Diploma in

Computer Science

C Language

Objectives

- Understand how C came into being and why it is such an important language
- Appreciate the features of C and their uses in programming
- Explore the applications of C in the real world
- Examine some of the popular libraries included in C

Introduction to C

A brief history

1972 - Dennis Ritchie creates C

A mixture of ALGOL, BCPL and B

A brief history

By mid-70 C used in many projects within the Bell system

Served as the basis for the implementation of C on several new platforms

1990 - C adopted by ISO

(International Standards Organisation)

Structure and organisation

C program is composed of:

- Header files
- Main function
- Variables
- Body
- Return statement

C header file

Contains C function declarations

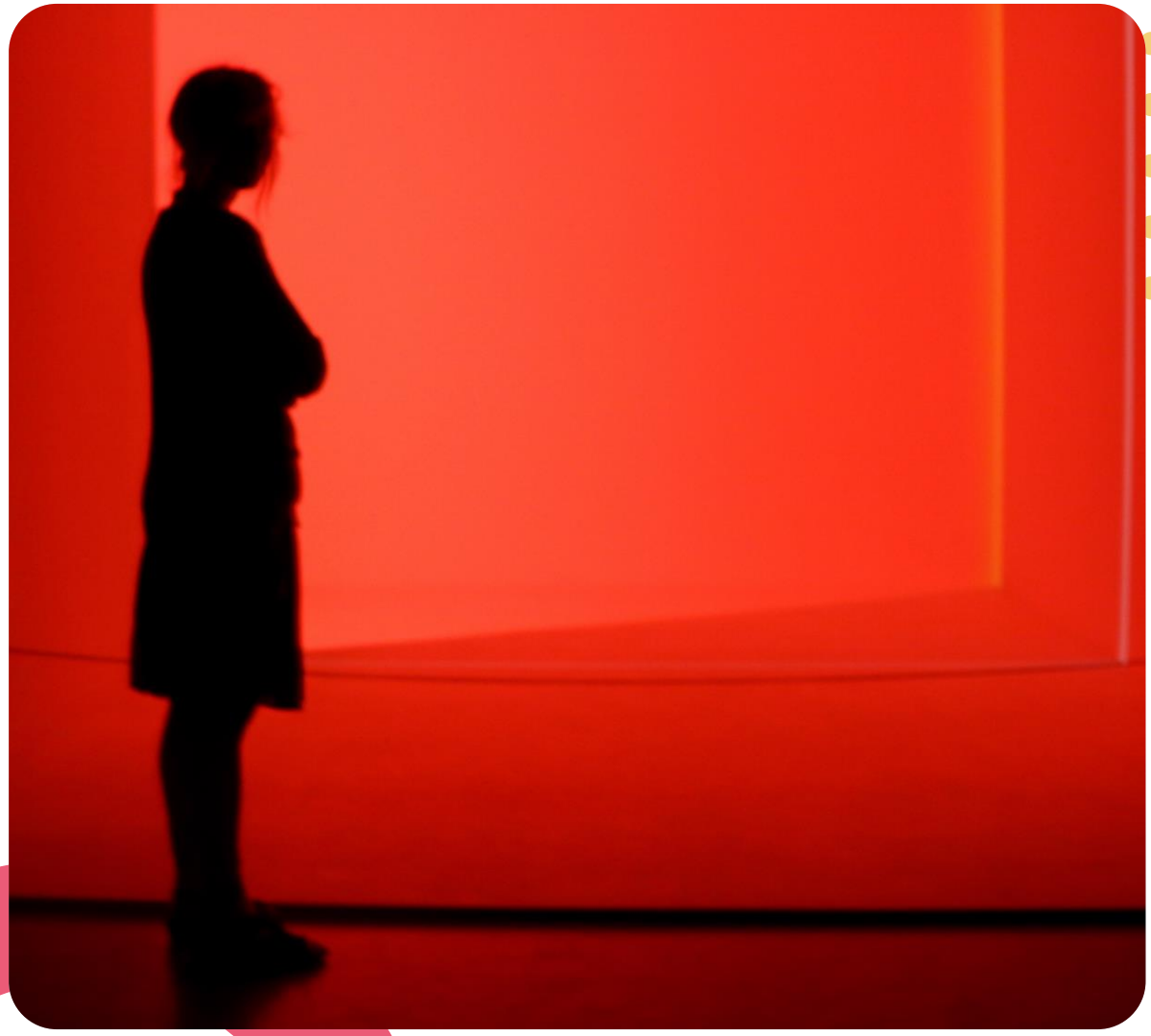
Has the file extension .h added to the program by pre-processer

Main function

- The core of every program
- Contains instructions for computer to perform specified task
- Computer looks for the main function

The body

- Contains the actual instructions the computer needs to carry out
- References outside the main function



Return Statement

Terminates the program and returns a value

If a function has the return type “void”, return statement can be used without a value

Return Statement

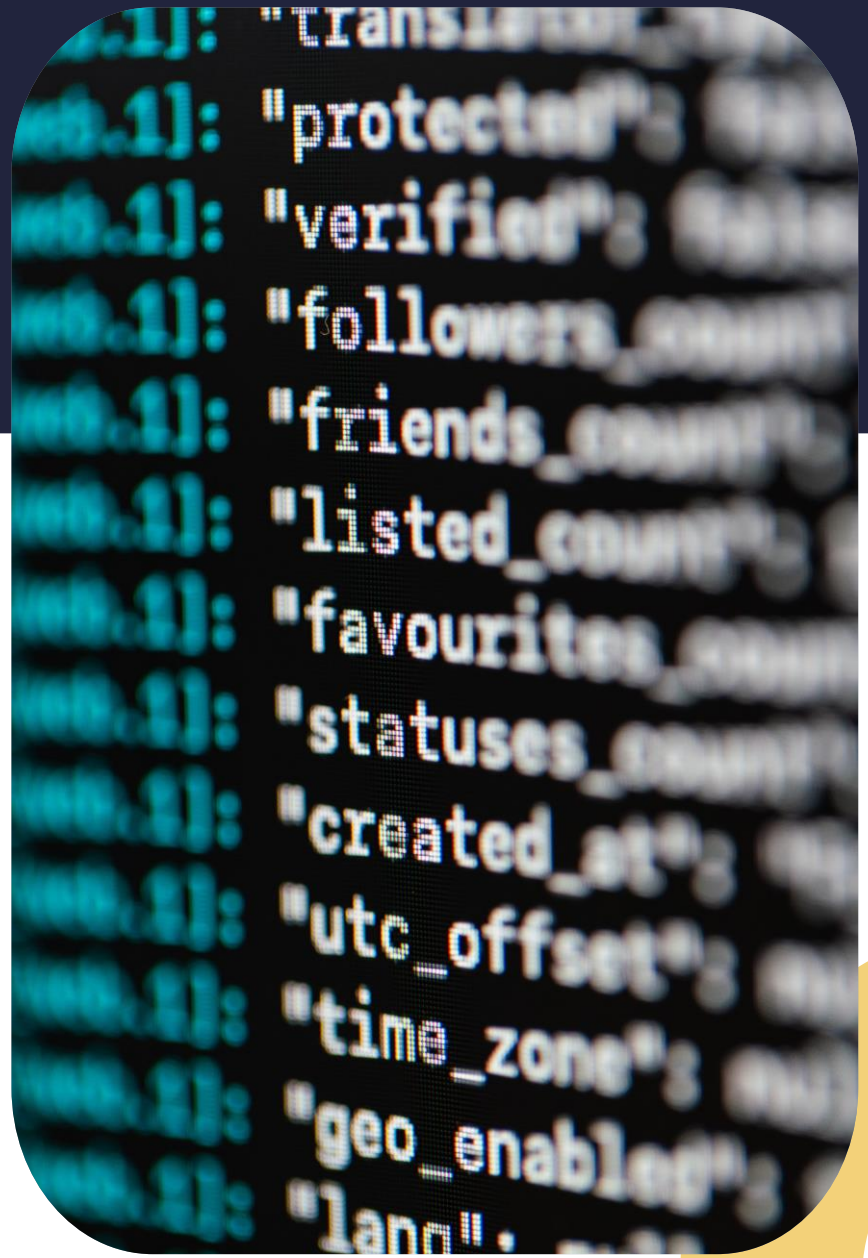
Comments are used in C to provide explanations and references

C supports inline and multiline comments

Features of C language

Speed and efficiency

- Middle level programming language - access to direct manipulation of computer hardware
- Statically typed - naturally faster than dynamically typed languages

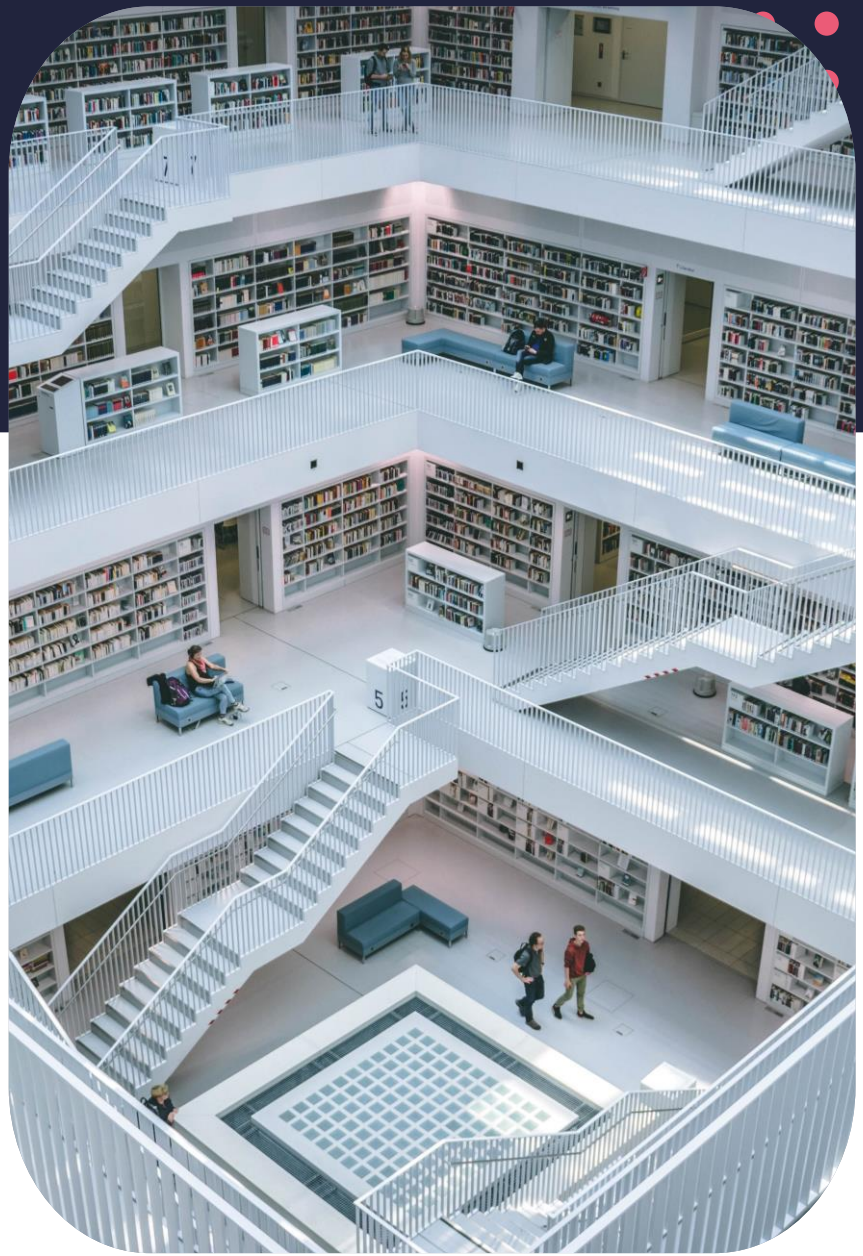


Procedural language

- Emphasis on the clarity of the code
- Code is divided into subroutines

Modular

- Language code stored in the form of libraries
- Library has wide range of built-in functions



Portable

Able to run the same code on different platforms

Compiled and run on any system with little to no changes!

General-purpose language

Used in a wide variety of applications

- Operating systems
- Databases
- Music players

Flexible

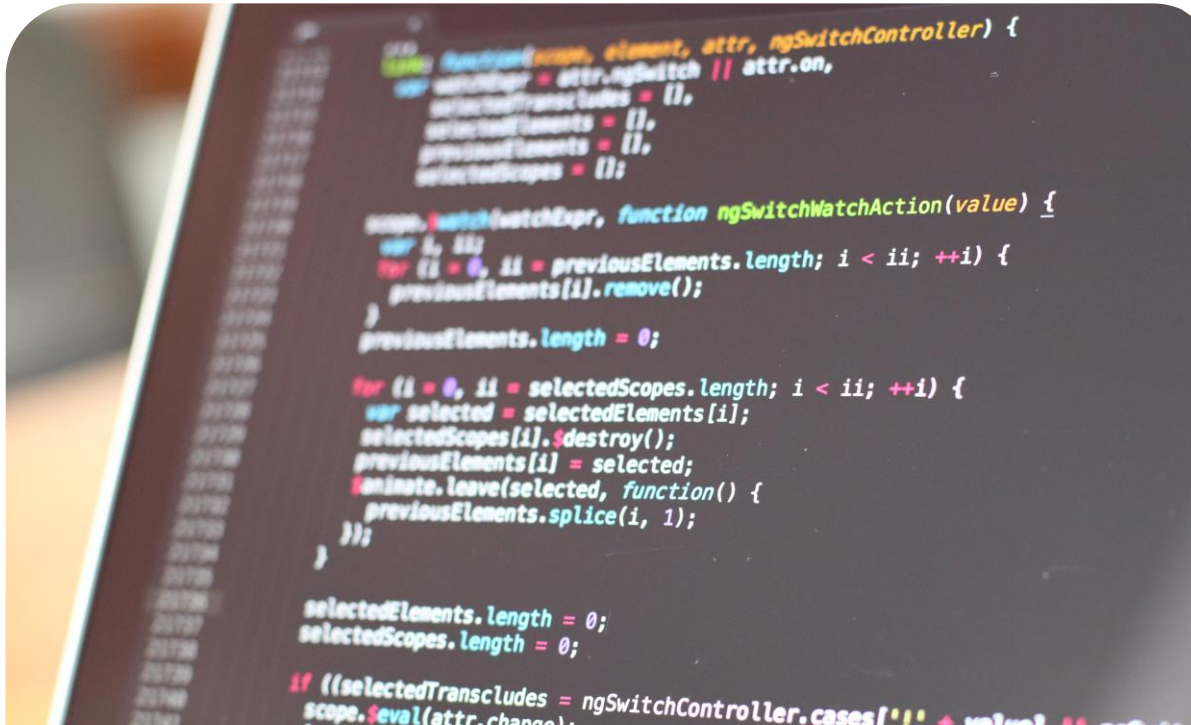
Programs written in C can easily be extended without significantly altering the existing program.

Related languages

C has influenced or is related to a number of languages.

C++ was an enhancement to C, adding object-oriented programming.

Objective C was developed to add Smalltalk like messaging to C.



Related languages

Java is directly related to both C and C++.

Inherits its syntax from C and object orientation from C++

Easy to learn Java if you know how to code in C++

Built to solve a different set of problems

Why learn C?

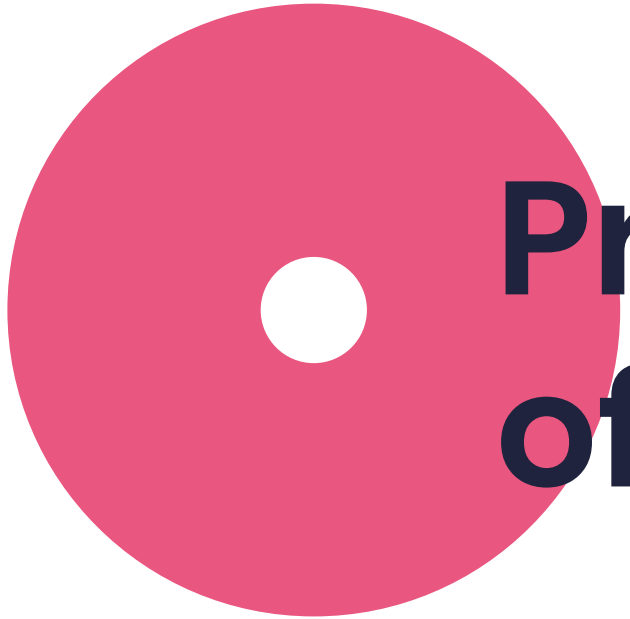
Allows you to produce portable code while maintaining performance

Is a stable language

Easier to master and apply the concepts of programming

C runs the world!

- Bulk of systems built using C
- Closer to the machine than other languages
- Been around for ages
- Concepts such as optimisation start to make sense



Properties of C

Programming languages are categorised into paradigms.

C language is the procedural imperative programming language

Does not have built-in support for other programming paradigms

C language describes exact procedure step by step

Generally easier to read and more maintainable and flexible

Libraries

Provides macros and type definitions

Provides functions for tasks like string handling, mathematical computations, input/output processing, memory management

Is presented in the form of header files

API (application programming interface)

Header files

- Provide function declarations and macro definitions
- Two types:
  - **System header files** declare the interfaces to parts of the operating system
  - **Your own header files** contain declarations for interfaces between the source files of your program

API of the C standard library

<assert.h>	Conditionally compiled macro that compares its argument to zero
<complex.h> (C99)	Complex number arithmetic
<ctype.h>	Functions to determine the type contained in character data
<errno.h>	Macros reporting error conditions
<fenv.h> (C99)	Floating-point environment
<float.h>	Limits of float types
<inttypes.h> (C99)	Format conversion of integer types
<iso646.h> (C95)	Alternative operator spellings
<limits.h>	Sizes of basic types
<locale.h>	Localisation utilities

The exact standard to which the header file conforms is in brackets if the header file is peculiar to that standard.

API of the C standard library

<math.h>	Common mathematics functions
<setjmp.h>	Nonlocal jumps
<signal.h>	Signal handling
<stdalign.h> (C11)	alignas and alignof convenience macros
<stdarg.h>	Variable arguments
<stdatomic.h> (C11)	Atomic types
<stdbool.h> (C11)	Macros for Boolean type
<stddef.h>	Common macro definitions
<stdint.h> (C11)	Fixed-width integer types
<stdio.h>	Input/output

The exact standard to which the header file conforms is in brackets if the header file is peculiar to that standard.

API of the C standard library

<stdlib.h>	General utilities: memory management, program utilities, string conversions, random numbers
<stdnoreturn.h> (C11)	no return convenience macros
<string.h>	String handling
<tgmath.h> (C99)	Type-generic math (macros wrapping math.h and complex.h)
<threads.h> (C11)	Thread library
<time.h>	Time/date utilities
<uchar.h> (C11)	UTF-16 and UTF-32-character utilities
<wchar.h> (C95)	Extended multibyte and wide character utilities
<wctype.h> (C95)	Functions to determine the type contained in wider character data

The exact standard to which the header file conforms is in brackets if the header file is peculiar to that standard.

Header files

Each header file provides particular functionality and services.

Example

Header file <stdio.h>

Memory management

- Functions are found in the <stdlib.h> header file
- Includes functions for allocating initialised and initialised arrays, releasing blocks of memory and extending allocated memory

Memory management

- Only reserve memory when you are using it
- Memory is not infinite – release it so that it can be used for other processes
- Embedded systems have built-in safety

ANYTHING can happen when a computer runs out of memory!

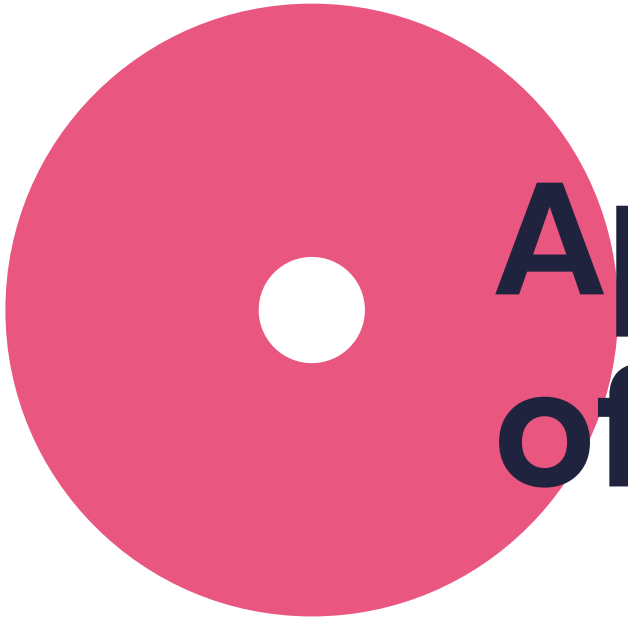
Memory management

- Does not have automatic memory management
- Program to avoid memory leak when dynamically allocated memory becomes unreachable

Language tools

Lint and *alint* provide a way for the developer to check the program for portability problems, syntax errors, wasteful style, and other forms of bugs.

This is a form of a debugger attached to the program.



Applications of C

Hardware programming

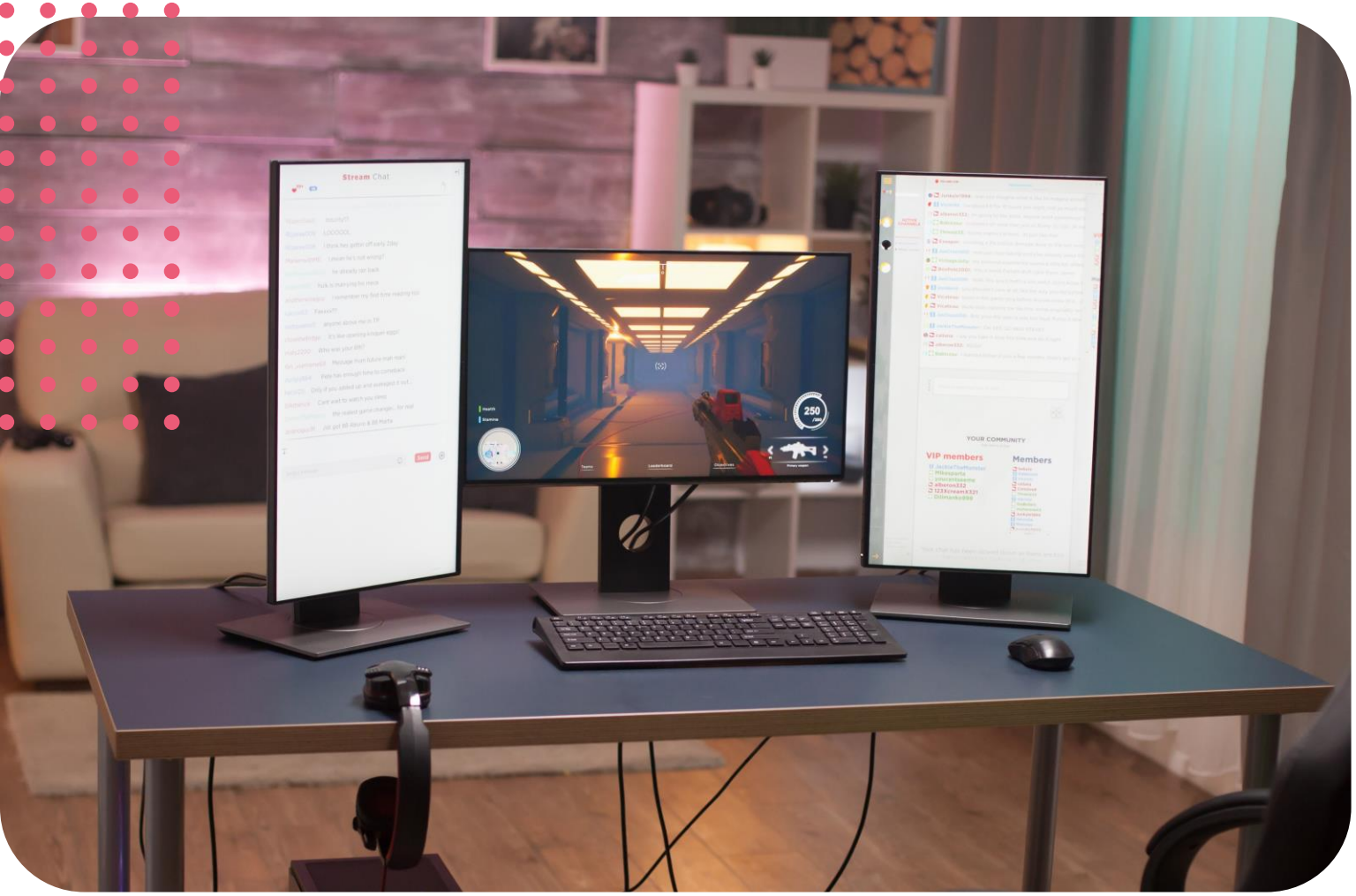
C is optimum choice when programming hardware

Small system footprint - good for embedded systems

Example

The Arduino Uno

Operating systems

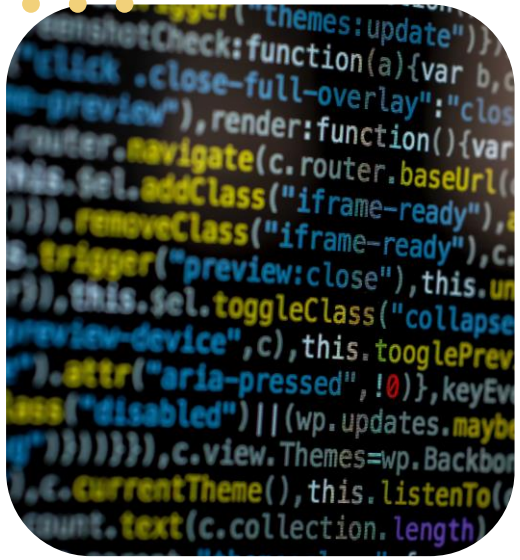


Most operating systems are written in C.

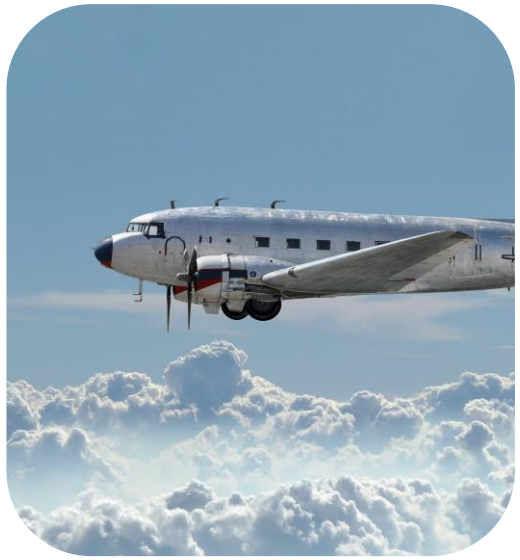
WHY?

It makes sense to use a programming language that allows you to access hardware with minimal abstraction.

Embedded systems

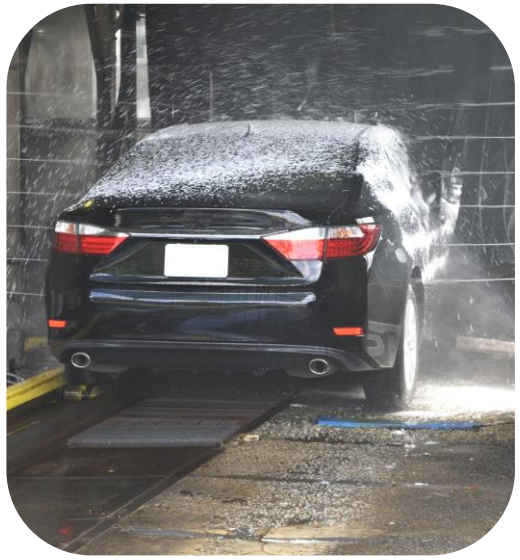


Specialised systems that run on time and memory constraints



Functions of embedded software initiated/ controlled through machine interfaces

Found in cars, washing machines, CCTV, routers, set top boxes and monitors



Required to use hardware resources as efficiently as possible and run as fast as possible

Other notable applications

Google Chrome

Mozilla Firefox

Compilers

Other notable applications

Games and graphics

Examples

Unity Unreal engine, Blender, Buildbox, CryEngine and Cube

Did you know?

C is the language that has existed for the longest period of time in computer programming history.

Why C?

Why C?

- C helps you to understand the internal architecture of a computer
- Precisely what we are trying to achieve in Computer Science!
- Easier to learn other programming languages
- Bare bones approach to solving problems

The king of operating systems!

Opportunity to work on open-source projects

On Linux you can edit your own kernel

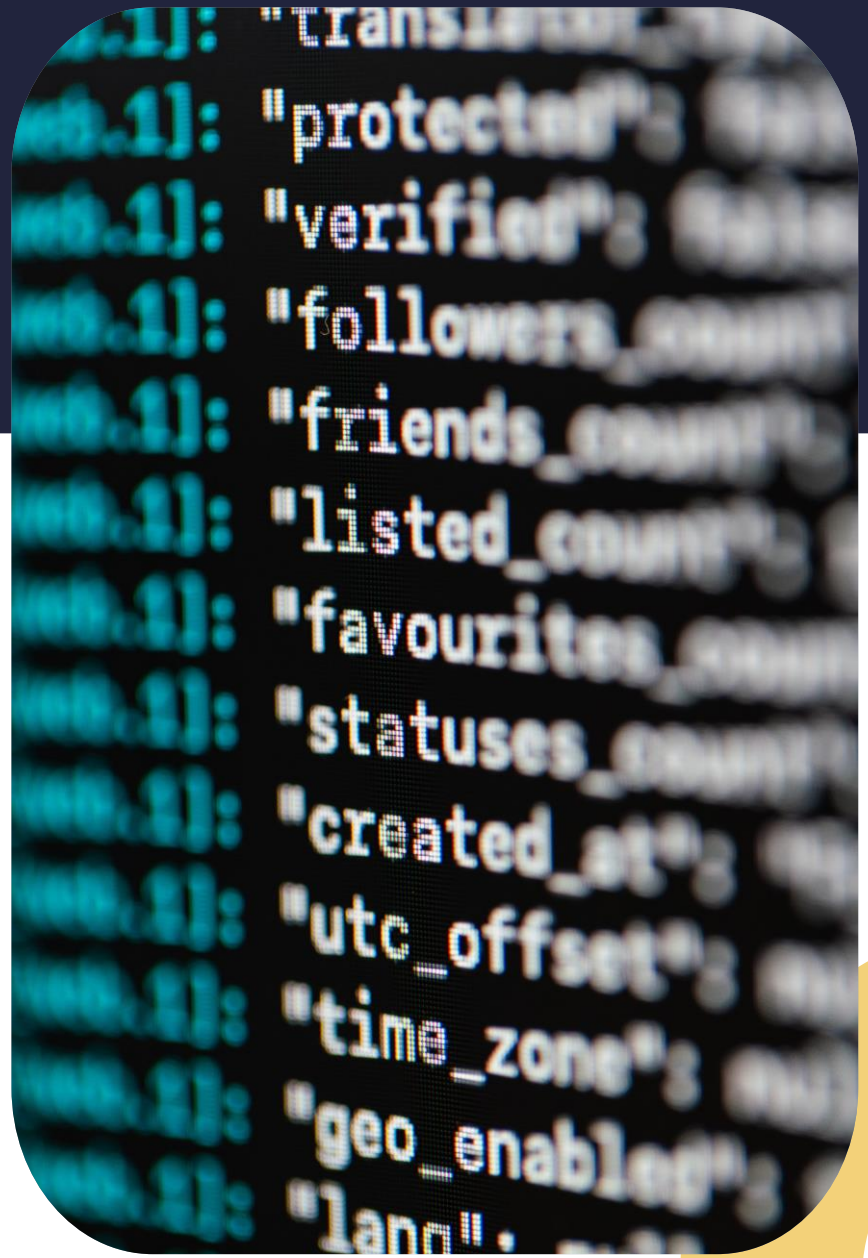
Understanding OS-level concepts can only be achieved in C

Computer Science

There are fields which are better understood using C than if you learnt them using any other language of choice.

Algorithms and data structures

- They are normally embedded in libraries and other resource files.
- C forces you to learn the concepts behind the code you are writing.



Not so fast!

- In other languages, the code feels light, but it is not.
- Everything in C compiles to a few highly optimised machine instructions.

C – a great jumpstart!

- C uses pointers that stretch your abstraction skills
- C close to the Von Neumann architecture and still expressive
- C enables you to give computer best possible instructions