

Co skrywają komputery?

O zerach i jedynkach, które napędzają dzisiejszy świat

Czym właściwie jest komputer?

- *Jak myślicie?*

Czym właściwie jest komputer?

- „Compute” – po angielsku „liczyć”
- Początkowo mówiono tak na ludzi, których pracą było wykonywanie obliczeń
- Tym mianem określa się także maszyny przeznaczone do wykonywania obliczeń



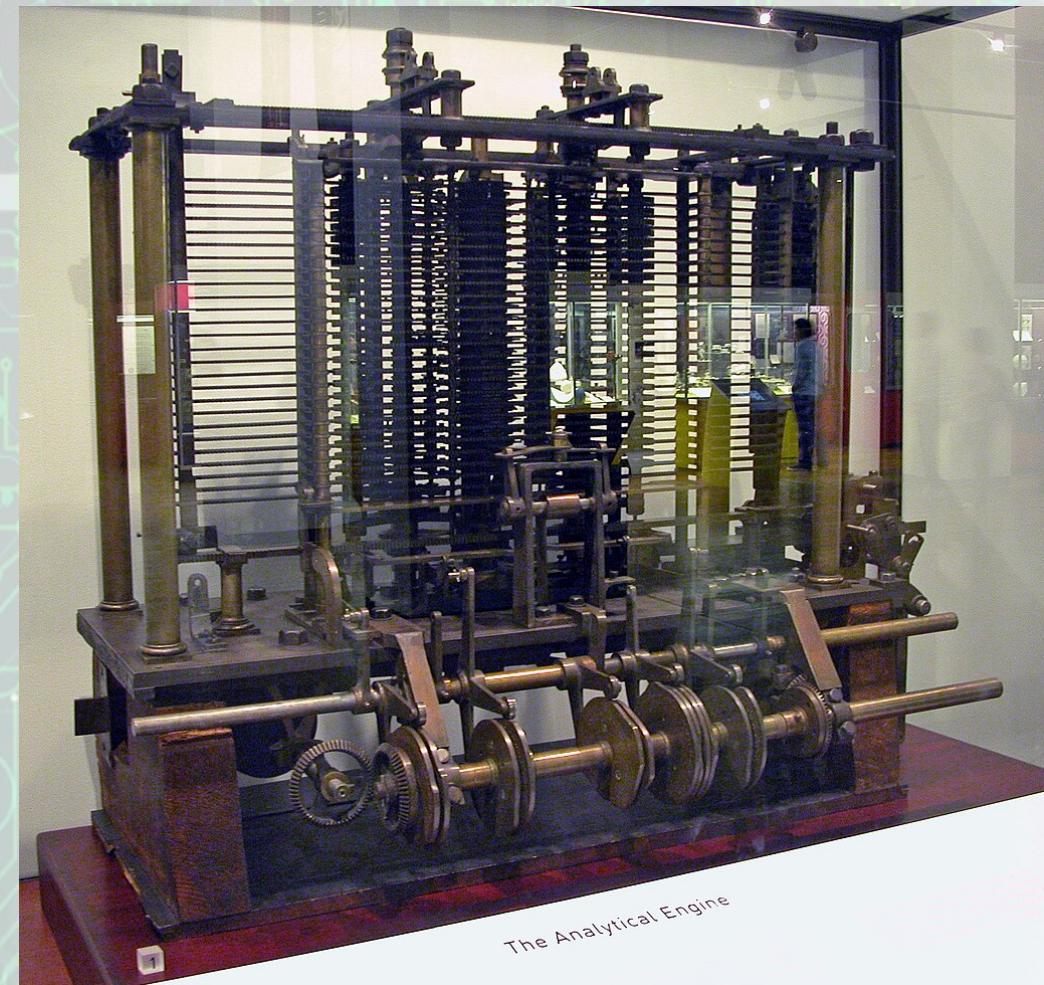
Ojciec Komputerów

- Alan Turing – brytyjski matematyk i wojskowy
- Mocno przyczynił się do złamania Enigmy (Bomba Turinga)
- Opracował matematyczne podstawy działania tzw. Komputera Ogólnego Przeznaczenia (Maszyny Turinga)



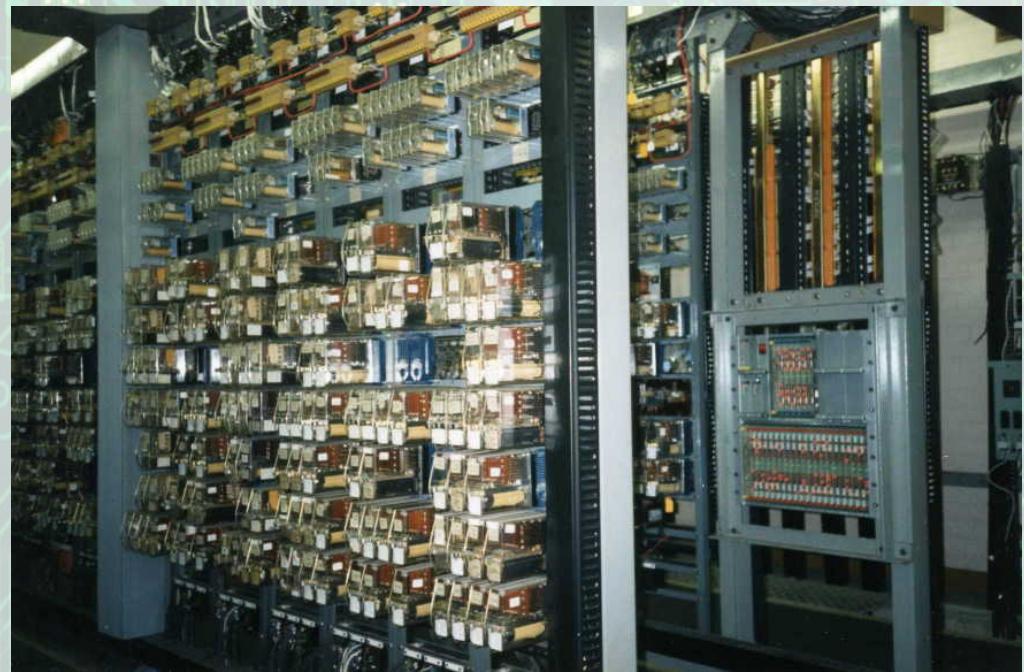
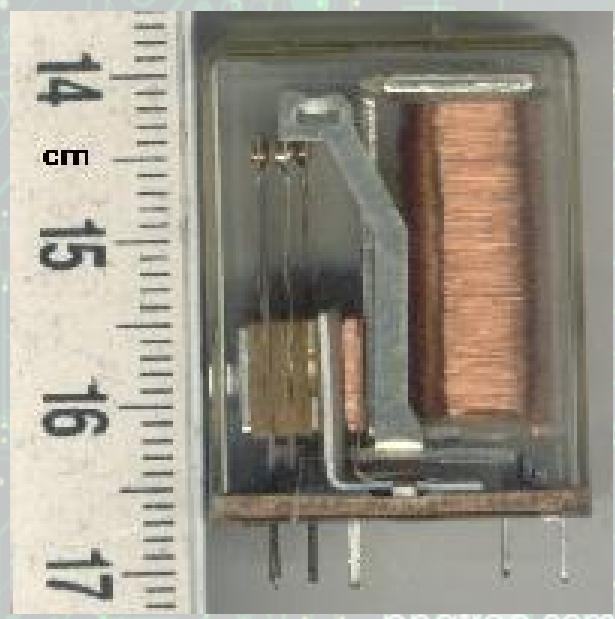
Komputery Analogowe

- Początkowo komputery były mechaniczne
- Często miały ograniczone zastosowanie
- Niektóre nie potrzebowaly nawet prądu:
https://youtu.be/aDN4s8ElxqE?si=_KckUDt74YSjSYzh&t=134
- Pierwszy komputer ogólnego przeznaczenia – Charles Babbage
- Pierwsza programistka – Ada Lovelace



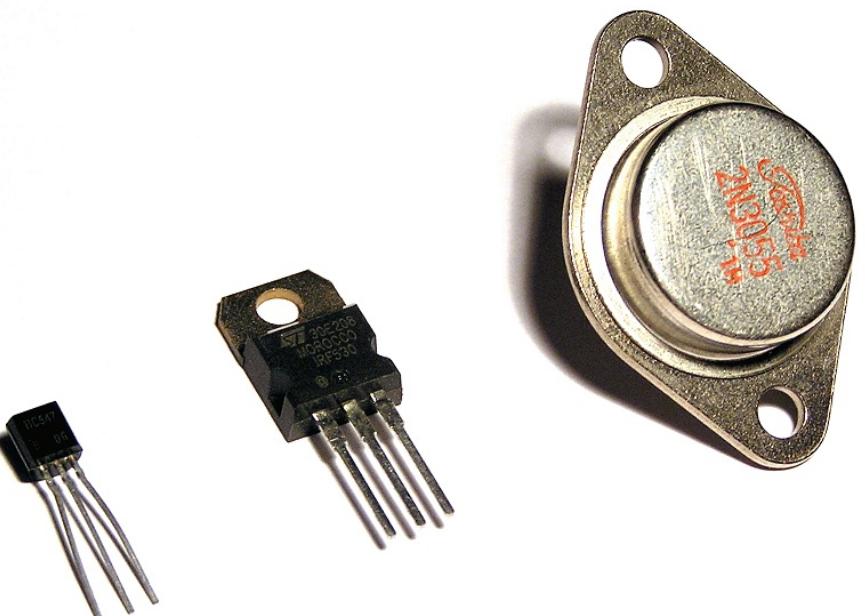
TRANZYSTOR!

- Przepływem prądu trzeba sterować
- Pierwotnie: lampy elektronowe i przełączniki elektromagnetyczne (które lubiły się psuć)



TRANZYSTOR!

- *Przełom: Tranzystor*
- *Przełącznik elektryczny bez ruchomych elementów*



System binarny

- Dlaczego?
- Trudno zbudować maszynę, która rozróżnia więcej niż dwie liczby
- Cyfry w informatyce nazywamy bitami
- 8 bitów tworzy 1 bajt
- Współczesne komputery posługują się liczbami o długości 32 albo 64 bity



System binarny

- Jak to działa?
- Podobnie jak system dziesiętny. Z dwiema różnicami:
 - Mamy tylko dwie cyfry
 - Kolejne pozycje nie oznaczają potęg dziesiątki, tylko dwójką

1000 100 10 1
 10^3 10^2 10^1 10^0

2137

2000 100 30 7
 1000×2 100×1 10×3 1×7

8 4 2 1
 2^3 2^2 2^1 2^0

1101

8 4 0 1
 8×1 4×1 2×0 1×1

Jak wygląda program?

- Procesor jest w gruncie rzeczy głupi – to maszyna, która po wprowadzeniu danej sekwencji bitów, wykonuje określone zadanie
- Te polecenia są ładowane z pamięci operacyjnej (RAM)
- Jak wygląda taki rozkaz? Bywa różnie, w zależności od konstrukcji procesora
- Na procesorach z rodziny x86 wygląda to następująco:

PROGRAM

```
0000000000401126 <main>:  
401126:    55                      push  %rbp  
401127:    48 89 e5                mov   %rsp,%rbp  
40112a:    48 83 ec 10            sub   $0x10,%rsp  
40112e:    c7 45 fc 00 00 00 00  movl  $0x0,-0x4(%rbp)  
401135:    c7 45 f8 00 00 00 00  movl  $0x0,-0x8(%rbp)  
40113c:    eb 1e                  jmp   40115c <main+0x36>  
40113e:    8b 45 f8                mov   -0x8(%rbp),%eax  
401141:    01 45 fc                add   %eax,-0x4(%rbp)  
401144:    8b 45 fc                mov   -0x4(%rbp),%eax  
401147:    89 c6                  mov   %eax,%esi  
401149:    bf 10 20 40 00            mov   $0x402010,%edi  
40114e:    b8 00 00 00 00            mov   $0x0,%eax  
401153:    e8 d8 fe ff ff            call  401030 <printf@plt>  
401158:    83 45 f8 01            addl  $0x1,-0x8(%rbp)  
40115c:    83 7d f8 09            cmpl  $0x9,-0x8(%rbp)  
401160:    7e dc                  jle   40113e <main+0x18>  
401162:    b8 00 00 00 00            mov   $0x0,%eax  
401167:    c9                      leave  
401168:    c3                      ret
```

To jest skomplikowane . . .

- Język który widzieliście to Assembler
- Istnieją wariaci, którzy korzystają z tego w pracy
- Przydatne do programowania prostych urządzeń
- Pisząc kod trzeba znać konstrukcję procesora
- Nie ma jakichś bardziej ludzkich języków?

Przedstawiamy C

- Oto ten sam program, ale napisany w języku C. Trochę bardziej zrozumiałe, prawda?

```
GNU nano 7.2                                basic_loop.c
#include<stdio.h>
int main(){
    int total = 0
    for (int i = 0; i < 10; i++){
        total += i;
        printf("%d\n", total);
    }
    return 0;
}
```

- Niestety, procesor nie potrafi wykonać kodu w takiej formie...

Kompilacja

- Aby wykonanie kodu C stało się możliwe, należy najpierw go przetłumaczyć do postaci zrozumiałej dla procesora (Assemblera)
- Proces ten nazywany jest komplikacją
- Wykonuje ją odpowiedni program – kompilator
- Kompilator przygotowuje kod odpowiedni dla danego procesora



Też możecie spróbować

- Kompilator C jest dostępny online
- Kompilacja i uruchomienie odbywa się automatycznie po wcisnięciu „Run”

<https://www.programiz.com/c-programming/online-compiler/>

Zadanie: Napisz program, który wyświetli wynik dodawania, odejmowania, mnożenia i dzielenia przez siebie dwóch liczb



Czy da się prościej?

- Języki kompilowane działają bezpośrednio na procesorze
- Istnieje inna opcja: możemy napisać program (interpreter), który będzie czytał nasz kod i go wykonywał
- Takie języki nazywamy interpretowanymi. Jednym z najbardziej znanych jest Python



Po co?

- Program w takim języku można łatwo przenieść na inny procesor / system (wystarczy, że istnieje interpreter)
- Program jest wykonywany w kontrolowanym środowisku
 - można przeciwdziałać błędom
- Użytkownik może w razie potrzeby łatwo zajrzeć do kodu
- Nie przejmujemy się szczegółami technicznymi, dlatego zasady języka są prostsze

Przykład

```
GNU nano 7.2                               fibonacci.py

def fibonacci(x):
    if x == 0 or x == 1:
        return 1
    return fibonacci(x-1) + fibonacci(x-2)

fibonacci(10)
```

```
GNU nano 7.2                               fibonacci.c

#include <stdio.h>

int fibonacci(int x){
    if(x == 1 || x == 2){
        return 1;
    }
    return fibonacci(x-1) + fibonacci(x-2);
}

int main() {
    printf("%d", fibonacci(10));
    return 0;
}
```

Pythona też możecie wypróbować

- Podobnie jak w przypadku C, mamy dostępny interpreter online:

<https://www.programiz.com/python-programming/online-compiler/>

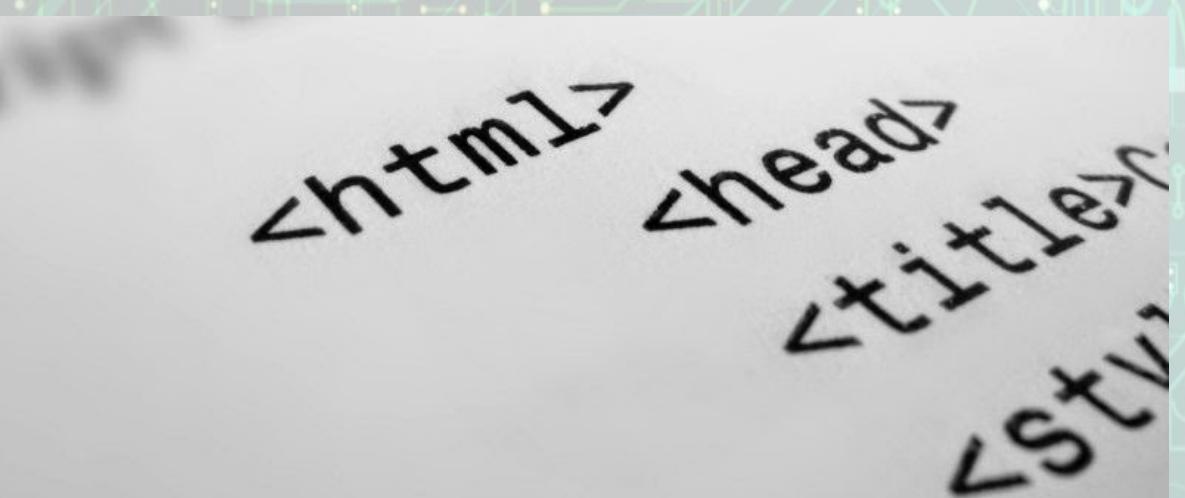
Zadanie: napisz program wyświetlający tabliczkę mnożenia

A co ze stronami internetowymi?

- Kod, który widzieliśmy do tej pory stanowił ciąg instrukcji do wykonania – tzn. program
- Kodem można się jednak także posłużyć do zapisania pewnej informacji – na przykład o treści i wyglądzie strony internetowej
- Przeglądarki wyświetlają strony właśnie na podstawie takich plików tekstowych – każda ma tzw. silnik renderowania, który interpretuje kod i tworzy obraz na ekranie

HTML

- Język HTML (HyperText Markup Language) służy do opisu zawartości strony internetowej
- Plik składa się ze znaczników
- Każdy element ma znacznik rozpoczynający i kończący
- Dzięki temu możemy umieszczać elementy wewnątrz innych



```
<html>
  <head>
    <title>
      <style>
        ...
      </style>
    </title>
  </head>
</html>
```

Przykład

← → C

file:///home/mikostoro/Documents/computer-secretes/computer-tables.html

Most Visited Fedora Docs Fedora Magazine Fedora Project User Communities Recent

HTML Tables

HTML tables start with a table tag.

Table rows start with a tr tag.

Table data start with a td tag.

3 Rows and 3 Columns:

100 200 300

400 500 600

700 800 900

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5 <h2>HTML Tables</h2>
6
7 <p>HTML tables start with a table tag.</p>
8 <p>Table rows start with a tr tag.</p>
9 <p>Table data start with a td tag.</p>
10 |
11 <hr>
12 <h2>3 Rows and 3 Columns:</h2>
13 <table>
14   <tr>
15     <td>100</td>
16     <td>200</td>
17     <td>300</td>
18   </tr>
19   <tr>
20     <td>400</td>
21     <td>500</td>
22     <td>600</td>
23   </tr>
24   <tr>
25     <td>700</td>
26     <td>800</td>
27     <td>900</td>
28   </tr>
29 </table>
30
31 <hr>
32
33 </body>
34 </html>
```

CSS

- CSS pozwala nadać określony wygląd stronie internetowej
- Plik HTML może zawierać kod CSS, albo nazwę pliku, z którego przeglądarka powinna pobrać styl



Przykład

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <style>
5 body {background-color: lightgreen;}
6 h1 {color: blue;}
7 p {color: red;}
8 </style>
9 </head>
10 <body>
11
12 <h1>This is a heading</h1>
13 <p>This is a paragraph.</p>
14
15 </body>
16 </html>
```

This is a heading

This is a paragraph.

JavaScript

- Większość stron, oprócz wyświetlania danych, wykonuje jakieś operacje – a zatem musi stać za nimi jakiś program
- Na stronach internetowych najczęściej używa się języka JavaScript – przeglądarki mają wbudowany interpreter
- Podobnie jak z CSS, możemy zawrzeć kod JS bezpośrednio w HTML albo załączyć osobny plik

Przykład

```
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5 <h2>Use JavaScript to Change Text</h2>
6 <p>This example writes "JavaScript has Arrived!" into an HTML element with id="demo":</p>
7
8 <p id="demo" onclick=change()> CLICK HERE </p>
9
10 <script>
11 function change(){
12     document.getElementById("demo").innerHTML = "JavaScript has Arrived!";
13 }
14 </script>
15
16 </body>
17 </html>
```

Use JavaScript to Change Text

This example writes "JavaScript has Arrived!" into an HTML element with id="demo":

[CLICK HERE](#)

Use JavaScript to Change Text

This example writes "JavaScript has Arrived!" into an HTML element with id="demo":

JavaScript has Arrived!

Skąd my te pliki właściwie bierzemy?

- Gdzieś po drugiej stronie internetu znajduje się komputer zwany serwerem, który „hostuje” stronę
- Może nam wysłać pliki, jeśli ładnie go poprosimy – znajduje się tam program, zdolny do odpowiadania na odpowiednio sformułowane prośby
- Co znaczy „Odpowiednio sformułowane”?



Protokoły

- Komputery komunikują się przy pomocy protokołów – czyli ścisłe ustalonych schematów komunikacji
- Do pobierania stron internetowych służy protokół HTTP – HyperText Transfer Protocol
- Dzisiaj jest już wypierany przez szyfrowany HTTPS (Secure)
- Przeglądarka pozwala nam podejrzeć wysyłane żądania:

HTTPS

[Article](#) [Talk](#)

From Wikipedia, the free encyclopedia

Hypertext Transfer Protocol Secure (HTTPS) is an extension of the Hypertext Transfer Protocol (HTTP). It uses encryption for secure communication over a computer network, and is widely used on the Internet.^{[1][2]} In HTTPS, the communication protocol is encrypted using Transport Layer Security (TLS) or, formerly, Secure Sockets Layer (SSL). The protocol is therefore also referred to as [HTTP over TLS](#),^[3] or [HTTP over SSL](#).

The principal motivations for HTTPS are authentication of the accessed website and protection of the privacy and integrity of the exchanged data while it is in transit. It protects against man-in-the-middle attacks, and the bidirectional block cipher encryption of communications between a client and server protects the communications against eavesdropping and tampering.^{[4][5]} The authentication aspect of HTTPS requires a third party to sign server-side digital certificates. This was historically an expensive operation, which meant fully authenticated HTTPS connections were usually found on secured payment transaction services and other secured corporate information on the World Wide Web. In 2016, a campaign by the Electronic Frontier Foundation supported web browser developers led to the protocol becoming more prevalent, now used more often by web users than the original, non-secure HTTP, primarily for page authenticity on all types of websites, secure accounts, and to keep user communication identity, and web browsing private.

Overview [edit]

Further information: [Transport Layer Security](#)

The Uniform Resource Identifier (URI) scheme **HTTPS** has identical usage syntax to the HTTP scheme. However, HTTPS signals the browser to use an added encryption layer of SSL/TLS to protect the traffic. SSL/TLS is especially suited for HTTP, since it can provide some protection even if only one side of the communication is authenticated. This is the case with HTTP

[Create account](#) [Log in](#) ...

文 A 57 languages ▼

[Read](#) [Edit](#) [View history](#) [Tools](#) ▼

Internet protocol suite
Application layer
BGP • DHCP (v6) • DNS • FTP • HTTP (HTTP/3)
• [HTTPS](#) • IMAP • IRC • LDAP • MGCP • MQTT • NNTP • NTP • OSPF • POP • PTP • ONC/RPC • RTP • RTSP • RIP • SIP • SMTP • SNMP • SSH • Telnet • TLS/SSL • XMPP • [more...](#)

Transport layer
TCP • UDP • DCCP • SCTP • [RSVP](#) • QUIC • [more...](#)

The **Resource Reservation Protocol (RSVP)** is a transport layer protocol designed to reserve resources across a network using the integrated services model. RSVP operates over an IPv4 or IPv6 and provides receiver-initiated setup of resource reservations for multicast or unicast traffic.


HTTP
Persistence • Compression • [HTTPS](#) • QUIC
Request methods
OPTIONS • GET • HEAD • POST • PUT • DELETE
• TRACE • CONNECT • PATCH
Header fields

[Filter URLs](#)

1 + Disable Cache No Throttling ...

All	HTML	CSS	JS	XHR	Fonts	Images	Media	WS	Other		
200	GET	en.wikipedia.org		HTTPS				document	html	51.80 kB	261.24 kB
304	GET	en.wikipedia.org		load.php?lang=en&modules=	stylesheet				css	cached	232.04 kB
304	GET	en.wikipedia.org		load.php?lang=en&modules=	script				js	cached	62.12 kB
200	GET	en.wikipedia.org		wikipedia.png	img				png	cached	16.50 kB
200	GET	en.wikipedia.org		wikipedia-wordmark-en.svg	img				svg	cached	64.46 kB
200	GET	en.wikipedia.org		wikipedia-tagline-en.svg	img				svg	cached	80.46 kB
304	GET	en.wikipedia.org		load.php?lang=en&modules=	stylesheet				css	cached	7.88 kB
200	GET	upload.wikimedia.org		180px-HTTP_logo.svg.png	img				webp	cached	2.16 kB
200	GET	upload.wikimedia.org		220px-Internet2.jpg	img				jpeg	cached	12.40 kB
200	GET	upload.wikimedia.org		226px-Extended_Validation_	img				png	cached	20.59 kB
200	GET	upload.wikimedia.org		300px-HTTPS_on_Firefox_89	img				png	cached	24.69 kB
200	GET	upload.wikimedia.org		253px-Self-signed_certificate	img				png	cached	20.59 kB
200	GET	upload.wikimedia.org		30px-Commons-logo.svg.png	img				webp	cached	1.14 kB
200	GET	upload.wikimedia.org		16px-Symbol_category_class	img				webp	cached	880 B
200	GET	upload.wikimedia.org		12px-Commons-logo.svg.png	img				webp	cached	496 B
200	GET	en.wikipedia.org		load.php?lang=en&modules=	load.php:9 (s...)				js	cached	0 B
200	GET	en.wikipedia.org		load.php?modules=skins.vector	img				svg	cached	44.30 kB
200	GET	en.wikipedia.org		load.php?modules=skins.vector	img				svg	cached	44.51 kB
200	GET	en.wikipedia.org		load.php?modules=skins.vector	img				svg	cached	45.06 kB
200	GET	en.wikipedia.org		load.php?modules=skins.vector	img				svg	cached	45.65 kB
200	GET	en.wikipedia.org		load.php?modules=skins.vector	img				svg	cached	46.38 kB
200	GET	en.wikipedia.org		arrow-down-progressive.svg	img				svg	cached	45.38 kB
200	GET	en.wikipedia.org		arrow-down.svg?f88ee	img				svg	cached	45.38 kB
200	GET	en.wikipedia.org		bullet-icon.svg?d4515	img				svg	cached	45.15 kB
200	GET	en.wikipedia.org		magnify-clip-ltr.svg?78330e	img				svg	cached	46.38 kB
200	GET	en.wikipedia.org		link-external-small-ltr-progr...	img				svg	cached	45.18 kB
200	GET	upload.wikimedia.org		Lock-green.svg	img				svg	cached	47.41 kB
200	GET	upload.wikimedia.org		Icon_pdf_file.png	img				png	cached	352 B
200	GET	en.wikipedia.org		load.php?modules=skins.vector	img				svg	cached	44.30 kB

Jak nasze żądanie trafia do celu?

- Internet to sieć połączonych komputerów – bez centrum
- Komunikacja zachodzi dzięki protokołowi IP (Internet Protocol)
- Każdy komputer posiada unikalny adres IP
- Adres wygląda mniej więcej tak: **192.0.2.1**
- Albo w formacie binarnym: **11000000.00000000.0000010.00000001**
- Można sprawdzić, czy pod danym adresem ktoś mieszka, przy pomocy narzędzia **ping**
- Uwaga: nie wszystkie zajęte adresy zawierają strony!

PING!

Google: 8.8.8.8

```
C:\Users\Bartek>ping 8.8.8.8
```

```
Pinging 8.8.8.8 with 32 bytes of data:  
Reply from 8.8.8.8: bytes=32 time=10ms TTL=118  
Reply from 8.8.8.8: bytes=32 time=9ms TTL=118  
Reply from 8.8.8.8: bytes=32 time=9ms TTL=118
```

Wikipedia: 94.23.242.48

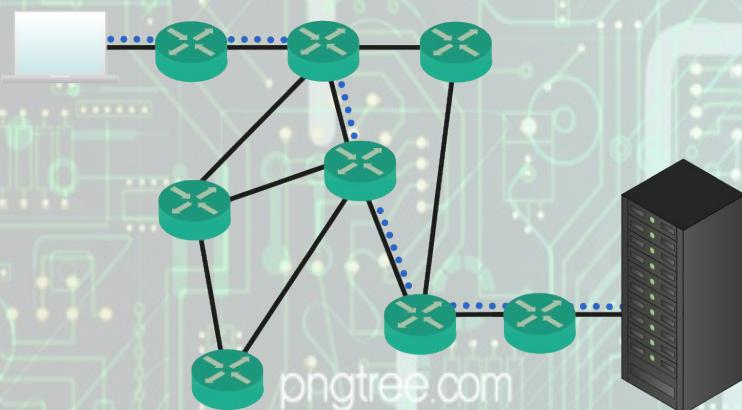
```
Pinging 94.23.242.48 with 32 bytes of data:  
Reply from 94.23.242.48: bytes=32 time=42ms TTL=53  
Reply from 94.23.242.48: bytes=32 time=39ms TTL=53  
Reply from 94.23.242.48: bytes=32 time=43ms TTL=53  
Reply from 94.23.242.48: bytes=32 time=43ms TTL=53
```

Ping statistics for 94.23.242.48:

```
Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),  
Approximate round trip times in milli-seconds:  
Minimum = 39ms, Maximum = 43ms, Average = 41ms
```

Ścieżka w sieci

- Jak wspomniano, w internecie nie ma żadnej centrali
- Wiadomości muszą szukać adresatów, przeskakując na oślep między kolejnymi stacjami
- Niektóre stacje pamiętają, w którą stronę należy iść, aby trafić pod dany adres – jeśli niedawno dostały od niego wiadomość
- Trasę pakietu można prześledzić narzędziem **tracert**



Tracert

Google: 8.8.8.8

```
Tracing route to dns.google [8.8.8.8]
over a maximum of 30 hops:
```

```
1    <1 ms    <1 ms    <1 ms  funbox.home [192.168.1.1]
2      4 ms      3 ms      4 ms  gda-bng4.neo.tpnet.pl [83.1.5.140]
3      3 ms      3 ms      3 ms  gda-r12.tpnet.pl [80.50.151.97]
4     10 ms      7 ms      9 ms  195.116.35.198
5      8 ms      7 ms      7 ms  72.14.214.158
6     13 ms     11 ms     12 ms  216.239.58.135
7     12 ms      8 ms      8 ms  142.250.238.3
8     10 ms      7 ms      7 ms  dns.google [8.8.8.8]
```

Trace complete.

Wikipedia: 94.23.242.48

```
Tracing route to tools.wikimedia.pl [94.23.242.48]
over a maximum of 30 hops:
```

```
1    <1 ms    <1 ms    <1 ms  funbox.home [192.168.1.1]
2      2 ms      4 ms      3 ms  gda-bng4.neo.tpnet.pl [83.1.5.140]
3      4 ms      3 ms      3 ms  gda-r12.tpnet.pl [80.50.151.97]
4      7 ms      7 ms      7 ms  195.116.35.194
5     10 ms     36 ms     14 ms  be103.waw-wa1-pb1-nc5.pl.eu [213.186.32.176]
6      *        *        *        Request timed out.
7      *        *        *        Request timed out.
8      *        *        *        Request timed out.
9     30 ms     31 ms     31 ms  fra-fr5-sbb2-nc5.de.eu [54.36.50.116]
10    44 ms     42 ms    325 ms  be103.rbx-g2-nc5.fr.eu [94.23.122.240]
11    *        *        *        Request timed out.
12    *        *        *        Request timed out.
13    *        *        *        Request timed out.
14    42 ms     39 ms     39 ms  tools.wikimedia.pl [94.23.242.48]
```

Jak z google.com robi się 8.8.8.8?

- Komputery operują na adresach IP – ale dla ludzi to niewygodne
- Aby ułatwić nam życie, powstała usługa DNS – Domain Name Service
- Serwer DNS możemy zapytać o dowolny adres www – a on odeśle nam odpowiedni adres IP
- Dzięki temu wystarczy, że znamy jeden adres IP (w praktyce zna go przeglądarka) – np. 8.8.8.8
- Większość usług internetowych automatycznie odpytuje DNS, jeśli podany zostanie adres www

DNS

Google Public DNS

DNS Name
www.wp.pl

Resolve

RR Type

A

EDNS Client Subnet

Disable DNSSEC validation



Show DNSSEC detail



Result for www.wp.pl/A with DNSSEC validation and without DNSSEC detail:

```
{  
    "Status": 0 /* NOERROR */,  
    "TC": false,  
    "RD": true,  
    "RA": true,  
    "AD": false,  
    "CD": false,  
    "Question": [  
        {  
            "name": "www.wp.pl.",  
            "type": 1 /* A */  
        }  
    ],  
    "Answer": [  
        {  
            "name": "www.wp.pl.",  
            "type": 1 /* A */,  
            "TTL": 35,  
            "data": "212.77.98.9"  
        }  
    ]  
}
```

You may also resolve directly at: <https://dns.google/resolve?name=www.wp.pl&type=A>

Niestety z większością z tych adresów nie połączymy się wpisując je w adres przeglądarki - jeśli serwer ma kilka stron, to bez pełnego adresu nie wie, co ma nam zaserwować

To by było na tyle

Pytania?

Dziękuję za uwagę !



Źródła

- [https://en.wikipedia.org/wiki/Computer_\(occupation\)](https://en.wikipedia.org/wiki/Computer_(occupation))
- <https://pl.wikipedia.org/wiki/Tranzystor>
- https://pl.wikipedia.org/wiki/Maszyna_analityczna
- https://pl.wikipedia.org/wiki/Ada_Lovelace
- https://pl.wikipedia.org/wiki/Alan_Turing
- https://pl.wikipedia.org/wiki/Lampa_elektronowa
- <https://pl.wikipedia.org/wiki/Przeka%C5%BAnik>
- https://pl.wikipedia.org/wiki/Lampa_elektronowa
- <https://www.python.org/psf-landing/>
- https://www.w3schools.com/html/html_examples.asp
- <http://tutorialsprogram.blogspot.com/2014/05/introduction-of-html.html>
- <https://www.pexels.com/photo/painting-brush-on-palette-1108532/>
- <https://en.wikipedia.org/wiki/HTTP>
- https://en.wikipedia.org/wiki/IP_address