Objectives

- Computer's representations of data types

Aside: String Format with Variable

- Question: Can we use a variable instead of a number for the width?
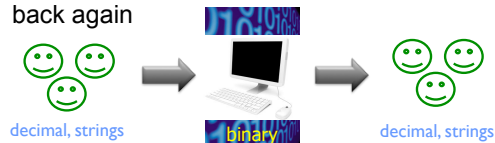- Answer: Yes!
  - But tread carefully

widthvar.py

Representations of Data

- Computer needs ways to represent different types of data
  - Eventually, all boils down to 1s and 0s
- Computer needs to translate between what humans know to what computer knows and back again

decimal, strings   binary   decimal, strings

Seems like a divergence on strings but just wait...

Decimal Representations

- Decimal is base 10
- Digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Each *position* in a decimal number represents a power of 10

Decimal Representations

- Decimal is base 10
- Digits: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9
- Each *position* in a decimal number represents a power of 10
- Example: 54,087

5	4	0	8	7
10^4	10^3	10^2	10^1	10^0

- $= 5*10^4 + 4*10^3 + 0*10^2 + 8*10^1 + 7*10^0$
- $= 5*10,000 + 4*1000 + 0*100 + 8*10 + 7*1$

Number Representations

Characteristic	Decimal	Binary
Base	10	2
Digits	0, 1, 2, 3, 4, 5, 6, 7, 8, 9	0, 1
Position represents	Power of 10	Power of 2

- Binary: two values (0, 1)
  - Like a light switch (either **off** or **on**) or booleans (either True or False)
- 0 and 1 are *binary digits* or **bits**
  - 64-bit machine: represents numbers (and other data) with 64 bits

Binary Representation

- Binary number: 1101

1	1	0	1
2^3	2^2	2^1	2^0

- $= 1*2^3 + 1*2^2 + 0*2^1 + 1*2^0$
- $= 1*8 + 1*4 + 0*2 + 1*1$
  - Decimal value: 13

Practice: what is the decimal value of the binary number **10110**?

Binary Representation

- Binary number: 10110

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

- $= 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 0*2^0$
- $= 1*16 + 0*8 + 1*4 + 1*2 + 0*1$
  - 22

Generalize this process into an algorithm...

Algorithm: Converting Binary → Decimal

Accumulator design pattern

1. Read in the binary number as a string
   - The starting exponent will be the length of the string-1
2. Initialize the result to zero
3. For each bit in the binary number
   - Multiply the bit by the appropriate power of 2
   - Add this to the result
   - Reduce the exponent by 1
4. Display the result

Implement algorithm
binaryToDecimal.py

Algorithm: Converting Decimal → Binary

1. Read in the decimal as an integer
2. Initialize the result to the empty string
3. Repeat until the decimal is 0:   We don't know how to implement yet
   - result = str(decimal % 2) + result
   - decimal = decimal // 2
4. Display the result

Try out algorithm with 22

String Representations

- A **string** is a *sequence* of characters
- Each character is stored as a binary number
- **ASCII** (American Standard Code for Information Interchange) is one standard encoding for characters
  - Limitation: ASCII is based on the English language
  - Cannot represent other types of characters
- Unicode is a new standard

ASCII Questions

- Lowercase letters are represented by what range of numbers?

- Uppercase letters are represented by what range of numbers?

- What is the difference between the decimal encoding of 'M' and 'N'?
  - Between 'm' and 'n'?

ASCII Questions

- Lowercase letters are represented by what range of numbers?
  - 97—122
- Uppercase letters are represented by what range of numbers?
  - 65—90
- What is the difference between the decimal encoding of 'M' and 'N'?
  - Between 'm' and 'n'?
  - 1

Translating to/from ASCII

- Translate a character into its ASCII numeric code using **built-in function** `ord`
  - `ord('a') ==> 97`
- Translate an ASCII numeric code into its character using **built-in function** `chr`
  - `chr(97) ==> 'a'`

`ascii_table.py`
`ascii.py`

Encryption

- Process of encoding information to keep it secure
- One technique: Substitution Cipher
  - Each character in message is replaced by a new character

Caesar Cipher

- Replace with a character X places away
  - X is the **key**
- Julius Caesar used technique to communicate with his generals
- "Wrap around"
- Write program(s) to do this in next lab

Caesar Cipher

- Using the ASCII handout, what would be the encoded messages?

Message	Key	Encoded Message
apple	5	
zebra	5	
the eagle flies at midnight	-5	

Caesar Cipher

Message	Key	Encoded Message
apple	5	fuuqj
zebra	5	ejgwf
the eagle flies at midnight	-5	ocz zvbgz agdzn vo hdyidbco

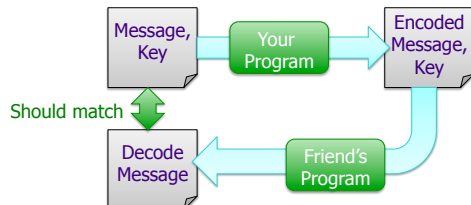What is your algorithm for the encoding process? How would you *decode* an encrypted message?

Next Lab

- Write an encoding/decoding program
  - ➢ Encode a message
  - ➢ Give to a friend to decode

Message, Key → Your Program → Encoded Message, Key

Should match

Decode Message ← Friend's Program

Exam 1 Results

	A	B	C	Total
Average	82%	78%	86%	88%
Median	80%	82%	91%	88%

- Out of 104 points
  - ➢ 108 points possible, plus 6 bonus points

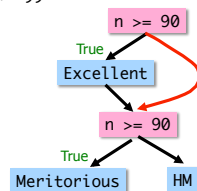- What is a condition?
- "Edit code above"

Awards

```
n = eval(input("Enter a score: "))
if n >= 90:
    print("Excellent")
if n > 70:
    print("Meritorious")
else:
    print("Honorable Mention")
```

n >= 90 → True → Excellent → n >= 90 → True → Meritorious / HM

Output when n = 95?

2nd if needs to be an `elif`

Number of Delegates

```
state = input("Enter the state: ")
delegates = eval(input("Enter the number of delegates: "))

print()
print("Allocation of", state, "Delegates:")
print()

print("Popular Vote Pct", "Number of Delegates")

for x in range(5, 101, 5):
    print(x, delegates * x/100)
```

Choosing a Candidate

```
years = eval(input("Enter the candidate's years of
experience: "))
pct = eval(input("What percentage of the issues do you
and the candidate agree on? "))

if years < 5 or years > 20:
    print("Do not vote for this candidate")
elif pct < 80:
    print("Do not vote for this candidate")
else:
    print("Vote for this candidate")
```

Other correct possibilities

Grading

- (38%) Programming projects
- (30%) Two hourly exams
- (20%) A comprehensive final exam
- (7%) Writeups and discussions of CS-related issues
- (5%) Participation and attendance

Looking Ahead

- Lab tomorrow
  - ➢ "That was a good review!"
  - ➢ Save yourself time and frustration if you review the "bag of tricks" we learned in the last week
- Broader Issue
  - ➢ Bug comparison
- More powerful data structures