwia budowanie wzmacniaczy sygnałów elektrycznych.**

Maszyna Turinga (Alan Turing, 1936 r.)

W pracy z 1936 roku „On Computable Numbers" Alan Turing podał opis niezwykle prostej teoretycznej maszyny (nazywanej maszyną Turinga) zdolnej do wykonania dowolnych obliczeń matematcznych pod warunkiem, że dają się one przedstawić jako algorytm. I choć zbudowanie takiej maszyny mimo jej prostoty nie jest możliwe, to jest ona dla nas istotna z tego powodu, że **żaden istniejący komputer nie ma większej mocy obliczeniowej niż ta prosta maszyna**. Eksperyment myślowy z maszyną Turinga pokazuje też bardzo wyraźnie, że obliczenia na współczesnych komputerach to nic innego jak elementarne manipulowanie symbolami.

Z1 (Konrad Zuse, 1938 r.)

Pierwszy, co prawda mechaniczny, komputer programowalny. Zawierał on praktycznie wszystkie, wyraźnie odseparowane od siebie (w sensie pełnionej roli), współcześnie znane podukłady: jednostkę zmiennoprzecinkową[a], jednostkę sterującą, pamięć, urządzenia wejścia/wyjścia. Wykorzystywał system binarny (dane wprowadzano i wyniki otrzymywano w systemie dziesiętnym) i liczby zmiennoprzecinkowe a dane wprowadzane były za pomocą perforowanej taśmy filmowej 35mm. Otrzymany przez Zusego patent[b] wskazywał także na znajomość idei identycznego traktowania danych i kodu programu, tj. przechowywania ich w modyfikowalnej pamięci komputera, choć sam komputer Z1 kod programu pobierł tylko z taśmy a nie z pamięci. Używane było 9 rozkazów[c] o czasie wykonania od 1 do 20 cykli co przy „zegarze" 1Hz dawało średnią prędkość dla dodawania 5 sekund a dla mnożenia 10. W pamięci mógł przechowywać 64 słowa o długości 22 bitów (176 bajtów). Waga całości to 1000kg.

[a] Brak było jednostki logicznej.
[b] Z23139/GMD Nr. 005/021

Z2 (Konrad Zuse, 1939 r.)

Zmodyfikowana wersja komputera Z1 oznaczana jako Z2 zbudowana została w roku 1939. W tym przypadku jednostka arytmetyczno-logiczna składała się z przekaźników elektrycznych co wraz ze zwiększeniem częstotliwości pracy do 5Hz zaowocowało skróceniem czasu dodawania do 0.8 sekundy a mnożenia do 3 sekund. Zamiast 22 bitowej arytmetyki zmiennoprzecinkowej wykorzystywał 16 bitową arytmetykę stałoprzecinkową. Zapotrzebowanie na moc wynosiło 1000W.

Z3, Z4 (Konrad Zuse, 1942, 1950 r.)

W maju 1942 Zuse zaprezentował maszynę Z3 będącą wersją maszyny Z1 zbudowaną na przekaźnikach.
Zbudowany w 1950 Z4 wykorzystywany był w Instytucie Matematyki Stosowanej Konfederacyjnej Wyższej Szkoły Technicznej (ETH) w Zurychu przez 5 lat i był to jedyny działający wówczas komputer w Europie.

Język programowania wyższego poziomu (Konrad Zuse, 1945 r.)

W 1945 Zuse opracowuje pierwszy język programowania wyższego poziomu – Plankalkül.

Mark I (IBM, 1937 r.)

W 1937 r. rozpoczyna pracę zespół konstruktorów złożony z pracowników firmy IBM, kierowany przez Howarda Aikena. Wynikiem ich prac było zbudowanie w roku 1944 największego w historii kalkulatora elektromechanicznego nazywanego IBM Automatic Sequence Controlled Calculator (ASCC) a inaczej Harvard Mark I.
Programowanie tej maszyny polegało na odpowiednim łączeniu kabelkami gniazd w specjalnej tablicy sterującej. Dane wprowadzano za pomocą kart dziurkowanych, wyniki wyprowadzano na taśmę perforowaną lub drukowano za pomocą elektrycznych maszyn do pisania.

Mark I (IBM, 1937 r.) (c.d.)

MARK I miał długość 16 metrów, wysokość 2,4 metra, głębokość 61cm. Składał się z 760 tys. części, w tym z 17 480 lamp elektronowych. Zawierał ponad 800 km przewodów z trzema milionami połączeń. Całkowita waga kształtowała się na poziomie 4500kg. Obsługę stanowiło 10 osób. Wykonywał 3,5 dodawania na sekundę oraz 1 dzielenie na 11 sekund. Częstotliwość pracy wynosiła 100 kHz. Szacowano, iż zastępuje on pracę 100 rachmistrzów wyposażonych w arytmometr mechaniczny. Najbardziej znaną programistką tej maszyny była Grace Hopper, znana m.in. z wprowadzenia do języka informatyków słowa *bug* (pluskwa, owad). Bardzo ciekawe informacje i zdjęcia maszyny przedstawione zostały w `http://www-03.ibm.com/ibm/history/exhibits/markI/markI_intro.html`

ENIAC (1942 r.)

W 1942 r. zespół specjalistów pod kierunkiem Johna Mauchley'ego i Johna Eckerta projektuje i buduje maszynę ENIAC (ang. Electronic Numerical Integrator And Computer). Jest to pierwsza maszyna, w której zastosowano wyłącznie elementy elektroniczne (lampy elektronowe), i jest uznawana powszechnie za pierwszy kalkulator elektroniczny.
Programowanie ENIAC-a polegało na ręcznym ustawianiu przełączników oraz wymianie specjalnych tablic programowych. Długość komputera wynosiła 15 metrów, jego szerokość to 9 metrów, waga 30 ton, składał się z ok. 18 000 lamp elektronowych. Liczby były pamiętane w systemie dziesiętnym, był on w stanie wykonać 5000 dodawań na sekundę, oraz od 50 do 360 dzieleń na sekundę.

Architektura von Neumannowska (1945 r.)

W 1945 r. do projektu EDVAC (ang. Electronic Discrete Variable Automatic Computer) przyłącza się John von Neumann (1903–1957), który w notatce zatytułowanej *First Draft of a Report on the EDVAC* zaproponował rozwiązania mające na celu zbudowanie komputera ogólnego przeznaczenia przechowującego program w pamięci. W ten oto sposób zrodziła się architektura według której są budowane komputery do dnia dzisiejszego. Została ona nazwana **von neumannowską**. Była ona wynikiem pracy von Neumanna nad problemem przechowywania w pamięci komputera, zarówno danych, podlegających przetwarzaniu, jak i programu, który na tych danych miał działać. Prace te umożliwiły odejście od sztywnych metod programowania sprzętowego (przełączanie kabelków czy zworek) i zastąpienie ich programowaniem wewnętrznym, poprzez umieszczenie w pamięci maszyny programu sterującego przetwarzaniem danych.

Architektura von Neumannowska (1945 r.) (c.d.)

Architektura von neumannowska wyróżniała następujące elementy składowe: pamięć złożoną z elementów przyjmujących stany 0 i 1, arytmometr wykonujący działania arytmetyczno-logiczne, jednostkę sterującą. Sterowanie odbywało się za pomocą programu, który był umieszczany w pamięci. Stanowiło to duży skok ideowy w stosunku do wcześniejszych koncepcji, w których program był zapisywany na kartach perforowanych i bezpośrednio z nich odczytywany oraz uruchamiany.

Architektura von Neumannowska (1945 r.) (c.d.)

W maszynie von neumannowskiej zarówno program, jak i dane, znajdowały się w pamięci fizycznej. Sam program mógł modyfikować zawartość tej pamięci, a co za tym idzie, mógł sam się modyfikować. Program składał się z ciągu instrukcji, które były pobierane i rozpoznawane przez jednostkę sterującą w takt zegara sterującego pracą całego komputera. Instrukcje te musiały odpowiadać poleceniom zakodowanym przez twórców układu elektronicznego. Taka idea powodowała, że nie było już różnicy pomiędzy danymi a rozkazami, wszystkie one były kodowane za pomocą systemu binarnego.

Złote myśli

I think there is a world market for maybe five computers.

Thomas Watson, chairman of IBM, 1943

Złote myśli

There is no reason anyone would want a computer in their home.

Ken Olson, president, chairman and founder of DEC

Złote myśli

Computers in the future may weigh no more than 1.5 tons.

Popular Mechanics, forecasting the relentless march of science, 1949.

Komputery osobiste

1947 wynalezienie tranzystora – pierwszego i podstawowego składnika elektroniki cyfrowej i analogowej;

1958 wynalezienie układu scalonego – układu, który zawiera w sobie tranzystory zamknięte w jednej obudowie i realizujące pewne konkretne funkcje;

1964 komputer IBM S/360 – pierwszy superkomputer zwany do dziś Mainframe;

1964 graficzny interfejs użytkownika i mysz;

1971 Intel 4004 – zawierał 2,3 tys. tranzystorów, był taktowany zegarem 740 kHz, mógł zaadresować 1 kB pamięci dla danych oraz 4 kB pamięci programu;

1972 Intel 8008 – zawierał 3,5 tys. tranzystorów, mógł zaadresować do 16 kB RAM;

Komputery osobiste

1974 Intel 8080 – zawierał 4,8 tys. tranzystorów, mógł zaadresować do 64 kB RAM, lista poleceń składała się z 75 rozkazów;

1975 Altair 8800 – pierwszy komputer domowy oparty na procesorze Intel 8080, posiadał 256 bajtów RAM;

1976 procesor Zilog Z80 – modyfikacja Intel 8080, lista poleceń zawierała 176 rozkazów, prędkość zegara wynosiła 4 MHz;

1976 procesor Intel 8086 i 8088;

1977 komputer Apple II;

1979 procesor Motorola 68000;

1981 komputer IBM PC – pierwszy komputer rozpoczynający całą rodzinę istniejących do dziś komputerów osobistych (ang. Personal Computer), był oparty na procesorze 8088, posiadał 64 kB RAM;

Komputery osobiste

1982 procesor Intel 80286 – zawierał 134 tys. tranzystorów, mógł zaadresować do 16 MB RAM, był taktowany zegarem 6 MHz;

1983 komputer PC XT – oparty na procesorze Intel 8086;

1984 komputer PC AT – oparty na procesorze Intel 80286;

1985 procesor Intel 80386 – zawierał 275 tys. tranzystorów, był taktowany zegarem 16 MHz;

1989 procesor Intel 80486 – zawierał 1,18 mln tranzystorów, był taktowany zegarem 25 MHz;

1992 procesor Power PC – zawierał 2,8 mln tranzystorów, początkowo był taktowany zegarem 66 MHz;

1993 procesor Intel Pentium – zawierał 3,1 mln tranzystorów, początkowo był taktowany zegarem 60 MHz;

Komputery osobiste

1993 procesor DEC Alpha – zawierał 1,7 mln tranzystorów, 300 MHz;

1995 procesor Intel Pentium Pro – 5,5 mln tranzystorów, był taktowany zegarem 200 MHz;

1996 procesor Intel Pentium MMX;

1997 procesor Intel Pentium II – zawierał 7,5 mln tranzystorów, początkowo był taktowany zegarem 300 MHz;

1999 procesor Intel Pentium III – zawierał 9,9 mln tranzystorów, początkowo był taktowany zegarem 600 MHz;