Do you speak binary?

Coding Basics
First homework is being graded, grade to be posted by the end of the week
Revisit homework policy: late submission, request regrading etc.
Worst quiz and worst homework grade to be dropped.
However, use this policy wisely!
A few of you (~4) have not submitted your 1st homework, if you missed the homework submission because you registered for this class at a very late time, please talk to me after class.
How many of you have programming experience prior to taking CS 1060?

Important: think about how you have progressed over the course of this class.
COMPUTATIONAL THINKING
Computational Thinking

inputs ➔ algorithms ➔ outputs
Think like a computer scientist!

PROBLEM SOLVING
Binary, Number Bases & converting between bases
Binary
0, 1
Decimal
0, 1, 2, 3, ..., 9
Why binary for computers?

- Computer use binary – digits 0 and 1 – to store data
- A binary digit, or bit, is the smallest unit of data in computing
- Circuits in a computer’s processor are made up of transistors
- The digits 1 and 0 reflect the on and off states of a transistor
- Computer programs get translated into binary machine code for a processor to execute
Advantages of using binary

Claude Shannon, Bell Lab, 1948 paper: “A Mathematical Theory of Communication”

- Binary devices are simple and easy to build: e.g. digital calculator
- Binary signals are unambiguous (noise immunity).
- Flawless copies can be made of binary data.
- Anything that can be represented with some sort of pattern can be represented with patterns of bits.

Credit: Bradley Kjell
Decimal

\[1 \times 100 + 2 \times 10 + 5 \times 1\]
binary

1 + 0 + 0 = 4 (decimal)
binary

\[ 4 \quad 2 \quad 1 \]

\[ 0 \quad 0 \quad 0 \]
binary

0 0 0 1
binary

0 1 0
binary

1 0 0
binary

4 2 1

1 0 1
binary

4 2 1

1 1 0
binary
Algorithm: base-2 to base-10
What number does 10010110 in base 2 represent?

\[
\begin{array}{cccccccc}
2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \\
128 & 64 & 32 & 16 & 8 & 4 & 2 & 1 \\
1 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
\end{array}
\]

\[1 \times 128 + 1 \times 16 + 1 \times 4 + 1 \times 2 = 150\]
What number does 10010110 in base 2 represent?

Using 0-based indexing

<table>
<thead>
<tr>
<th>$2^7$</th>
<th>$2^6$</th>
<th>$2^5$</th>
<th>$2^4$</th>
<th>$2^3$</th>
<th>$2^2$</th>
<th>$2^1$</th>
<th>$2^0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>64</td>
<td>32</td>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>D0</td>
<td>D1</td>
<td>D2</td>
<td>D3</td>
<td>D4</td>
<td>D5</td>
<td>D6</td>
<td>D7</td>
</tr>
</tbody>
</table>

$1 \times 128 + 0 \times 64 + 0 \times 32 + 1 \times 16 + 0 \times 8 + 1 \times 4 + 1 \times 2 + 0 \times 1 = 150$

$D_0 \times 2^{(7-0)} + D_1 \times 2^{(7-1)} + \ldots + D_i \times 2^{(7-i)} + \ldots + D_7 \times 2^{(7-7)} = 150$
What number does 10110 in base 2 represent?

Using 0-based indexing

<table>
<thead>
<tr>
<th>$2^4$</th>
<th>$2^3$</th>
<th>$2^2$</th>
<th>$2^1$</th>
<th>$2^0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>8</td>
<td>4</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

$D_0 \times 2^4 + D_1 \times 2^3 + D_2 \times 2^2 + D_3 \times 2^1 + D_4 \times 2^0$

$D_i \times 2^{(4-i)}$

$1 \times 16 + 0 \times 8 + 1 \times 4 + 1 \times 2 + 0 \times 1 = 22$

$D_0 \times 2^{(4-0)} + D_1 \times 2^{(4-1)} + D_2 \times 2^{(4-2)} + D_3 \times 2^{(4-3)} + D_4 \times 2^{(4-4)}$
**Algorithm: base-2 to base-10**

An **algorithm** is a precise set of steps to solve a problem

1. **Input:** a binary number with digits $D_0 D_1 D_2 D_{n-1}$.
2. **Initialization:** set $Sum = 0$, $i = 0$
3. **While** ($i$ is less than the number of digits)
   a. Add $D_i \times (2^{(n-1-i)})$ to $Sum$
   b. Increment $i$
4. **Output** $Sum$
The corresponding Python code

D=raw_input('Enter binary # to be converted: ')
n=len(D); sum=0; i=0
while (i<n):
    sum=sum+int(D[i])*2**(n-i-1)
    i=i+1
print 'The decimal # of the given binary # is', sum

http://www.tutorialspoint.com/execute_python_online.php
The corresponding Python code explained:

```python
D = raw_input('Enter binary #: ')  # raw_input([prompt message]) is a build-in function: it reads a line from input, converts it to a string and returns it.

n = len(D); sum = 0; i = 0  # initialization, len([string]) another build-in function, it returns the length of an object

while (i < n):  # while loop statement
    sum = sum + int(D[i]) * 2 ** (n - i - 1)  # summing up, int([number/string]) returns an integer object from a number or string
    i = i + 1  # increment

print 'The decimal # of the given binary # is', sum  # print both string and number, print the converted decimal #
```

https://docs.python.org/2/library/functions.html#
A SIMPLER VERSION USING BUILD-IN FUNCTIONS

```python
binary = raw_input('Enter binary #: ')
decimal = int(binary, 2)
print 'The decimal # of the given binary # is', decimal
```
Algorithm: base-10 to base-2
Algorithm by examples

Converting $16$ (base-10) to base-2: $10000$
Converting $27$ (base-10) to base-2: $11011$
Converting 16 (base-10) to base-2: 10000
Converting 27 (base-10) to base-2: 11011
Algorithm: base-10 to base-2

1. **Input:** a decimal number \( \text{dec} \)
2. **Initialization:** set \( s = 0, \ i = 1 \)
3. **While** \( (\text{dec} > 0) \)
   a. remainder = \( \text{dec} \mod 2 \)
   b. divide \( \text{dec} \) by 2
   c. append remainder to the left of \( s \), i.e., multiplying by 10 and add to \( s \)
4. **Output** \( s \)
In Python

dec=input("Enter decimal # to be converted: ")
s=0; i=1
while dec>0:
    remainder=dec%2
    dec=dec/2
    s=s+(i*remainder)
    i=i*10
print s
print "The binary of the given # is ",s
1. What is 1000 in base 2 converted to base 10?

2. Convert 36 in base 10 to base 2.
Python programming is fun and productive
Quiz 2: Binary and Decimal
Think beyond Binary
Quantum Computing

- Theoretical computation systems: quantum computers, use quantum-mechanical phenomena to perform operations on data
- Different from digital electronic computers based on transistors.
- Uses quantum bits (qubits), which can be in superpositions of states: e.g. linear combination of basic states of particles
- Quantum Superposition: any 2+ quantum states can be added together and the result will be another valid quantum state
- Quantum Turing machine or the universal quantum computer
- Non-deterministic and probabilistic
- Further reading: https://en.wikipedia.org/wiki/Quantum_computing
Quantum Computing

- A quantum bit corresponds to a single electron in a particular state. Using the trajectories of an electron through two closely spaced channels for encoding.
- In principle, two different states are possible: the electron either moves in the upper channel or in the lower channel – a binary system.
- However, a particle can be in several states simultaneously, that is, it can quasi fly through both channels at the same time.
- These overlapping states can form an extensive alphabet of data processing.

Quantum computer science

Further reading: [http://qist.lanl.gov/qcomp_map.shtml](http://qist.lanl.gov/qcomp_map.shtml)  
THANKS!

Any questions?

You can find me at beiwang@sci.utah.edu

http://www.sci.utah.edu/~beiwang/teaching/cs1060.html
Credits

Special thanks to all the people who made and released these awesome resources for free:

- Presentation template by SlidesCarnival
- Photographs by Unsplash