CS101C: Computer Programming
(Using C)
Autumn 2025

Nikhil Hegde
Achyut Mani Tripathi

Week1: Logistics, Introduction, Basic C Program

CS101C: Computer Programming

- Description:

This course provides an introduction to problem solving with computers using ‘C’ as a programming language.

- Credit structure (L-T-P-C) : 3-0-3-9

3 contact hours (three 50min lectures) per week. 6 credits.

3 lab hours (150 mins in total) per week. 3 credits.

Full-semester (14 week + 2 exam week) core course.

- Prerequisites: None

CS101C: Computer Programming

- Assessment Plan - Theory
  - Two paper based exams: Midsem and Endsem
  - Weightage: Midsem = 40 points, Endsem=60 points

- Assessment Plan - Lab
  - Lab 1 - Practice
  - Labs 2-3 = 6 points each (12 points in total)
  - Labs 4-14 = 8 points each (88 points in total)
  - Venue: CIF 401 and CIF 402
- Grades

If your numerical score is at least:	Your course grade will be at least:
90	AA
80	AB
70	BB
60	BC
50	CC
45	CD
40	DD

CS101C: Computer Programming

- Teaching assistants and their role
  - Bonthu Vyuhita, Kumud Singh, Yogesh Kumar, Mridul Chandravamshi. Additionally: 22 TAs for labs.

  - Outside the class, the TAs are your first point of contact regarding doubts. You can **write an email** or **post in the discussion forum** (TBD). If writing an email:
    - Mention the TA email ID in 'to'.
    - Mention the instructor(s) (Prof. Achyut and/or myself) email ID in CC.
    - Mention "CS101 doubt" in the subject line
    - Mention the issue in the body.
    - Do not worry about grammar, etc.
    - DO NOT write an email only to instructors unless otherwise required (considering a large class, most likely the email will be missed)

Course Takeaways

- Write code (essential in creating a piece of software)
- Get to know *one* of the programming languages
  - An old language and still widely used if you want your software to ‘perform’ best
- Get to know features common to any programming language

CS101C: Computer Programming

- Developer essentials
  - Editors, Integrated Development Environment (IDE), Unix Shell, Library-based development, Compiler toolchain
- Programming in C
  - Machine representation, data types and control flow, operators, arrays and strings, functions and recursion, pointers and structures, Input and output using files
- Applications: Sample problems in engineering, science, text processing, and numerical methods.

CS101C: Computer Programming

- References and Texts:

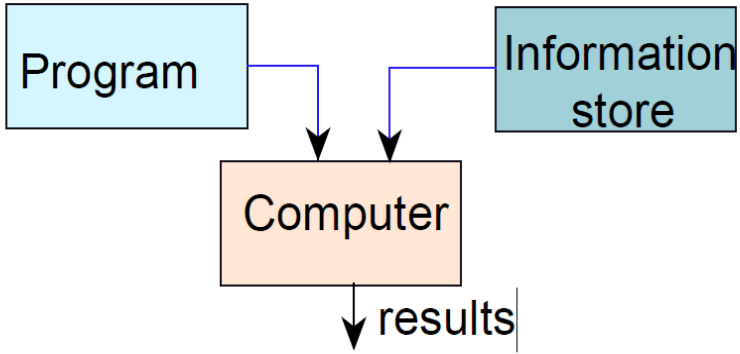
1. **The C Programming Language,** Brian W Kernighan, Dennis M Ritchie, Prentice Hall India , 2nd edition, 1988
2. **Programming with C (Second Edition)** Byron Gottfried, Schaum's Outlines Series, Tata-Mcgraw Hill, 2011
3. **How to Solve It by Computer,** by G. Dromey, Prentice- Hall, Inc., Upper Saddle River, NJ, 1982.
4. **How to Solve _It (2nd ed.),** by Polya, G., Doubleday and co, 1957.
5. **Let Us C**, by Yashwant Kanetkar, Allied Publishers, 1998.
6. **Programming in ANSI C,** by E. Balaguruswamy

there are a number of copies of 1, 5, and 6 in the library.
Class slides and notes (if any) will be posted at:

https://hegden.github.io/cs101

Based on the same principles

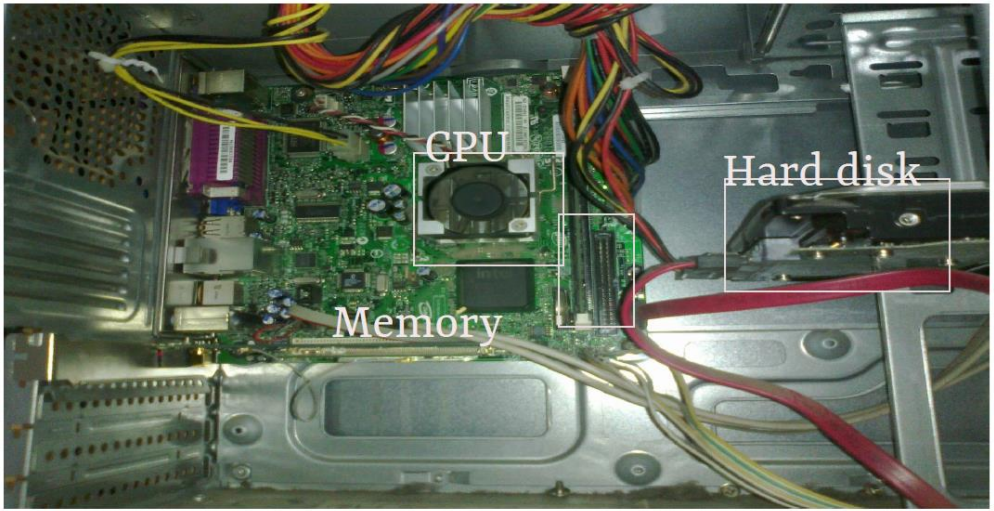
How does it work ?

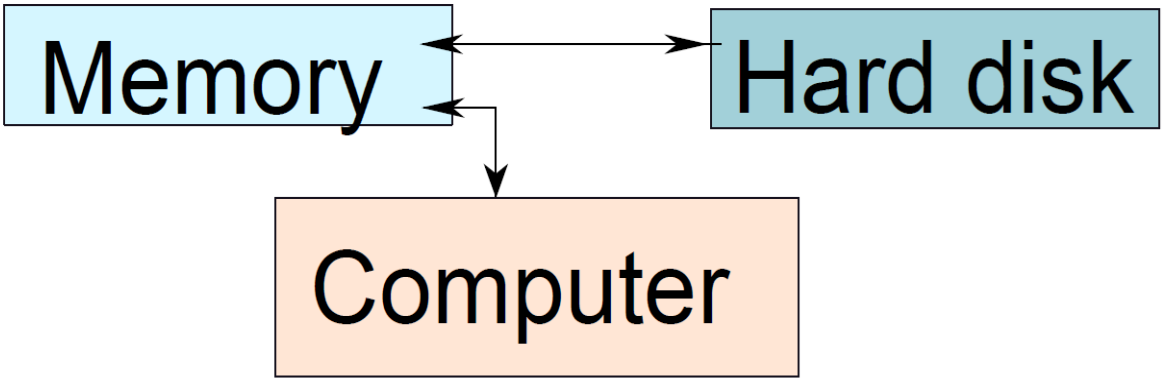


* Program – List of instructions given to the computer
* Information store – data, images, files, videos
* Computer – Process the information store according to the instructions in the program

What does a computer look like ?

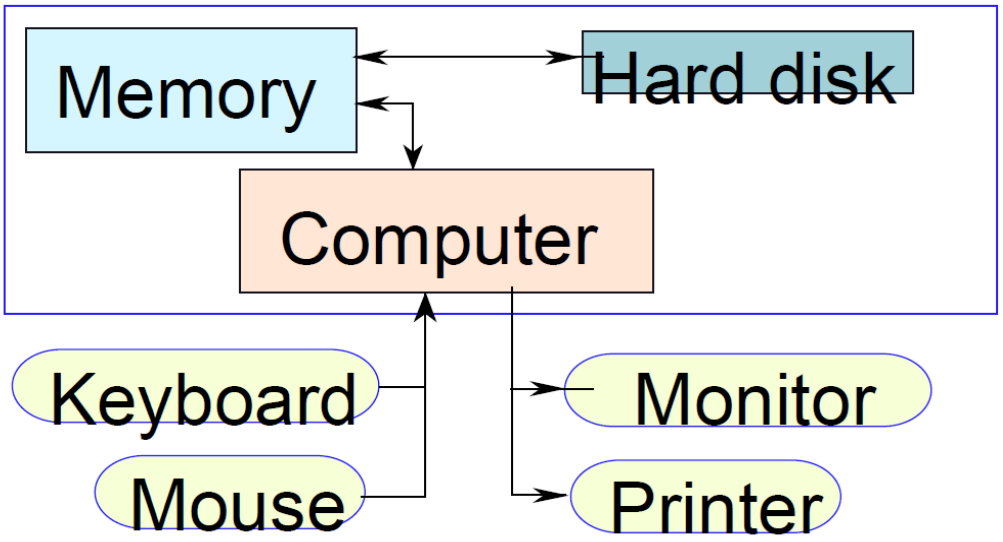
* Let us take the lid off a desktop computer



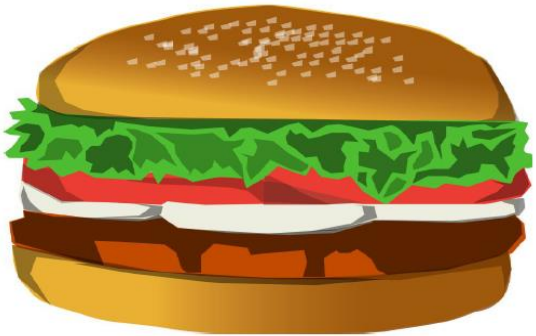


* Memory – Stores programs and data. Gets destroyed when the computer is powered off
* Hard disk – stores programs/data permanently

Let us make it a full system ...

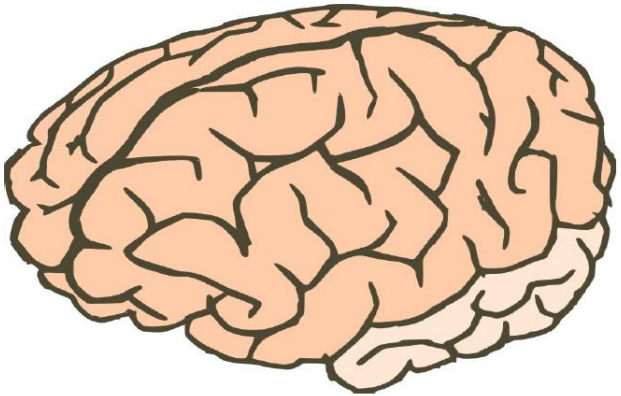


Food for Thought...



* What is the most intelligent computer ?

Answer …

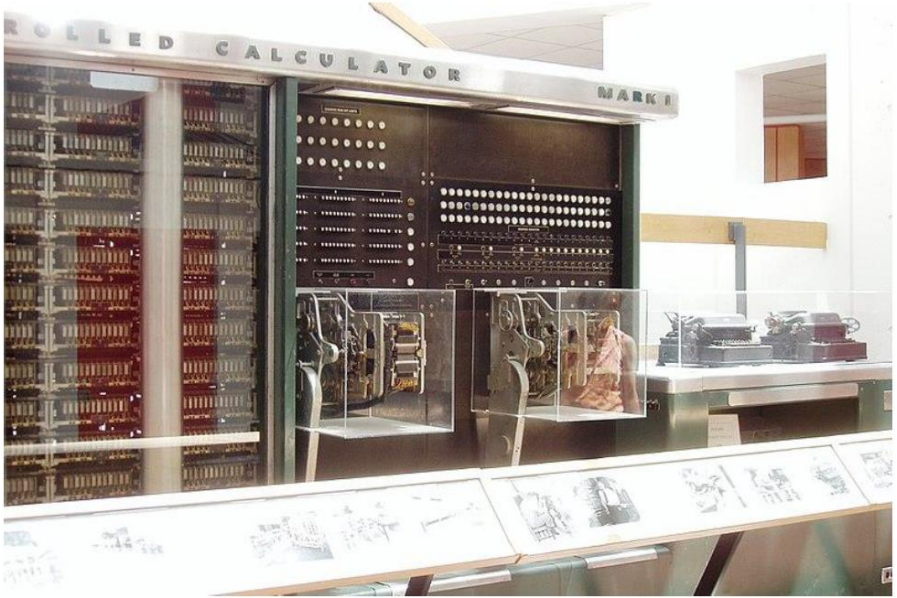


* Our brilliant brains

How does an Electronic Computer Differ from our Brain ?

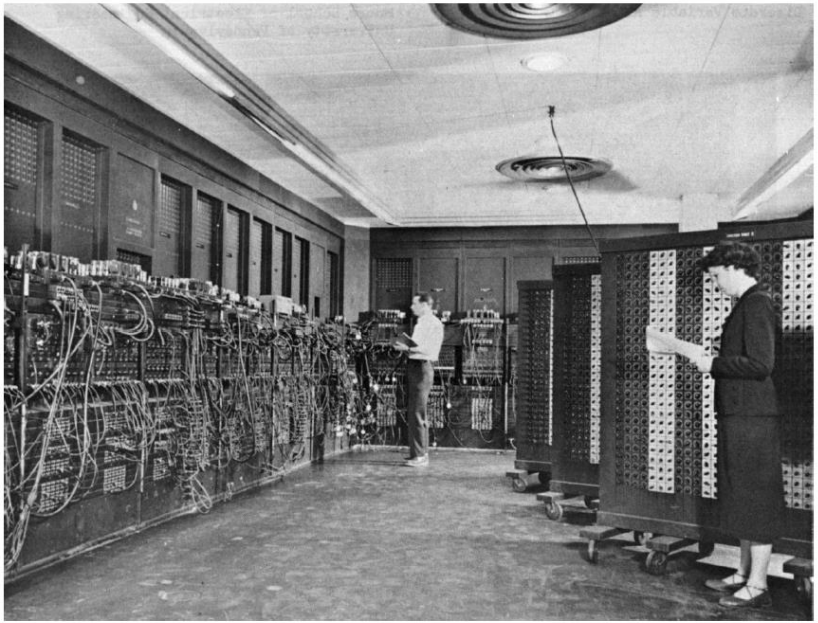
Feature	Computer	Our Brilliant Brain
Intelligence	Dumb	Intelligent
Speed of basic calculations	Ultra-fast	Slow
Can get tired	Never	After sometime
Can get bored	Never	Almost always

* Computers are ultra-fast and ultra-dumb



The Harvard Mark I
(Harvard University, 1944)

[image from wikipedia]



The ENIAC
(University of Pennsylvania 1946)

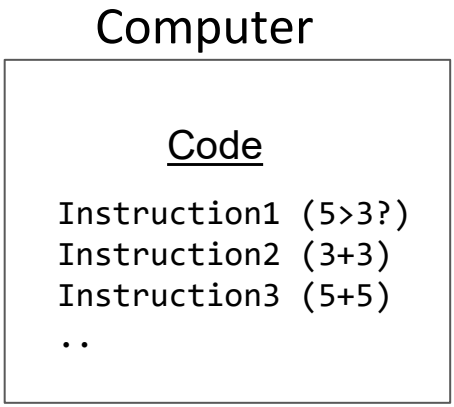
[image from wikipedia]

Next: First C program

Before Coding… Patience

- We start with something simple (e.g. is a > b?). Eventually build something more powerful / usable
- When you structure the code for the computer, the errors are pretty common. These errors are called ***syntax errors***. *Happen all the time*.
- Even professional programmers make syntax errors!
- Try to take a *bunch* of syntax errors and fix them. Repeat. This is a *small, normal* step.

Code: sequence of instructions (operating on data)



- Let us begin with an instruction that prints on the screen

printf: instruction to print

- This is a line of code that calls the printf *function* to print on the screen: `printf("Hello World");`
- printf is the **function**, a verb representing an action that the computer takes
- "Hello World" is the **data** on which the function acts (note: specified within the parenthesis). This data is also called **argument** of the function

Programming Language	English Language
Function	Verb
Function's Argument	Object

printf: note on strings

- Note that the argument to printf is enclosed in quotes “ ”

```
printf(“Hello World”);
```

- Strings are sequence of characters
- Strings are used to store text such as names, urls, paragraphs etc.

More on strings later

- Next: more useful stuff - moving from a line of code to multiple lines