Numbers

Introduction to Computer Science

Module: 4CC509

Today

- What do we mean by numbers?
  - In our daily life, we do not really tend to think about this, but it is important when it comes to representing them in a computer
- Start to look at how we represent them in a computer.

What number is this?

- This is the number we would represent as 273, but using the Egyptian system for representing numbers.

 = one

 = ten

 = one hundred

- The Egyptians did amazing things using this way of representing numbers, but it does present some problems.

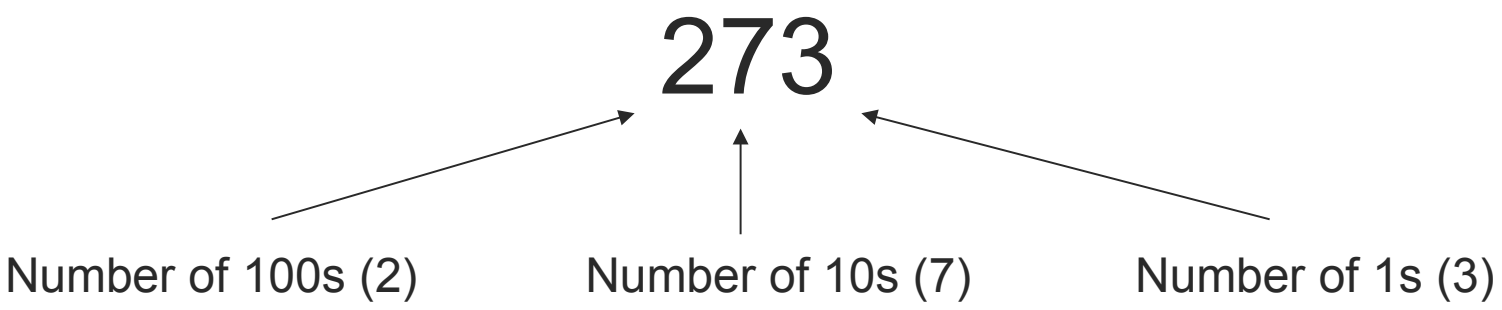
Problem

- What character is used to represent:
  - thousands?
  - ten thousands?
  - etc
- The problem with this system is that a different symbol is used for each order of magnitude. This means that new symbols are needed as larger numbers need to be represented.

Solution

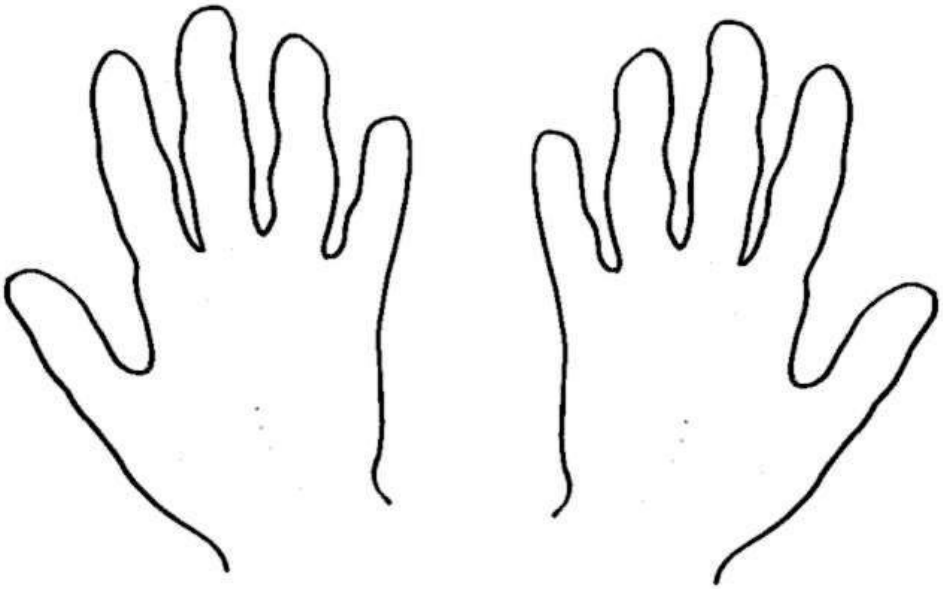
- Today, we use a positional system for representing numbers.
- The number system we are used to uses a place value for the number of 1s, the number of 10s, the number of 100s and so on

What does 273 Mean?



$2 \times 10^2 + 7 \times 10^1 + 3 \times 10^0$

Why do we use 10s ?



Whole Numbers

- 1, 2, 3, 4, 5, etc.
- Also referred to as *counting* numbers, *cardinal* numbers or *natural* numbers
- Probably the first mathematical objects conceived by early humans

Back to 

- There is another problem with this system of representing numbers.
- What is it?

How do you represent zero?

Integers

- For many centuries, zero was not included in the set of whole numbers
- Neither were negative numbers
- When talking about whole numbers that include negative values and zero, we refer to them as *integers*.

  … -3, -2, -1, 0, 1, 2, 3, …

Rational Numbers

- Rational numbers are numbers that can be expressed in terms of the ratio of two integers
- For example, is a rational number since it is the ratio of the two integers 3 and 5. We can also write this number in its decimal form 0.6.
- Note that a denominator of 0 (i.e. zero on the bottom) is not allowed.

Rational Numbers

- The set of rational numbers also contains all of the integers since any integer can be written as a ratio with 1 on the bottom
- For example, 47 can be written as

- Any number that can be written as a finite set of decimal digits is also a rational number
- For example, -23.45678 can be written as

Rational Numbers

- Some rational numbers require an infinite number of digits in order to be expressed in decimal form
- For example, requires an infinite number of digits after the decimal point when expressed in digital form, e.g. 0.3333333…

- However, this is still a rational number since it can be expressed as a ratio of the digits 1 and 3.

Irrational Numbers

- Irrational numbers are those numbers that *cannot* be expressed as the ratio of two integers
- For example, is irrational. If you try to write it as a decimal, the digits keep going with no discernable pattern, e.g. 1.41421356237309504…
- The key point however, and what makes it irrational, is that cannot be expressed as the ratio of two integers.

Irrational Numbers

- Another classic irrational number is . When written as a decimal, it is a series of digits with no known recurring pattern, e.g. 3.14159265…
- Once again, this is irrational because there is no known ratio of two integers that can be used to represent it.

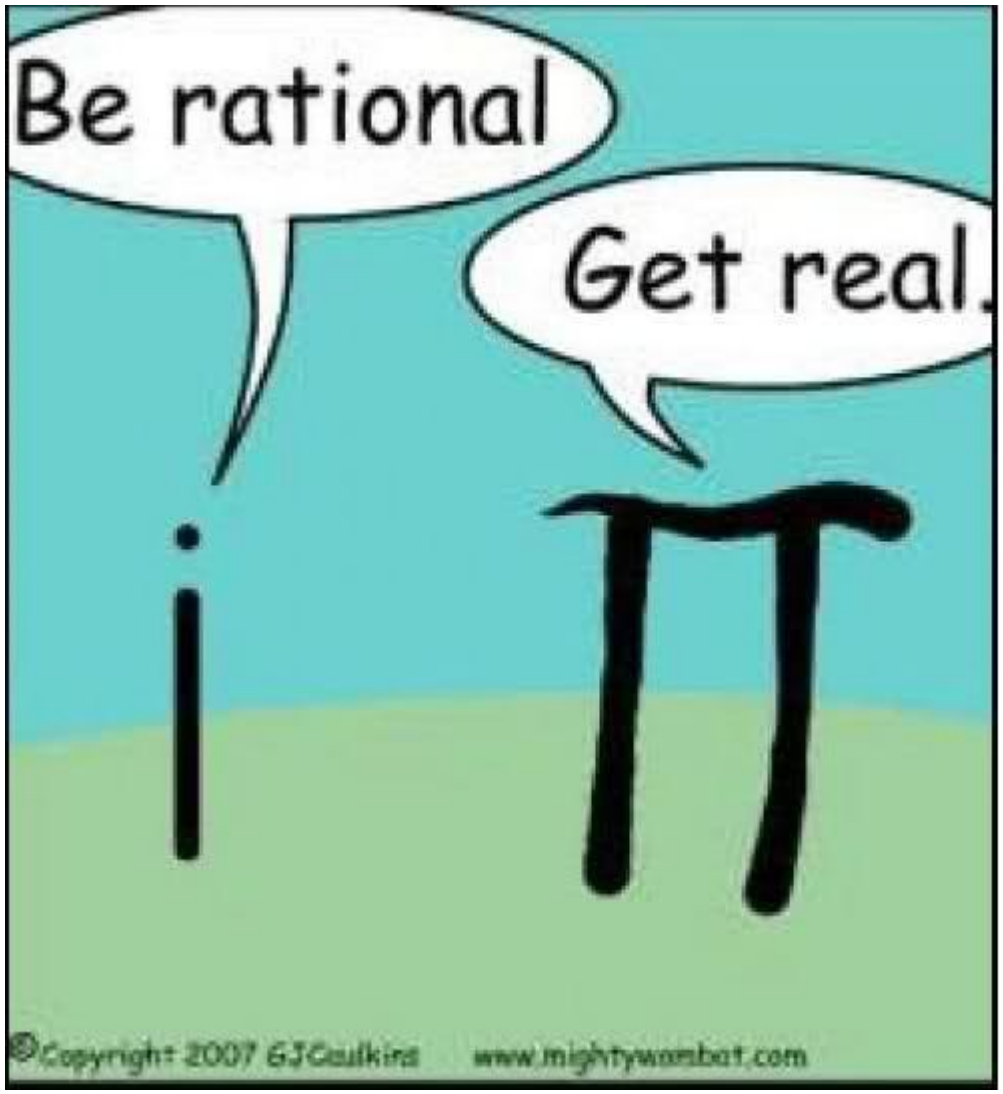
Real Numbers

- Includes all numbers except numbers that involve the square root of negatives
- That is, real numbers include:
  - All integers
  - All rational numbers
  - All irrational numbers

Complex Numbers

- The set of complex numbers are numbers that contain both a real part and an imaginary part
- These are often written in the form:

- Where is defined as , i.e. the square root of -1.
- We will not be doing anything with complex numbers in this module, but those of you who do the Graphics modules in the second year may encounter them (in areas like using Quaternions).


Be rational
Get real.
i
π
©Copyright 2007 GJCaulkins
www.mightywombat.com

Representing Numbers in a Computer

- One of the first issues we need to deal with is how we represent numbers in a computer
- You will see that there are many different ways in which a number can be stored, depending on the *type*
- In this lecture, we will start by looking at how we can represent positive integers.
- Next Thursday we will look at how negative numbers and non-integer numbers can be represented.

The Base Value of Number Systems

- Humans work in base 10 (known as decimal)
- Computers work in base 2 (binary)
- Why?

273 in Binary

2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
256	128	64	32	16	8	4	2	1
1	0	0	0	1	0	0	0	1

273 decimal = 100010001 binary

Converting from Decimal to Another Base

1. Divide the decimal number by the new base
2. Write down the remainder of the division to the left of any previous digits in the answer
3. Replace the decimal number with the quotient (the result of the division)
4. If the quotient is not zero, go back to step 1

Converting from Decimal to Binary

1. Divide the decimal number by 2
2. Write down the remainder of the division (either 0 or 1) to the left of any previous digits in the answer
3. Replace the decimal number with the quotient (the result of the division)
4. If the quotient is not zero, go back to step 1

Converting 273 to binary

Calculation	Binary Result
273 / 2 = 136 remainder 1	1
136 / 2 = 68 remainder 0	01
68 / 2 = 34 remainder 0	001
34 / 2 = 17 remainder 0	0001
17 / 2 = 8 remainder 1	10001
8 / 2 = 4 remainder 0	010001
4 / 2 = 2 remainder 0	0010001
2 / 2 = 1 remainder 0	00010001
1 / 2 = 0 remainder 1	100010001

Converting 218 Decimal to Binary

Calculation	Binary Result
218 / 2 = 109 remainder 0	0
109 / 2 = 54 remainder 1	10
54 / 2 = 27 remainder 0	010
27 / 2 = 13 remainder 1	1010
13 / 2 = 6 remainder 1	11010
6 / 2 = 3 remainder 0	011010
3 / 2 = 1 remainder 1	1011010
1 / 2 = 0 remainder 1	11011010

Converting Binary to Decimal

- Simply add up the values at each position in the number
- For example

1 1 1 0 0 1 1 1 0

Converting Binary to Decimal

- Simply add up the values at each position in the number
- For example

256	128	64	32	16	8	4	2	1
1	1	1	0	0	1	1	1	0

Converting Binary to Decimal

- Simply add up the values at each position in the number
- For example

256	128	64	32	16	8	4	2	1
1	1	1	0	0	1	1	1	0

256 + 128 + 64 + 0 + 0 + 8 + 4 + 2 + 0

Converting Binary to Decimal

- Simply add up the values at each position in the number
- For example

256	128	64	32	16	8	4	2	1
1	1	1	0	0	1	1	1	0

256 + 128 + 64 + 0 + 0 + 8 + 4 + 2 + 0

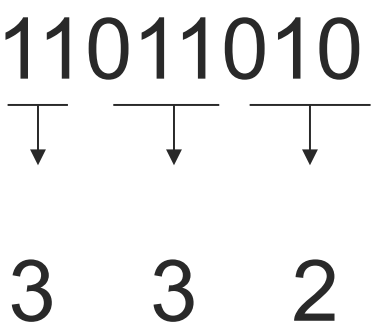
= 462

Octal

- Writing long strings of binary digits can be tedious and error prone, so using other bases that are multiples of two are commonly used to simplify things
- One that can be used is base 8 – Octal
- Converting a binary number to Octal is easy

Converting 11011010 to Octal

- Group into sets of 3 binary digits, starting from the right.
- Convert each set of 3 binary digits into its octal equivalent



Octal	Binary
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Hexadecimal

- A more common way these days of representing long streams of binary digits is to use hexadecimal (hex for short). This is base 16.
- The problem with using base 16 is that we need more digits – 0 to 9 is not enough.
- Therefore, we use the letters A to F as well.

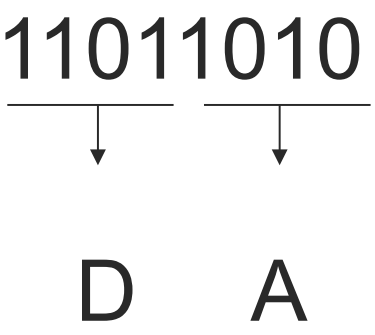
Hexadecimal

Decimal	Binary	Hex
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7

Decimal	Binary	Hex
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

Converting 11011010 to Hexadecimal

- Group into sets of 4 binary digits, starting from the right.
- Convert each set of 4 binary digits into its hexadecimal equivalent



Common Terminology

- Bit
- 4 bits in a nybble (or nibble)
- 8 bits in a byte
- Usually 2 or 4 bytes in a word
- 1,024 bytes = 1 Kilobyte (1 KB)
  - $1024 = 2^{10}$
- 1,048,576 bytes = 1 Megabyte (1 MB)
  - $1{,}048{,}576 = 2^{20}$

Summary

- Today you have been given an introduction to binary, octal and hexadecimal.
- You should make yourself very familiar with these number systems as you will come across them a lot throughout your course
- A formative test has been provided on Course Resources (in the same location as the slides for this lecture). This is not graded, but will enable you to test your own knowledge – and will give you an idea of the sort of questions that could appear on the exam at the end of the module. We will also use these tests as a measure of your engagement in the module.
- Next Thursday we will look at how we can use binary to represent different types of number in a computer.