C

...

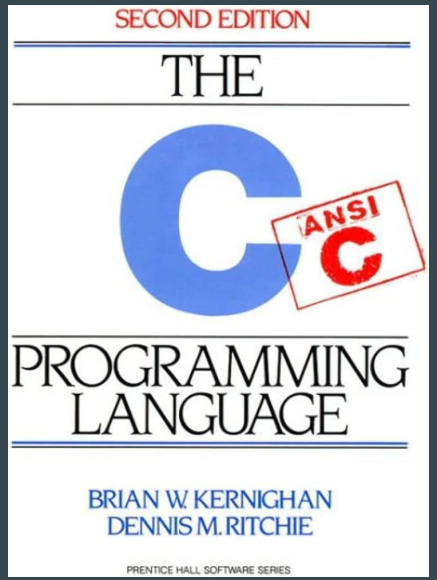
By Igor and Aja

Background

- Created in 1972 by Dennis Ritchie and Ken Thompson at Bell Labs
- Ritchie and his peers used language B (based on BCPL) to create Unix
- B had problems:
  - Single “cell” data type, which did not handle characters and strings well
  - No floating point arithmetic
  - Pointers had to convert from an index of an array of words to a byte address during runtime
- Realized they needed a typing system to deal with characters and byte addresses, and a way to work with floats

Background

- Ritchie and Thompson improved B, leading to the creation of C
  - Addition of types such as int, char and struct
  - More effective pointer system (values of array converted to pointers to objects in array
- In 1973, Ritchie and Thompson rewrote Unix in C
- In 1978 “The C Programming Language” was written
- C became the most popular programming language during the 1980s
- Still popular choice for writing operating systems, applications and embedded system software



Code sample

- scanf() accepts input similar to scanner in Java, requiring an expected input type
- When printing out a variable defining type of variable is required
- return 0 indicates code was successfully ran, return of -1 indicates a problem occuring
- Simple program like this using primitives does not need to worry about memory allocation
- https://replit.com/@IgorPono/AverageNumber#main.c

main.c > main

```c
#include <stdio.h>

int main(void) {
  double total = 0;
  int iterations = 0;
  while (1) {
    iterations = iterations + 1;
    double input;
    printf("enter your input\n");
    scanf("%lf", &input);
    total = total + input;
    double average = (total) / iterations;
    printf("your current average is: %lf \n", average);
    printf("\n");
  }
  return 0;
}
```

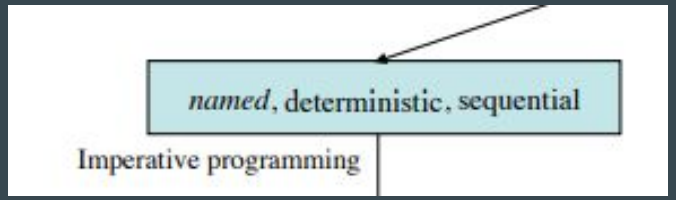
Classification (Paradigms)

Looking at expressiveness of state...

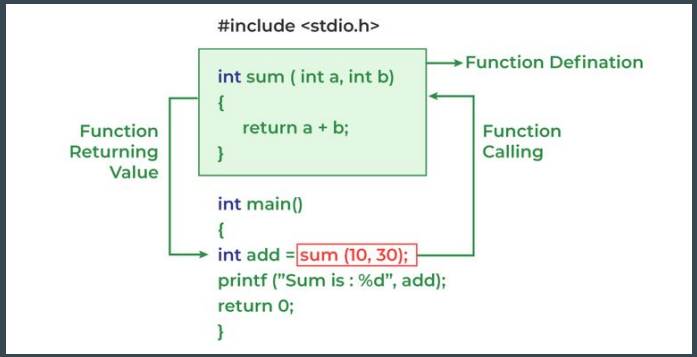
- Named arguments (order of func arguments doesn't matter)
- Deterministic (will always produce same results given same input)
- Sequential (executed in specified order)

Imperative

- Named arguments possible through passing struct as function parameter
- Being abstraction of assembly, C controls how a computation is done
- Programs are a sequence of commands that change the state.



Procedural (series of computational steps)



Classification (Design Principles)

- Abstraction
  - C is an abstraction of assembly, eliminating repetition
- Labeling
  - goto: (labeled section)
- Portability
  - C Compilers are available on any platform
- Preservation of Information
  - Values are stored BUT not very protected

C does not have

- Orthogonality
  - You can return a struct, but not an array
- Security
  - By design does not secure memory
- Defense in Depth
  - Easy to make mistakes, write over memory, segmentation faults

Evaluation

- Readability: Relatively Simple Syntax compared to BCLP (ASM) and B (low level), languages it was written to replace
- Writability: Less build in abstraction, not an Object Oriented Programming Language, it does not have the support other languages such as Java or C++ have for objects, No garbage collection
- Reliability: Incredibly easy to make bugs and mismanage memory, lack of garbage collection

```
GET "LIBHDR"

GLOBAL $(
        COUNT: 200
        ALL: 201
$)

LET TRY(LD, ROW, RD) BE
        TEST ROW = ALL THEN
                COUNT := COUNT + 1
        ELSE $(
                LET POSS = ALL & ~(LD | ROW | RD)
                UNTIL POSS = 0 DO $(
                        LET P = POSS & -POSS
                        POSS := POSS - P
                        TRY(LD + P << 1, ROW + P, RD + P >> 1)
                $)
        $)

LET START() = VALOF $(
        ALL := 1
        FOR I = 1 TO 12 DO $(
                COUNT := 0
                TRY(0, 0, 0)
                WRITEF("%I2-QUEENS PROBLEM HAS %I5 SOLUTIONS*N", I, COUNT)
                ALL := 2 * ALL + 1
        $)
        RESULTIS 0
$)
```

```
main() {
    extrn putchar, n, v;
    auto i, c, col, a;

    i = col = 0;
    while(i<n)
        v[i++] = 1;
    while(col<2*n) {
        a = n+1;
        c = i = 0;
        while (i<n) {
            c =+ v[i] *10;
            v[i++]  = c%a;
            c =/ a--;
        }

        putchar(c+'0');
        if(!(++col%5))
            putchar(col%50?' ': '*n');
    }
    putchar('*n*n');
}
v[2000];
n 2000;
```

Evaluation cont.

- Cost: C being a compiled language means faster runtimes, properly written C programs have fastest runtime compared to other languages
- Portability: any hardware capable of running C can execute a C based program with little to no modification, portability extremely beneficial for embedded systems
- Major Projects: Unix kernel, Windows utilities, portions of Android OS, OS x, Databases like Oracle, MySQL (C and C++), Embedded Systems applications and drivers, Interpreters and libraries of other languages often written in C (Python, Ruby, PHP, Javascript, etc)
- Community: Language still incredibly popular for systems and low level programming
- Community: “The C Programming Language” published in 1988 is well regarded

Conclusion

- Originally created to replace B programming language for projects at Bell Labs
- More difficult syntax than higher level languages but much simpler than the ASM and low level languages it was made to replace
- Lacks features of more “modern” languages but has far more adaptability for more platforms and projects
- Superior runtime for properly managed memory/lack of garbage collection
- Maintains dominance in OS creation, embedded systems, databases and libraries for other languages
- Overall great for learning lower level memory management in computer systems

Sources

https://www.bell-labs.com/usr/dmr/www/chist.html

https://www.simplilearn.com/tutorials/c-tutorial/use-of-c-language

https://www.unixmen.com/dennis-m-ritchie-father-c-programming-language/

https://www.toptal.com/c/after-all-these-years-the-world-is-still-powered-by-c-programming

https://unstop.com/blog/advantages-and-disadvantages-of-c-programming-language

https://www.geeksforgeeks.org/benefits-c-language-programming-languages/#

https://en.wikipedia.org/wiki/Named_parameter

https://en.wikipedia.org/wiki/Deterministic_algorithm

https://lucproglangcourse.github.io/principles.html

https://www.info.ucl.ac.be/~pvr/VanRoyChapter.pdf

https://www.geeksforgeeks.org/c-functions/