Slides

C_pl
Practice Problem: Write a function print binary that has one unsigned int parameter and prints out the binary representation of that number.

Knowledge Check: What does c >= 'a' && c <= 'z' check for us?
What does c = c - 'a' + 'A' accomplish? What did we need to remember to write that?

Knowledge Check: What's the largest number you could represent in base-10 with 6 digits? What about 15 digits? What's the largest number you could represent in binary with 12 bits? What about with 32 bits?

Pointers
Practice Question: Write a function argmax that returns the index of the maximal item in an array of integers. What parameters does your function need to have?

Practice Question: Write a function replace which takes in an array and is parameterized by two integers tar and repl and replaces every instance of tar in the given array with repl.

Practice Question: Write the function str replace as described below.
int str_replace(char *s, char find, char repl);
str replace replaces every instance of find in string s with the character repl and returns the number of instances replaced.

Knowledge Check: What would happen if we hadn't switched the break to continue? What would happen if we just deleted break?

Knowledge Check: how does strlen work without being provided a size parameter?

Dynamic_mem
Practice Question: Write a function void primes(int n) that takes as a parameter one int and prints out (in order) all of the prime numbers smaller than that int.

Practice Question: Write a function int str_to_int(const char *s) that takes in a string texttts that is a numeric only string and

returns the integer that string represents.

Practice Question: Write a function digits that takes in an unsigned int and returns a pointer to an array that stores the digits of the int in order from most significant to least significant.
For example digits(257) would return an array that contained 2, 5, and 7 in that order. digits(72519) would return an array that contained 7, 2, 5, 1, and 9 in that order.

Knowledge Check: Why is it dangerous to return a pointer to a local stack variable from a function, and what kind of behavior can that cause?

Knowledge Check: How does using a doubling strategy to grow an array improve the performance of appending elements compared to growing the array one element at a time?

Cmd_line
Practice Question: Write a program that accepts two command line arguments as strings, converts them to integers, multiplies them, and prints the result. Use the parameters from main (argc and argv).

Practice Question: Write a function that creates an n×m identity matrix. Allocate memory for a 2D array, set the diagonal elements to 1 and all others to 0, then return a pointer to the matrix. Also write a main function to display the matrix and free all allocated memory.

Knowledge Check: What does the argc parameter represent in main, and how does it help when processing command line arguments?

Knowledge Check: How does the declaration "char *argv[]" relate to "char **argv" in terms of data structure and pointer behavior?

Knowledge Check: What are the pros and cons of using an array of pointers to build a 2D array compared to simulating a 2D array with a single 1D array?

Lab 3
Create a bash script called produceOutputs. This script takes two command line arguments: a program executable and a test set file. It should run the executable on each test case defined in the test set file by redirecting input from the corresponding .in file and supplying arguments from the .args file (if they exist). The script must handle cases where a test case has only an input file, only an arguments file, or both, and print a usage message when not given exactly two arguments.

Lab 4

Write a C program that demonstrates two input-handling tasks. First, implement a function readint() that skips leading whitespace, reads an integer from stdin using getchar(), and stops when a non-digit is encountered. In main(), use this function to read an integer, multiply it by 3, and print the result. Second, write a short program that uses getchar() twice to read two characters from stdin and then prints those characters along with their ASCII values.

Lab 5
Develop two C programs:

1. decimal_hamming.c: Read two unsigned integers (ensuring they have the same number of digits) from stdin and compute their base-10 Hamming distance (i.e., count the positions where their digits differ).
2. binary_hamming.c: Read two unsigned integers from stdin, convert them to their 32-bit binary representations, and compute the base-2 Hamming distance by counting the differing bits.

Lab 6
Implement a version of the Bulls and Cows game in C. The program should take a secret codeword as a command line argument. For each guess the player makes (via standard input), the program must report how many Bulls (correct character in the correct position) and Cows (correct character in the wrong position) are present. The game should continue until the player either guesses the codeword correctly, makes 6 guesses, or receives EOF as input.

Assignment 1
Create a set of Bash scripts to automate the testing of programs. Your tasks are as follows:

1. Clone the provided course repository (using Git) and locate the file hello.txt within the a1 directory. This file must be submitted with your assignment.
2. Write a script named testDescribe that takes one command line argument—a filepath to a test set file. For each file stem listed in the test set file, the script should:
   - Print the contents of stem.desc if the file exists.
   - Otherwise, output “stem: No test description”.
3. Write a script named runInTests that takes two arguments: a command to run and a test set file. For each file stem in the test set file, your script should:
   - Run the command with input redirected from stem.in.
   - Compare the program’s output with the contents of stem.out.
   - Report “Test stem passed” if they match or “Test stem failed” (followed by expected and actual output) if they do not.
4. Update runInTests to create a new script named runTests. This script must, for each test stem, pass both the input from stem.in and any command line arguments from stem.args when executing the command.
5. (Extra) Enhance your runTests script to handle cases where the expected output file is missing and to optionally print test descriptions from stem.desc.

Assignment 2

Implement three separate C programs according to the following specifications. All programs must be compiled with the flags:
gcc -Wall -Wvla -Werror
and are only allowed to include the header stdio.h.

1. rotate.c:
   - Read a single integer from standard input.
   - Print every right rotation of the integer (where a rotation moves the least significant digit to the most significant position) on a new line.

For example, an input of 5347 should produce:
5347
7543
4753
3475

2. luhns.c:
   -Read an account number from standard input by processing digit characters until a non-digit is encountered.
   -Compute the checksum using Luhn's Algorithm without using any arrays.
   -Print "Valid" if the computed checksum matches the check digit, or "Invalid" otherwise.
3. numerals.c:
   -Read one Roman numeral (ignoring leading whitespace) from standard input.
   -Convert it to its equivalent Arabic numeral (integer) following the rules of subtraction for smaller numerals preceding larger ones.
   -Print the resulting integer.