Preprocessor directives

#means is compiler directives, include, define macro, ifdef/ifndef

extern, function are extern by default, makes things global

static, scope of file declared, keeps val across function call

Array is contiguous in mem, can be more than 1 d, in c is pass by reference, and is pointer to start, strings are arrays as well, they end with /0.

Pointers are pointing to memory, can be null, pointer stores memory address, every byte has address,. pointers have addresses too. Can do arithmetic with pointer, some don't make sense. Arrays are pointers to the start of an array, can be add to the pointer to go through the array. Adding to pointer adds in base size.

Function pointers point to executable code in memory.

Sorting adds info, makes merging and finding duplicate in list fast

Time complexity, how many times it is done given input size, O(n^2), insert, bubble sort. O(n^5/2) shell sort, merge, quick, O(nlogn).

Dynamic allocation, allocating memory for vars on the heap during run time, not done by the compiler. Use if we don't know mem size yet, heap is slower than stack.

Make, compiles the code. rule is a target with a set of dependencies and commands, phony target is a target that doesn't make a thing as named. Can use variables and assign contents in make. Dependency, some rule needs another. Can use shell commands in make. Can call another makefile. Topological order of targets.

Data compression. We will lose some in transfer, called entropy, calculate with entropy is sum of (probability of getting a certain thing times log2(probability of a certain thing).)

Bash, command interpreter, gives commands to OS. Bash reads input with one line is one command, line split into tokens by whitespace. Can do functions, or execute files. echo is to print to screen. First line of scr is interpreter directive. dotfiles are hidden files in homedir.

Params and vars in bash, can do strings or ints or non homogeneous arrays, assign with =.

$ is expand, substitutes variable with value. ex j=1 echo $j.Arithmetic is done in (()), can expand result with $.

Functions in bash can have many commands and does local variables.

Arrays is a list of strings, can do arr=(), can do explicit indexing, and have gaps in it.

=+ to array append. 4{[arr[n]} to get nth index of array. Do testing in [[]], logical operands. Does lexicographic compares with string. -z checks if is empty, -n checks if not empty. Also test arithmetics in integers, can also test < in arithmetic context

Pipes, stdout->stdin from process to process. awk is a text parser that separates by line.

Arithmetic right shift clones bit size to front.

Linked list, nodes instead of place in memory, no fixed memory allocation, no shift after insert or delete. But uses more memory,, no random access.

Tree is a DAG, preorder is to go to a thing then its children, inorder is left subtree then node then right subtree, Postorder is both subtrees, then self.

File extensions are convention in linux, files can be accessed sequentially or with random access.

Files are in directories, which are in partitions of disk. open() is sequential access. Path names start at root, all files are just bytes.

Cryptography, encrypt with cipher, only cipher holder can read, ex. one time pad. Distributing a key is hard. Public key crypto, make a private key for dec, public key for enc, can share pubkey, is usually slow.

Language translators.

compilation of program. Preprocessors do macros and include->compiler, make code to assembly->assembler, makes files into binary->linker links into one executable->Loader allocates and manages memory.

Memory has text, uninit data, init data, heap, and stack.

Processes are things that run on CPU, only one process per core at once, multiprogramming is processes take turns running on a core. Protect memory of one process from others. Fixed memory partition is when each process gets a partition in memory. Can do logical addresses, are “fake” addresses with base in limit, that are not starting at physical address 0, the program does not know this. Allocation: find space large enough for the program to run, run it there.

Virtual memory, give more memory than exists, keep frequently used stuff in ram, move less used stuff to disk, again hide this from the process itself. Given the program virtual addresses, the memory management unit translates this.

Paging: unit of mapping is a page, each entry in a page table is a page. Virtual addresses mapped to physical addresses, Not all virtual memory has a physical page, and not all pages are used. PTE is the page table entry and contains translation of a physical page.

Processes are made by os, execution, or other processes, Can end either voluntarily, (normal/error exit) or involuntarily(fatal error or killed). Process states are created, reading, running, blocked, exit. This is managed by the scheduler, which makes it blocked if waiting on something.

Multithreading: process is a thing with core data and a state each process can run in states, each thread is its own stack in address space, threads mean a program can be waiting and still running. Threads share data, and context swap thread is faster than process. Race conditions: threads share memory,we don't know if R/W is back to back. Fix with Critical regions which means: No two process in in one crit region, No assume speed, No process outside a critical region may block another, A process may not wait forever for crit region.

Deadlock is when cyclical resource wait, needs mutual exclusion, hold and wait, no preemption, and circular wait.

how to fix it? Pretend it doesn’t happen. Or try to recover by taking resources from other process, rolling back to checkpoints, or killing processes. Prevent deadlock: allow programs to spool things into a resource, not claiming it. A process needs to request everything at the start, no hold and wait. No preemption, just take resources from other processes, this breaks things. No circular wait, give resources priorities to get used.

Regex: Series of chars in a search pattern, matches chars with chars in pattern.

Finite automata: Abstract machines that recognize patterns. Regex is a generator of patterns of strings.

Types of: DFA: single transition per letter, recognize regex, NFA(non deterministic): have many transitions per letter, have transitions for the empty string. Push down is an automata with stack. Go from a starting state to a final state by matching the states. NFA, can match between different states. Backtrack is stuck, NFA can also simulate multiple states at once, and is faster.