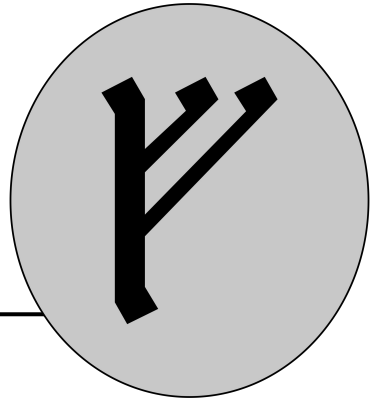
Function Details

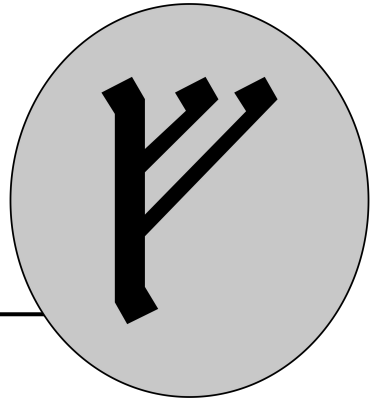
Pass-By-Value

- C is a pass-by-value language
  - When a function is invoked…
  - ...the *value* of the variable is **copied** to a new memory address



Pass-By-Value

- C is a pass-by-value language
  - When a value is *returned* by a function…
  - ...the value of the variable is **copied** to a new memory address



Pass-By-Value

- As opposed to pass-by-reference
  - Here, the memory location of the variables are passed - as opposed to the values being copied
  - Ex: Java

Memory Addresses

- “Every variable refers to one - and only one - unique memory address”

- Can access the memory location of variables by using the “&” key → called the address-of operator
- Can print memory addresses using “%p”*

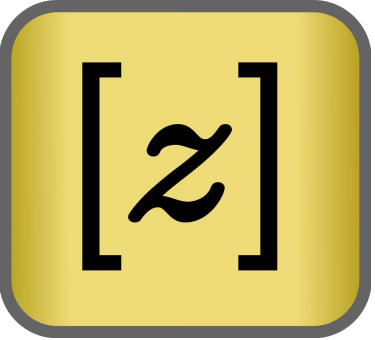
*Use add_one.c on github to see this in action

Scanf()

- We’ve seen the & before…

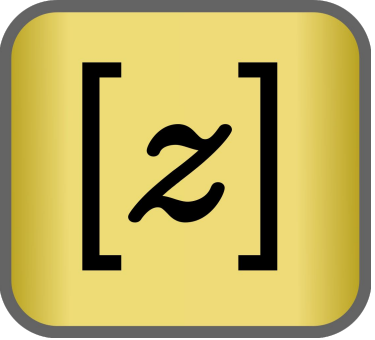
scanf(“%d”, **&**num);

- The & allows access to the *memory address* associated with the variable named num
- Scanf() takes the value typed in, and overwrites the value of that particular memory location



Function Parameters (Inputs)

- **Actual parameters**: the variables passed into a function
  - AKA: The value at the memory location associated with that variable
- **Formal parameters**: the variables in a function definition
  - AKA: The memory location that receives the copied value



Function Parameters (Inputs)

- The compiler matches the actual parameters with formal parameters
  - Performs the step of copying the actual parameters into the formal parameters
  - C requires the parameters to be the same type

Local Variables

- All functions (including main()) have their own set of **local variables**
  - These variables can ONLY be accessed inside that function

Variable scope

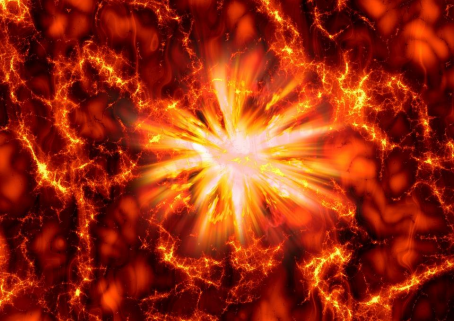
- Each local variable exists within its own function
  - The values may be copied, but the actual memory locations have no relation

- Add_one.c … how many *unique* variables (variables that have their own unique memory address) are present?

Variable scope

- Each variable exists within its own function
  - The values may be copied, but the actual memory locations have no relation

- Add_one.c … how many *unique* variables are present?
  - Answer = 2



Big-Bang Programming

- Writing the entire code at once
- DO NOT DO THIS
  - Will run into compilation / logical errors

Top-Down Programming

- Breaks the code into small, manageable pieces (aka functions)
- Much easier to…
  - Test
  - Visualize
  - Also faster to type (in the long run)

Function Details Coding

- Implement the ROT-13 homework (mid-October) with a function
- Requirements: Should have one function (which accepts one number as input, and returns one number) that performs the ROT-13 cipher
  - ALSO: At the top of the code, answer “How many unique variables are present?” in a comment

Function Details Coding

- Implement the Loop code from Exam 3 with functions

- Design is up to you
  - Challenge: Main() should be <= 13 lines
- ALSO: At the top of the code, answer “How many unique variables are present?” in a comment