It's all about information sections/1/its-all-about-information

Simplification through Standartisation Scalability So, we now know, how to use *Hartley function* to speculate abou
But do we really need to calculate how many bits of data will it occupy to do something and create own encodings
This caveat, regardless of the activity type, is solved by pretty much the only practical way possible — standartisat
Most of the products and services in IT segment are way more scalable than fields requiring physical manipulation
It wasn't always like this, though. Not so long ago, computers weren't as widespread as they are today. Back in a d
Nomenclature and text encodings
So, back in a day, when computers were ineffecient, programs *had to* be effecient to somehow compensate for hardw
Text encodings are one of the most fundamental standards there is in programming world. One of the first *text enc*

Eventually a nomenclature was formed:

Nomenclature of bit capacity
Perception differences
Since we know, that space being occupied on computers directly tied to a total number of variants, we can now em
9 + 13 = 22
Cats are believed to have 9 lives
It just so happens, I have an extra 9.5$
9.8 + 0.2 = 10
In each of these cases, we can encounter '9' in some form. If we consider all previous examples to be an ASCII text
Examples like these show this difference, between a human mind and a way of how computers work. See, we are so
For computers to compute (pun intended) we must firstly to *make up a finite set of possible variants*. This is usual

But wait — we cannot create negative numbers in this example! Should we use structure just like this for negative

It's all good and all, but these examples show only *integer* numbers. What if we have some number like 9.23? We c
We can make up a type, that works with *fractions*[14]. For example, let's create a type taking 2 bytes. In this case w
Let's take a number, for example — 1024.
$1024 = 1 * 10^3 + 0 * 10^2 + 2 * 10^1 + 4 * 10^0$. Mind that, this method works not only for $base_{10}$ numbers, but for any

e.g.
$110101011_2 = 1 * 2^8 + 1 * 2^7 + 0 * 2^6 + 1 * 2^5 + 0 * 2^4 + 1 * 2^3 + 0 * 2^2 + 1 * 2^1 + 1 * 2^0 = 427_{10}$

$1054_8 = 1 * 8^3 + 0 * 8^2 + 5 * 8^1 + 4 * 8^0 = 556_{10}$

$1AC3_{16} = 1 * 16^3 + 10_{10} * 16^2 + 12_{10} * 16^1 + 3 * 16^0 = 6851_{10}$
So, you might wonder: *How can it help us with fractions*? And I can answer: we present fraction part as some pow
$100.75_{10} = \{$ 0 $1100100_2 - - - integerpart$
$11000000_2 - - - fractionpart$
Notice, we can trim all leading zeros in integer part *and all trailing zeros in the* fraction part *without affecting a v*
We already know, how we got an *integer part* of our number. But how on Earth did we calculate *fraction part*?
To calculate a fraction part in binary, we should perform following operations:
Trim integer part from our number, *we should use only fraction part.*
Multiply fraction part by 2. Store resulting *integer part* somewhere.
Repeat[16] steps 1–2, passing result from the second step to the first step.
List of *integer* parts is your binary number.
e.g.
|c|c|c|c|

Step #	Number	Result	Result's integer part
1	0.75 -> 0.75	0.75 * 2 = 1.5	1
2	1.5 -> 0.5	0.5 * 2 = 1.0	1

So, 11_2 is our result for *fraction part.*

Countess, weaving patterns and count machines
Ancient calculators
So, essentially, what is a computer? And yes, there is a definition from Merriam-Webster dictionary in the first pag
I want to devote this section to a brief historical overview of computers. Please, keep in mind, that it is a *vast* topi
But where should we start? Well, since computer was oversimplied to rather big calculator, I suggest we start with

Abaci in different cultures

However, those are purely mechanical and manual devices, main principle of which is very similar to how we conver
lh0.3 [scale=0.1]images/persons/person$_blaise_pascal.jpgBlaisePascal$
The first succesful automation attempt is attributed to Blaise Pascal, with his *Arithmetic Machine* which is also ca
There are several pascalines still intact nowadays, most of the remaining ones are in european museums. Being the
Pascaline wasn't a computer, but it was first in many ways — first calculator, which was afterwards commercialized
Luddites' nightmare
r0.3 [scale=0.2]images/persons/person$_joseph_jacquard.jpgJosephMarieJacquard$
The next machine of our interest is not a computer either. It isn't even a calculator — it's a loom. I cannot say mu
Beginning of the 19^{th} century is pretty much a middle of industrial revolution. New fancy industrial looms are prac
r0.5 [scale=0.2]images/devices/device$_jacquard_loom.jpgJacquardLoom$
So, amidst all this revolution going, we could see how a work, that was being done by 100 men before can be done
The main purpose of Jacquard's attachment was an *pattern weaving automation*. It used a chain of special cards la
l0.5 [scale=0.2]images/misc/luddites.jpg 1844's depiction of Luddites destroying the loom
This machine would *drastically* impact effeciency, since it was no longer required to be of high skill to weave compli
Steam, calculators and British Government l0.3 [scale=0.2]images/persons/person$_charles_babbage.pngCharlesBabba$
Well, the idea of using automation in weaving patterns have touched deeply not only the Luddites, but also at least
Charles Babbage was an inventor of 2 machines, that are of interest in this essay: *Difference Engine* and *Analytical*
r0.5 [scale=0.25]images/devices/device$_babbage_difference_engine.jpgPartofBabbage'sDifferenceEngine$
The Difference Engine, essentially was a giant mechanical calculator, that was powered by steam and printed result
One copy of this letter did reach a Lord of Treasury, who referred it to the Royal Society. After receiveing an endor
In 1833 Charles Babbage threw a party, where he demostrated his guests, mostly members of high society, a part th

Photo by Karoly Lorentey, sourced from wikimedia commons, under Creative Commons Attribution 2.0 Generic license
As was mentioned before, Difference Engine project did end up exceeding given funding. Charles Babbage wasn't a
Analytical Engine was designed to consist of four major elements: the mill, the store, the reader and the printer. F
So, that's where an inspiration from Joseph Jacquard really kicks in! The principle behind punch cards, used in Jac
So, operating with those cards would give to one an ability to code necessary instructions for Analytical Engine to
Enchantress of Numbers and Italy's Prime-Minister
Using punch cards as a format of input data not only have fascinated Babbage, but Ada Byrone too. For the next
Babbage gave lectures about his inventions sometimes. On one of such occasions he had a very special listener — *L*
Ada decided to translate Menabrea's work in English, titled 'Sketch of the Analytical Engine invented by Charles B
She also illustrated in her notes a sequential solution of various problems, through input in a form of punch cards i

Countess of Lovelace
Although there are some disputes regarding the title of the 'first programmer ever'[42], there is one thing virtually no
However, the history of Analytical Engine have ended due to a lack of funding, and it remained mostly on paper fo
Charles Babbage continued to work on his machine until his death in 1871. As was said, machine was never finishe
Brief timeline of the computer history
Computer history is filled with many significant occasions since 19^{th} century. As was said, it is indeed a vast topic,
p0.2p0.8 Year Details
1941 Konrad Zuse finishes first programmable, electric digital computer, called Z3. Basically, all the Babbage wanted,
1945 John von Neumann described an architecture, that is the basis for virtually all of the computers we have today. T
1948 Claude Shannon formulated his first thoughts regarding what will be regarded as 'Information Theory' in the futu
(r)2-2 First computer program ran on the computer
1952 Grace Hopper invented first high-level programming language, A-0. It will evolve into COBOL later.
1956 Keyboard was succesfully connected to computer. Prior to this point, all programming had to be done by punch c
1957 FORTRAN created, *first widely used high-level language.*
1958 LISP developed
1960 COBOL developed. COBOL *still* highly in use today, especially in financial sector. Up to 80% of the world's dail

(r)2-2 ALGOL-60 developed
1969 ARPANET first online. ARPANET is a direct predecessor for the *Internet* we know today
(r)2-2 Kenneth Thompson and Dennis Ritchie developed UNIX. It's not too much of a stretch to say, that it is one of t
1970 First ATM can be used by general public.
(r)2-2 Pascal programming language developed.
1972 C language developed. C is one of the *most influential* programming language there is. It's average popularity, by
1973 First handheld cellular mobile phone invented. Start of a mobile network we know today.
(r)2-2 First successful inter-network communications. Birth of the Internet as we know it today.
1977 Apple II computer is developed by Steve Wozniak and marketed by Steve Jobs. One of the first computers, intend
(r)2-2 Atari video game console released. One of the first game consoles in history
1978 First Multi-User Domain games appeared. They allowed for multiple players play against each other. One of the
(r)2-2 First Computer Worm created.
1983 Internet officially launched
(r)2-2 Microsoft Word officially launched
1985 C++ released
1987 Basic parameters for GSM standard agreed. GSM is the main standard used in mobile network today.
(r)2-2 Perl developed
1990 First commercially succesful Windows OS version released, Windows 3.0
(r)2-2 Photoshop initially released
1991 Linus Torvalds released Linux kernel
(r)2-2 Charles Babbage's Difference Engine #2 is constructed at London Museum.
1993 First online ads appeared
1995 Java 1.0 introduced
(r)2-2 Javascript released
1999 WiFi routers started to gain popularity for in-home usage.

Modern computers Hardware every computer needs

This section is mostly devoted to computers as we know them today. We are surrounded by computers. Our PCs[56]

Well, first of all — they all have different *hardware*. Hardware — is all of the computer's tangible parts or compon

Hardware is the basis of any computer. Once you got limited by hardware, there is not much you can do without c

So, *hardware often acts as a physical limit of what your computer can do.* If you absolutely need to save 100 GB of

You cannot change hardware without physically installing/reinstalling some computer component. It cannot be don

So, what hardware every computer needs? Well, maybe not every little one of them, but the majority? What will b

CPU[58]

Central component, 'brain' of the computer, doing all the actual computing.

PSU[59]

Anything that provides whole system with electric power. Batteries, accumulators and such.

RAM[60]

The type of memory to hold partial results of something. It's something like a bank's cash register, where all today's ca

Persistent Memory storage

Acts as a 'long-term' memory storage. Similar to a vault in the bank — it's not convenient to get your hand in it every

GPU[61]

Special type of processors with a specific purpose of showing something on display. Without it you can use your comput

Motherboard

Component that acts as a medium for all other components to communicate

And finally, it all should be mounted and secured in some case. Usually end-user interacts and associate hardware with

The list of possible hardware some computer may utilize in some form is just enormous. It makes no sense to try a

It's worth noting, that each and every one piece of hardware described here is a complex technological device. To u

CPU

As was said before — CPU is a 'brain' of the computer. In the end, all operations, that somehow process any data

So... that's it? To learn programming — means simply to learn those machine code instructions and find some way

Memory GPU Motherboard and PSU