Week 2 - Friday

COMP 2400

Last time

- What did we talk about last time?
- C literals
- Binary representation
- Math library

Questions?

Project 1

Quotes

Unity can only be manifested by the Binary. Unity itself and the idea of Unity are already two.

Buddha

Math Library

Math library

Function	Result	Function	Result
`cos(double theta)`	Cosine of **theta**	`exp(double x)`	e^x
`sin(double theta)`	Sine of **theta**	`log(double x)`	Natural logarithm of **x**
`tan(double theta)`	Tangent of **theta**	`log10(double x)`	Common logarithm of **x**
`acos(double x)`	Arc cosine of **x**	`pow(double base, double exponent)`	Raise **base** to power **exponent**
`asin(double x)`	Arc sine of **x**	`sqrt(double x)`	Square root of **x**
`atan(double x)`	Arc tangent of **x**	`ceil(double x)`	Round up value of x
`atan2(double y, double x)`	Arc tangent of **y/x**	`floor(double x)`	Round down value of **x**
`fabs(double x)`	Absolute value of **x**	`fmod(double value, double divisor)`	Remainder of dividing **value** by **divisor**

It doesn't work!

- Just using **`#include`** gives the headers for math functions, not the actual code
- You must link the math library with flag **`–lm`**

```
> gcc hypotenuse.c -o hypotenuse -lm
```

- Now, how are you supposed to know that?

```
> man 3 sqrt
```

Example

- You are sitting at the origin
- There's a hyperspace ghost demon at location (***x***,***y***)
- Write a program to determine the angle to fire your C-controlled proton accelerator in order to remove the deadly menace

Single Character I/O

`getchar()`

- We haven't talked about any input in C yet
- To read the next character from input, you can use the **`getchar()`** function
- It will return the value of the next character (as an **`int`**) or **`-1`** if the end of the file is reached
  - Store the value as an **`int`** first to check to see if the end of the file has been reached
  - If not, you can then store it as a **`char`**

```
int value = getchar();
if (value == -1) {
    printf("End of file!");
}
```

`putchar()`

- **`putchar()`** is the output equivalent of **`getchar()`**
- It outputs a single character at a time
- You could use **`printf()`** with the **`%c`** formatter instead, but **`putchar()`** can be more convenient for single characters

```
char letter = 's';
putchar('q');
putchar(letter);
```

Input example

- Let's write a function that reads input, character by character, and returns the equivalent **`int`** value
  - For example, the characters **`'4'`**, **`'5'`**, **`'1'`**, and **`' '`** would be interpreted as the **`int`** **`451`**
- We'll read **`char`** values until we get a space, newline, or EOF
- Each time, we multiply our sum by 10 and then add the numerical value of the input
  - We have to subtract **`'0'`** from the input, otherwise we'll get the character values of the digits 0 through 9 (which are **not** 0 through 9)
- Note: a function like this will be provided for you for some labs

Ticket Out the Door

Upcoming

Next time…

- `sizeof` and `const`
- System limits
- Bitwise operations

Reminders

- Keep reading K&R chapter 2
- Read LPI chapter 11
- Keep working on Project 1
  - Due next Friday by midnight!