CS101C: Introduction to Programming (Using C)

Autumn 2025

Nikhil Hegde
Achyut Mani Tripathi

Week1: Logistics, Introduction, Basic C Program

Logistics: https://docs.google.com/presentation/d/1O3aAKDcRC4niZY-V_NsRkPaKeVnDTY_Z/edit?usp=sharing&ouid=115667276014179442180&rtpof=true&sd=true

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.

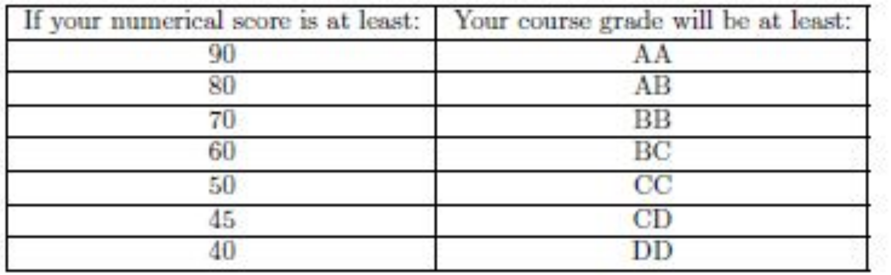
Full-semester (14 week + 2 exam week) core course.

- 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: CI
- 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

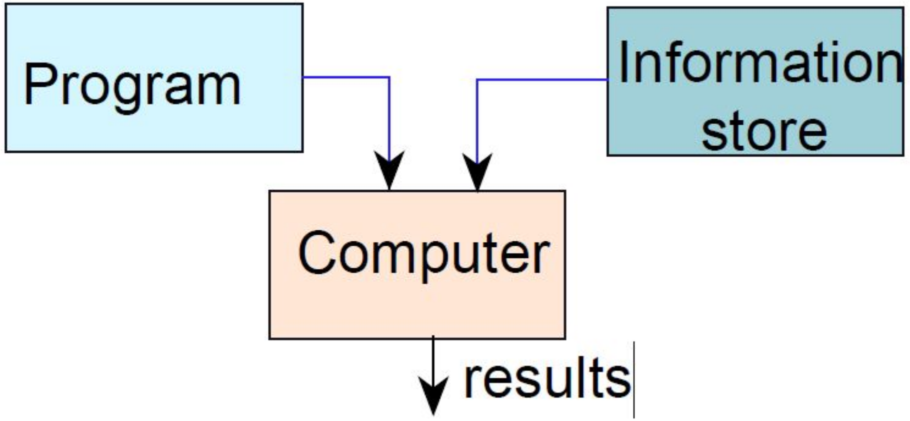
- References and Texts:

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

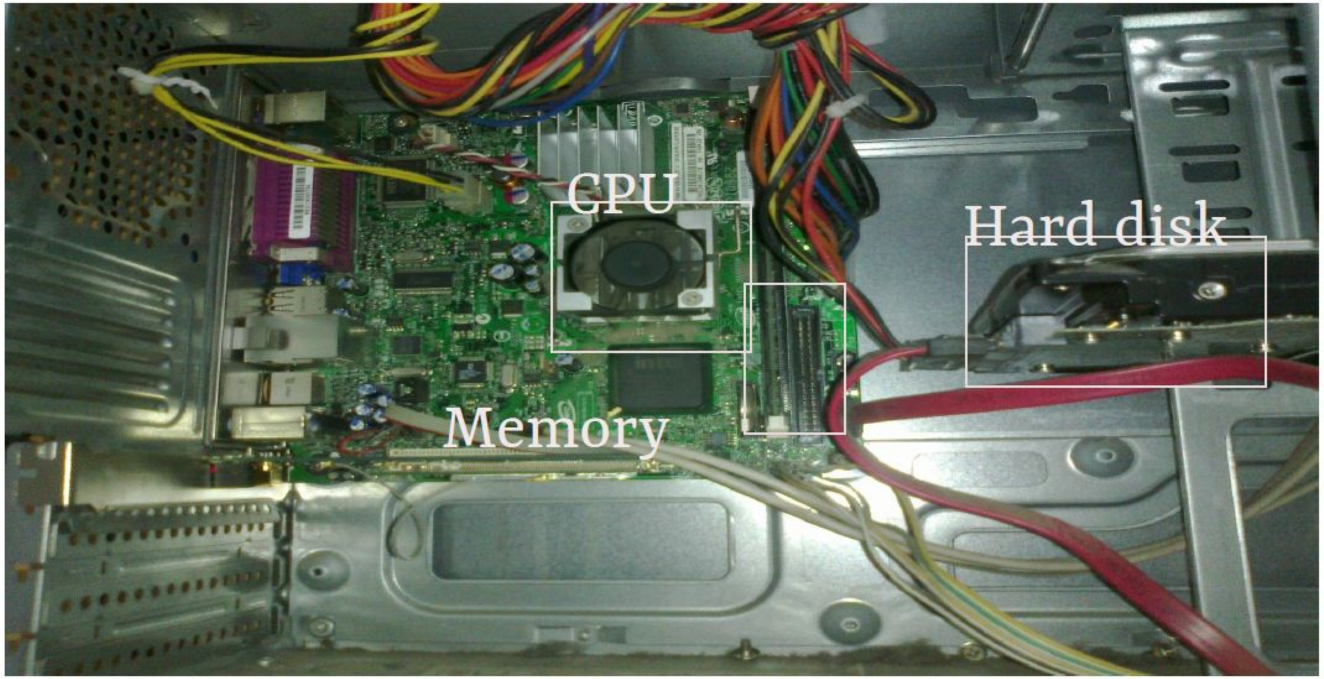
How does it work ?

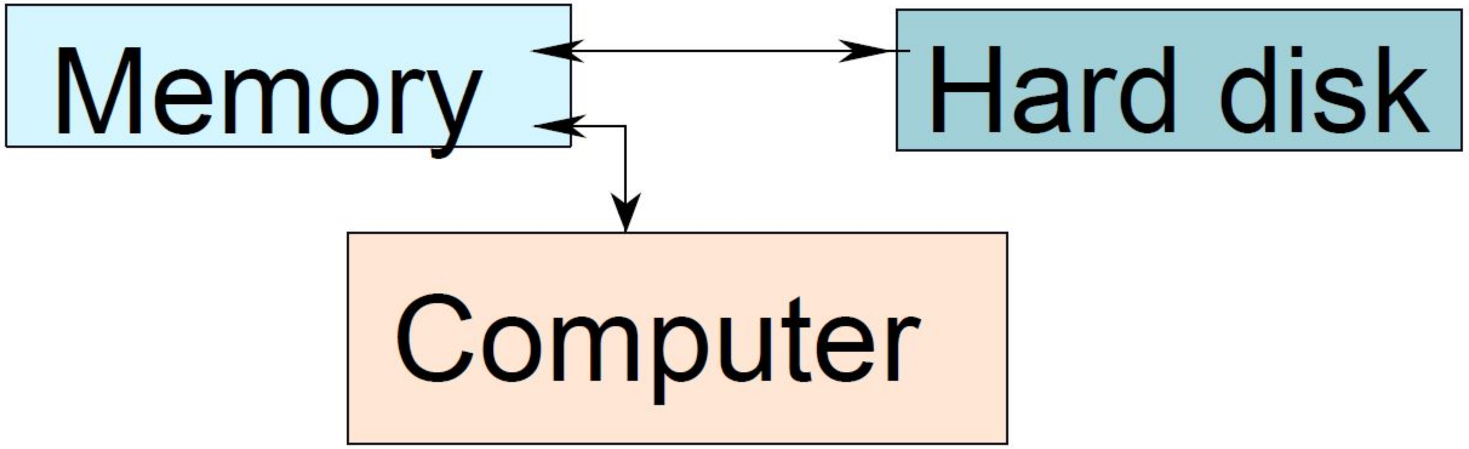


* Program – List of instructions given to the computer
* Information store – data, images, files, videos
* Computer – Process the information store according to the instructions in the program

What does a computer look like ?

* Let us take the lid off a desktop computer

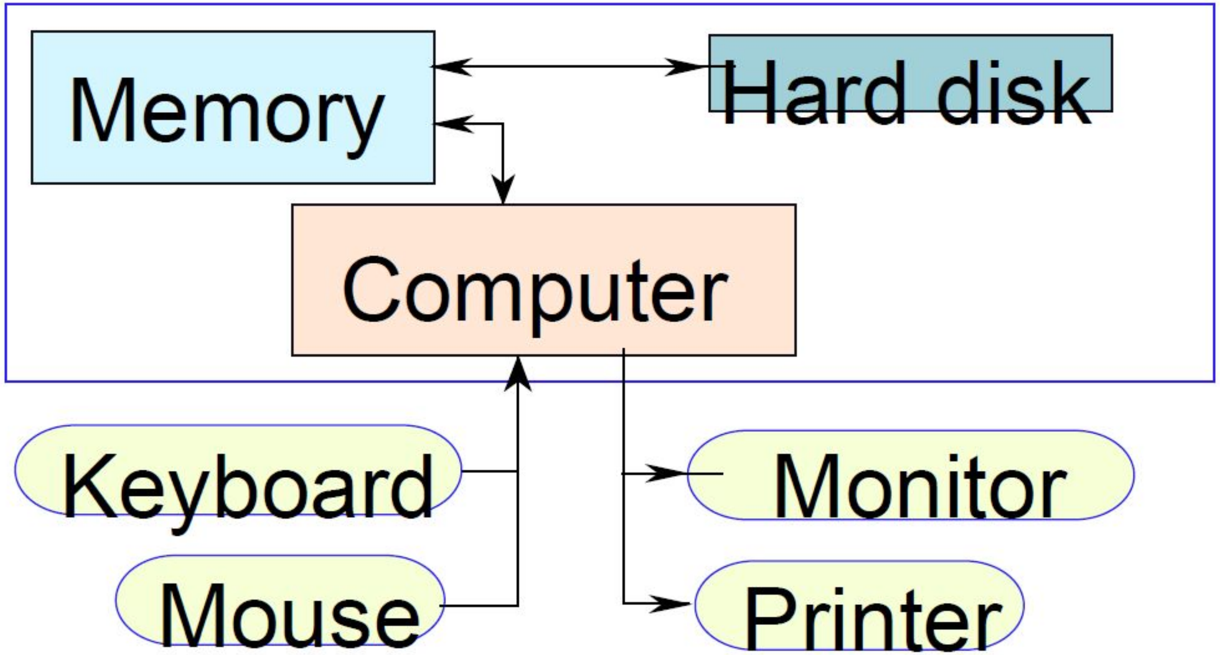




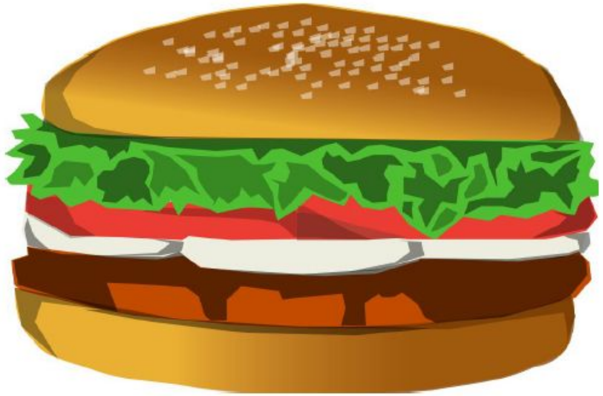
* Memory – Stores programs and data. Gets destroyed when the computer is powered off

* Hard disk – stores programs/data permanently

Let us make it a full system ...

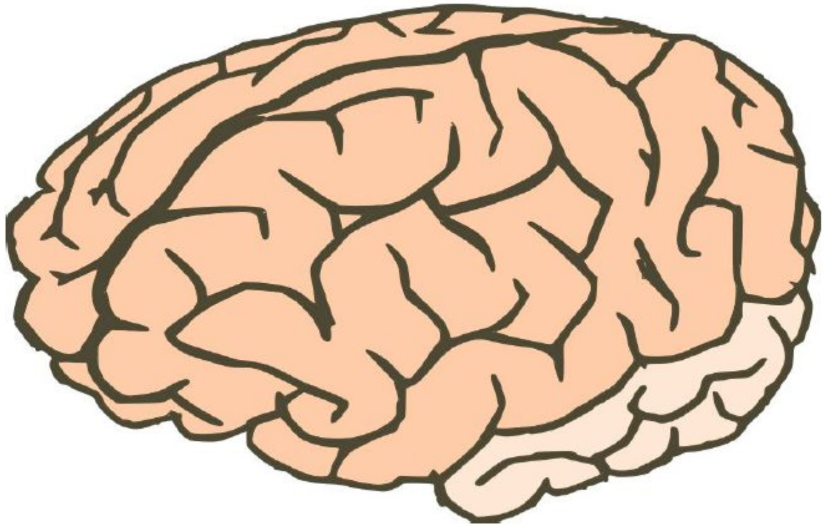


Food for Thought...



* What is the most intelligent computer ?

Answer …

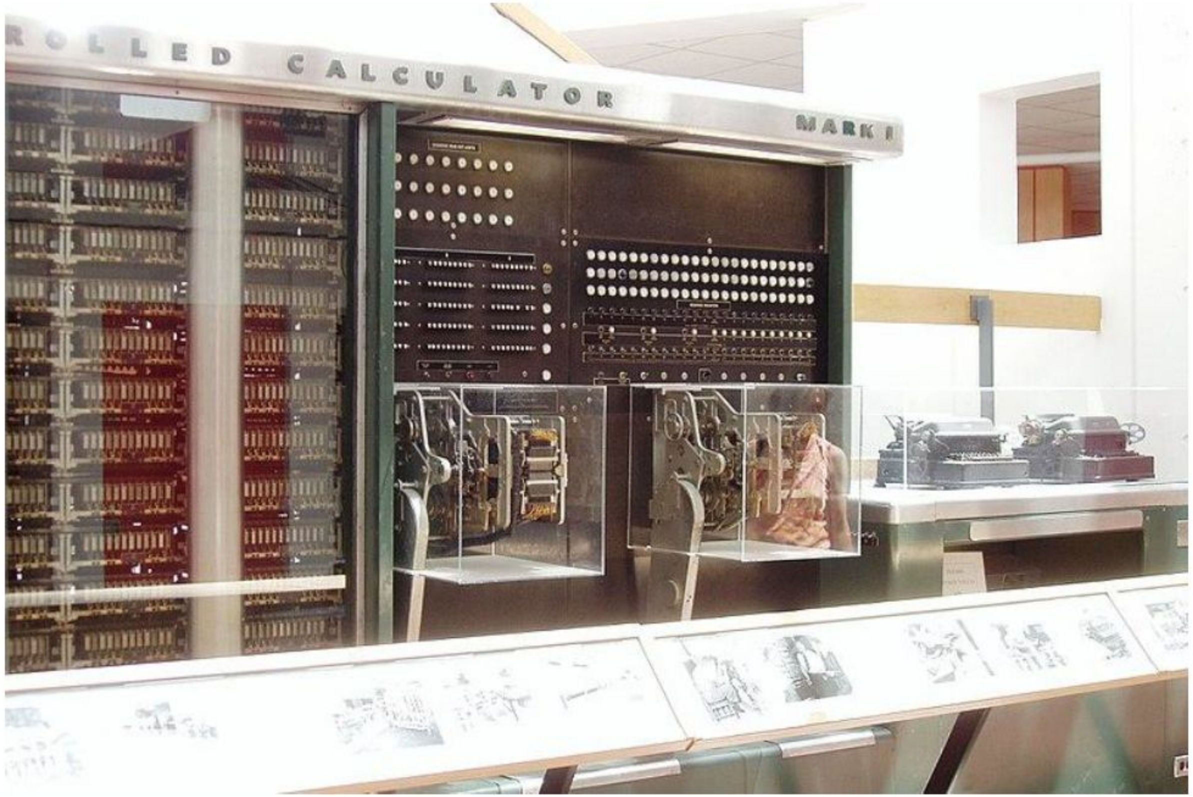


* Our brilliant brains

How does an Electronic Computer Differ from our Brain ?

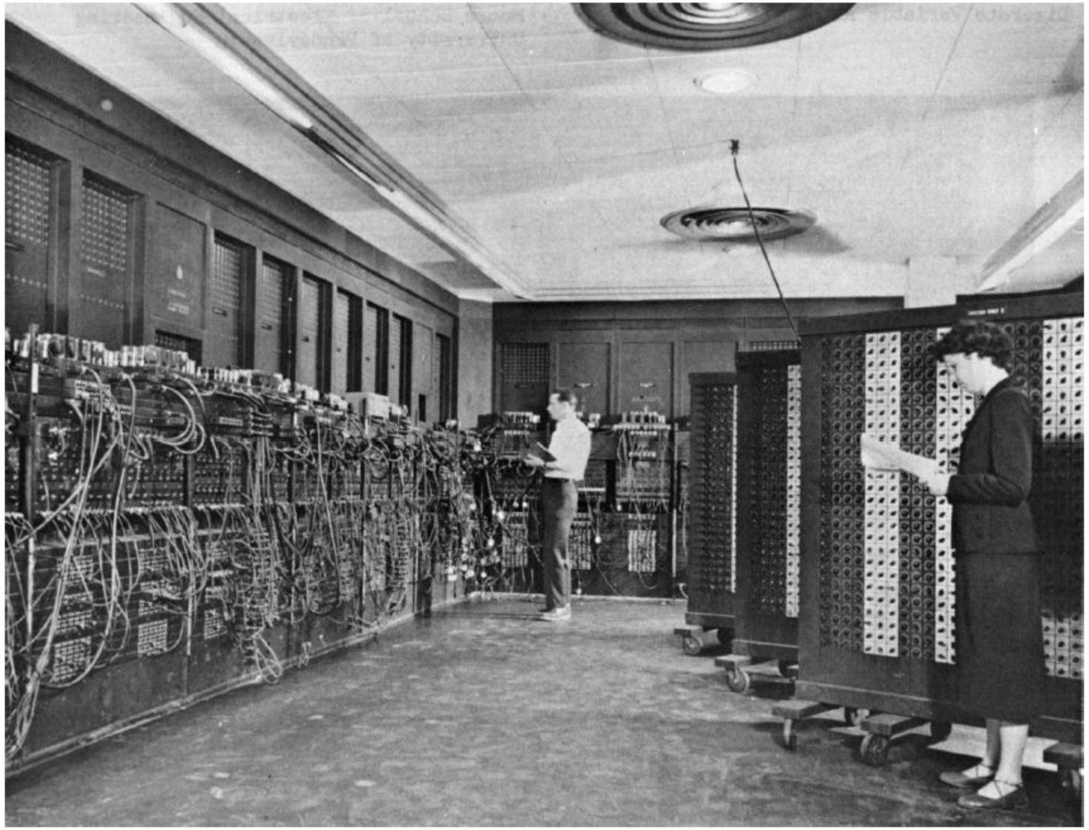
Feature	Computer	Our Brilliant Brain
Intelligence	Dumb	Intelligent
Speed of basic calculations	Ultra-fast	Slow
Can get tired	Never	After sometime
Can get bored	Never	Almost always

* 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)

Computer

Code

```
Instruction1 (5>3?)
Instruction2 (3+3)
Instruction3 (5+5)
..
```

- 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.

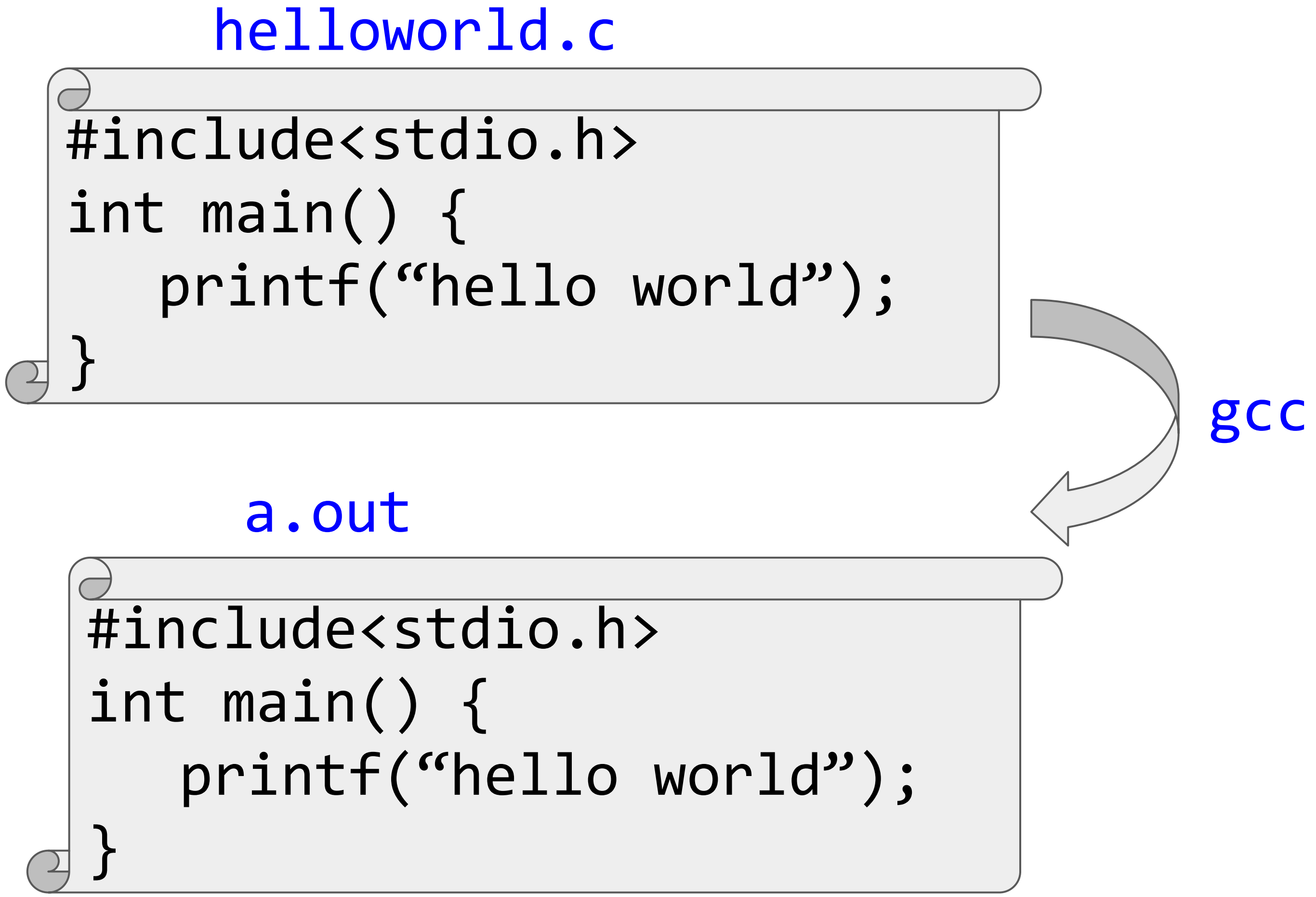
More on strings later

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

Today’s class (6/8/2025)

- Prerequisite software to write a C program
- C program to add two integers
- Concepts: variables, data types, operators, printing integers on terminal, control flow, and data flow

Recap: Hello World, a complete C program



```
#include<stdio.h>
int main() {
   printf("hello world");
}
```

```
#include<stdio.h>
int main() {
   printf("hello world");
}
```

Recap: Hello World, a complete C program

helloworld.c

```
#include<stdio.h>
int main() {
   printf(“hello world”);
}
```

An editor is used to create helloworld.c

a.out

```
#include<stdio.h>
int main() {
   printf(“hello world”);
}
```

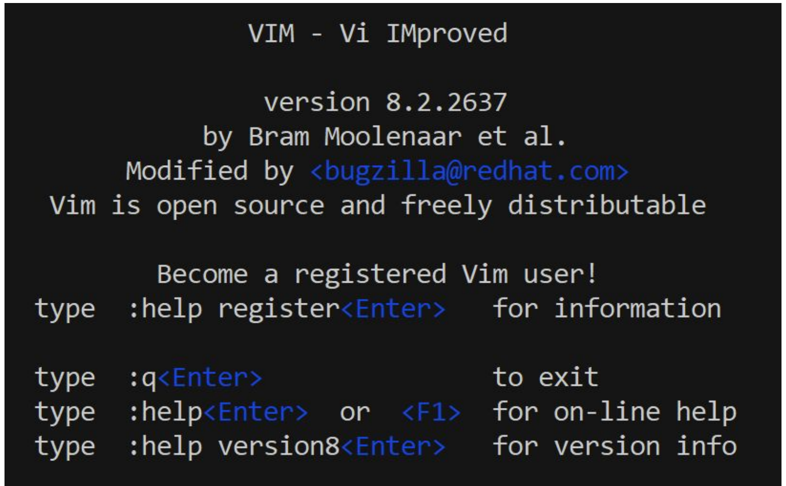
A compiler is used to create a.out

Recap: Hello World, a complete C program

helloworld.c

```
#include<stdio.h>
int main() {
   printf(“hello worl
}
```

Example editors: VIM, VSCode, Gedit, emacs etc.

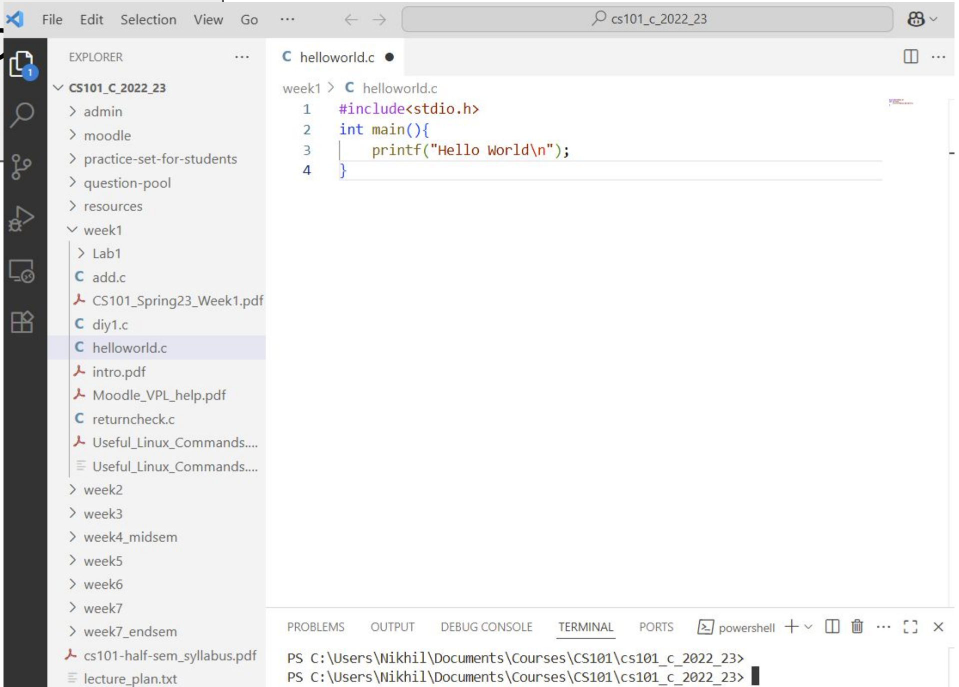


```
                 VIM - Vi IMproved

                 version 8.2.2637
              by Bram Moolenaar et al.
        Modified by <bugzilla@redhat.com>
  Vim is open source and freely distributable

           Become a registered Vim user!
type  :help register<Enter>   for information

type  :q<Enter>               to exit
type  :help<Enter>  or  <F1>  for on-line help
type  :help version8<Enter>   for version info
```



```
#include<stdio.h>
int main(){
    printf("Hello World\n");
}
```

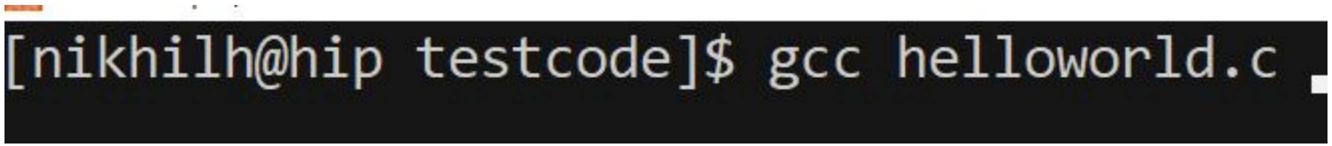
```
PS C:\Users\Nikhil\Documents\Courses\CS101\cs101_c_2022_23>
PS C:\Users\Nikhil\Documents\Courses\CS101\cs101_c_2022_23>
```

Recap: Hello World, a complete C program

a.out

Example compilers: gcc, clang

```
#include<stdio.h>
int main() {
   printf(“hello world”);
}
```



```
[nikhilh@hip testcode]$ gcc helloworld.c
```

Prerequisite Software

Any one of:

- VS Code and plugins
- Putty (if using a Windows system and using a remote machine to execute a C program)
- WSL (Windows Subsystem for Linux). Then install gcc.

`=` A note about the assignment operator

- In mathematics `=` is the equality operator (e.g. `x+1=3` implies value of `x=2`)

- In programming `=` is the assignment operator (e.g. `a=b` implies copy the data in variable `b`'s memory and store that data in variable `a`'s memory)

More about different operators later this week.

Data Types

- What is a data type?
- Way of indicating what a variable is.
- Example:

```
int x;
```

1. What is the set of values this variable can take on?
2. How much space does this variable take up?
3. How should operations on this variable be handled?

Data Types

- `int x;`

What is the set of values this variable can take on in C?

-2^31 to (2^31 – 1)

2. How much space does this variable take up?

32 bits

3. How should operations on this variable be handled? integer division is different from floating point divisions

3 / 2 = 1 =>integer division 3.0 / 2.0 = 1.5 =>floating-point division

Data Types

- Basic

`int, char, float, double.`

- Modifiers

`short, long, signed, unsigned.`

- Compound types

`pointers, structs, enums, arrays, etc.`

Common Errors

- Semi-colon missing
- Bracket mismatch
- Incorrect filename given in the command
- Incorrect location of file
- Not saving the changes.
- Not compiling the changes.

Best practices: Configure your editor properly, Indent your program! (tabs before code), Comment your program!

Today's class (8/8/2025)

- Operators
  - Arithmetic (+, -, *, /, %)
  - Relational (==, !=, >, <, >=, <=)
  - Assignment (+=, -=, *=, /=, %, <<=, >>=, &=, ^=, |=)   later
  - Increment / Decrement (++, - -)
  - Special: ternary, sizeof
- C program to print size of data types

Arithmetic Operators - example program

```c
int main() {
    int a = 10, b = 3;
    printf("a + b = %d\n", a + b);
    printf("a - b = %d\n", a - b);
    printf("a * b = %d\n", a * b);
    printf("a / b = %d\n", a / b);
    printf("a %% b = %d\n", a % b);
}
```

Relational Operators - example program

```c
int main() {
    int a = 10, b = 3;
    printf("a == b: %d\n", a == b);
    printf("a != b: %d\n", a != b);
    printf("a > b: %d\n", a > b);
    printf("a < b: %d\n", a < b);
    printf("a >= b: %d\n", a >= b);
    printf("a <= b: %d\n", a <= b);
}
```

Assignment Operators - example program

```
int main() {
    int x, a = 10, b = 3;
    x = a;
    x += b;
    printf("x += b: %d\n", x);
    x = a;
    x -= b;
    printf("x -= b: %d\n", x);
    x = a;
    x *= b;
    printf("x *= b: %d\n", x);
```

Assignment Operators - example program contd..

```
    x = a;
    x /= b;
    printf("x /= b: %d\n", x);
    x = a;
    x %= b;
    printf("x %%= b: %d\n", x);
}
```

Increment Decrement Operators - example program

```c
int main() {
    int a=10;
    int x=a;
    printf("x++: %d\n", x++);
    printf("++x: %d\n", ++x);
    printf("x--: %d\n", x--);
    printf("--x: %d\n", --x);
}
```

Special Operators - example program

```
int main() {
    int a=10, b=3;
    printf("Size of a: %zu\n", sizeof(a));
    int result = (a = b + 2, a * 2);
    printf("Comma operator result: %d\n", result);
    int max = (a > b) ? a : b;
    printf("max(a, b): %d\n", max);

}
```

More Operators

- Logical (&&, ||, !)
- Bitwise (&, |, ^, ~, <<, >>)

More on this next week..