Programming Workshops

Principles of Computing

Dr. Joseph Walton-Rivers

Week 3

Today's Topics

- Numerical Representation
- Pixels
- Libraries

What number is this?

What number is this?

42

What number is this?

42

OK, no surprises there then. But what do these number symbols actually *represent*?

Left to Right, High to Low

- We're writing the **most significant** digit to the left
- Followed by the next most and so on
- Each extra digit means 'shift by a factor of 10'

Another way of thinking about it

- I have 4 quantities of 10 (4 x 10), and
- I have 2 quantities of 1 (2 x 1)

Did I have to choose 10?

- We chose 10 to be the thing we shift by for every space, but do we have to?
- What if we chose 8 as the amount to shift by?
- 9 no longer makes sense as a digit, - we don't have (I have 11 ones, we say I have 1 quantity of ten and 1 quantity of 1)
- So 9 would actually be 1 quantity of 8, and 1 quantity of 1

Base 8, 42

- OK, so let’s figure out how many 8s we have: 5 (and a bit)
- So that’d be (5 x 8 = 40) + ??
- How many units?
- 2!

In general...

- We call our ‘shift amount’ a **base**

In general...

- We call our ‘shift amount’ a **base**
- When our case is 8, we call it octal

In general...

- We call our ‘shift amount’ a **base**
- When our case is 8, we call it octal
- When our base is 2, we call it binary

In general...

- We call our ‘shift amount’ a **base**
- When our case is 8, we call it octal
- When our base is 2, we call it binary
- Each shift can be thought of a adding or subtracting one to the exponent of the base

In general...

- We call our ‘shift amount’ a **base**
- When our case is 8, we call it octal
- When our base is 2, we call it binary
- Each shift can be thought of a adding or subtracting one to the exponent of the base
- We start at 0, because maths loves computer scientists

In general...

- We call our ‘shift amount’ a **base**
- When our case is 8, we call it octal
- When our base is 2, we call it binary
- Each shift can be thought of a adding or subtracting one to the exponent of the base
- We start at 0, because maths loves computer scientists
- Also, boring math reasons...

Lets take a look more generally

8^1	8^0
5	2

10^1	10^0
4	2

2^5	2^4	2^3	2^2	2^1	2^0
1	0	1	0	1	0

(aside) 42 looks quite pretty in binary (base 2).

Ambiguity

OK, what’s this number?

101

Ambiguity

OK, what's this number?

101

- Are you sure?
- What base was I using?
    - If this was base 2, I would mean 5

Ambiguity

OK, what’s this number?

$$101$$

- Are you sure?
- What base was I using?
  - If this was base 2, I would mean 5
  - If this was base 8, I would mean 65

Ambiguity

OK, what's this number?

101

- Are you sure?
- What base was I using?
  - If this was base 2, I would mean 5
  - If this was base 8, I would mean 65
  - If this was base 10, I would mean 101

Ambiguity

OK, what's this number?

101

- Are you sure?
- What base was I using?
    - If this was base 2, I would mean 5
    - If this was base 8, I would mean 65
    - If this was base 10, I would mean 101
    - If this was base 16, I would mean 257

Ambiguity

OK, what’s this number?

101

- Are you sure?
- What base was I using?
    - If this was base 2, I would mean 5
    - If this was base 8, I would mean 65
    - If this was base 10, I would mean 101
    - If this was base 16, I would mean 257

Ambiguity

OK, what's this number?

101

- Are you sure?
- What base was I using?
  - If this was base 2, I would mean 5
  - If this was base 8, I would mean 65
  - If this was base 10, I would mean 101
  - If this was base 16, I would mean 257
- Now I have introduced ambiguity where there was none

Prefix

- When working in different bases, maths tends to denote this somehow

Prefix

- When working in different bases, maths tends to denote this somehow
- They need to work with arbitrary bases, we don’t generally need this in CS

Prefix

- When working in different bases, maths tends to denote this somehow
- They need to work with arbitrary bases, we don't generally need this in CS
- (unless we're stealing their homework for our algorithms...)

Prefix

- When working in different bases, maths tends to denote this somehow
- They need to work with arbitrary bases, we don’t generally need this in CS
- (unless we’re stealing their homework for our algorithms…)
- When working in computing, we tend to also do this, with a prefix

Prefix

- When working in different bases, maths tends to denote this somehow
- They need to work with arbitrary bases, we don’t generally need this in CS
- (unless we’re stealing their homework for our algorithms…)
- When working in computing, we tend to also do this, with a prefix

Prefix

- When working in different bases, maths tends to denote this somehow
- They need to work with arbitrary bases, we don't generally need this in CS
- (unless we're stealing their homework for our algorithms...)
- When working in computing, we tend to also do this, with a prefix

0b	Binary
0o	Octal
0x	Hexadecimal

Ambiguity

OK, what's this number?

0b101

Ambiguity

OK, what’s this number?

0b101

- This would be 5 in base 10
- It’d be read/execute in Linux
- as in chmod 550
- Linux permissions are actually 3-bit bit masks, that we write in octal for fun

Hex and Digits

- Write the digit representing 9 + 1 in hexadecimal

Hex and Digits

- Write the digit representing 9 + 1 in hexadecimal
- What do you mean ‘10‘ doesn’t fit in the box?

Hex and Digits

- Write the digit representing 9 + 1 in hexadecimal
- What do you mean ‘10‘ doesn’t fit in the box?
- We need a way of spelling ‘10‘ using a single character?

Hex and Digits

- Write the digit representing 9 + 1 in hexadecimal
- What do you mean ‘10‘ doesn’t fit in the box?
- We need a way of spelling ‘10‘ using a single character?
- errr… A, that’ll do.

Hex and Digits

- Write the digit representing 9 + 1 in hexadecimal
- What do you mean ‘10‘ doesn’t fit in the box?
- We need a way of spelling ‘10‘ using a single character?
- errr... A, that’ll do.
- aside, (also, by the way A has 65 as an ASCII value)

Hex and Digits

- For bases lager than 10, we run out of numbers
- It’s common for use to use letters as the glyphs for these
- The ‘humans should use base-12’ people invented their own.

Hex and Digits

- For bases lager than 10, we run out of numbers
- It’s common for use to use letters as the glyphs for these
- The ‘humans should use base-12’ people invented their own. Let’s ignore them.

Colours

Images

- Our monitors are conceptually little blocks of tiny lights with red, green and blue parts
- We represent colour as amount of brightness of each of these lights
- We arange them on a square grid
- I quite like grids.

(one of) Joe's Favourite Equation: Row-major ordering

$$i = y * width + x$$

- Y quantities of width
- X quantities of 1s

(one of) Joe’s Favourite Equation: Row-major ordering

$$i = y * width + x$$

- Y quantities of width
- X quantities of 1s
- Sound familiar?
- It’s like counting in base-grid!

Meanwhile... Back at the point

- We have 255 levels of brightness for each tiny little lamp
- 0 - 255 (256 possible values), fits nicely into a byte
- so, R = 255, G = 255, B = 255 describes one ’picture element’ (pixel)
- If only we had a more concise way to represent this.
- Did you know a single Hexadecimal digit can represent half a byte?

Pictures as grids

- We can think of a picture as a grid of cells containing Red, Green and Blue elements
- We can use the row/column-major ordering trick to store this as one big block in memory
- We can also iterate through X and Y using loops!

Recap: loops

- We know the width
- We want to start at a given point (x = 0) and keep going till we reach the width
- What kind of loop makes the most sense?

Recap: loops

- We know the width
- We want to start at a given point (x = 0) and keep going till we reach the width
- What kind of loop makes the most sense?
- For loop