COMPUTER SCIENCE

LESSON 25+26, TUESDAY DEC 12TH + THURSDAY DEC 14TH

WWW.CITA.UTORONTO.CA/~WOODFORD

WOODFORD@CITA.UTORONTO.CA

THIS WEEK IN CS AND STEM

- Google's AI AlphaZero dominates the chess world – in less than 4 hours
  - https://futurism.com/4-hours-googles-ai-mastered-chess-knowledge-history/
- Time flowing backwards?! Not quite, but maybe as close as we can get.
  - https://futurism.com/scientists-experimentally-demonstrate-reversal-arrow-time/
- Also this week: A new black metal album dropped – it's creator is an AI, Unanimous' swarm AI has correctly predicted TIME's person of the year twice in a row

INPUT AND OUTPUT FILES: SUMMARY

- A lot of the data we've been using thus far has been input from the user on the command line or hard-coded into the file. This isn't reasonable for large data sets or complicated entry methods.
- It's typically easier for users to create an input file that contains all the data needed by a function.
- f. = open.("filename.txt","r"), f.read(), for line in f: line.split()
- f = open("filename.txt","w"), f.write()
- f.close()

ARGPARSE

- A bit more formalism for this one. Take a look at the reference material for this on the python docs page for full details.

Import argparse

parser = argparse.ArgumentParser(description = “Get an input file name”)

parser.add_argument(“—i_filename”, required = True)

args.parser.parse_args()

- Lets take a look in Lecture24_25_argparse.py

EXAM REVIEW

- Note it's only for Term 1 (ie. units A, B, C). Pay attention to how much class time we spent on certain topics as well as how much assignment time we spent on certain topics.
  - Top time spenders: unit testing, logic, AI /CS and society, pre/post conditions
  - Middle time spenders: recursive functions+pitfalls, finite data representation, defining functions
  - Lowest time spenders: type conversion, bash, dictionaries, integration+regression testing

EXAM REVIEW: UNIT TESTING

- The definition was to test each “unit” of a code individually to check for errors and inconsistencies. What could a “unit” be?

- What are some ways of unit testing?

EXAM REVIEW: UNIT TESTING

- The definition was to test each “unit” of a code individually to check for errors and inconsistencies. What could a “unit” be?
  - Functions, typically (ie. def func(args):… return)
- What are some ways of unit testing?
  - Pass in values that you know what the result should be and compare
  - Pass in a list to see if it changes correctly
  - Insert print statements and additional return statements
  - Test from ipython line OR from a runnable script

EXAM REVIEW: PRE/POST CONDITIONS

- For each function, there should be a pair of pre and post conditions. A pre condition is a statement of what must be true before the function, and the post condition is a statement of what must be true after the function. A pre condition can include what kind of arguments to include, but neither need or should include HOW the function operates.
- You can include these as comments or docstrings (preferred)

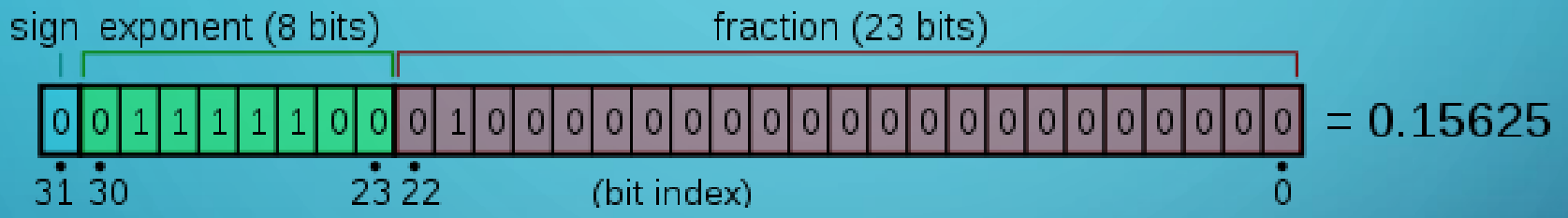
EXAM REVIEW: FINITE DATA REPRESENTATION

- Let's review binary quickly first:
  - Decimal: Base 10, includes 0,1,2,3,4,5,6,7,8,9
  - Binary: Base 2, include 0,1
  - Each digit represents (in not just these, but all number systems) the value of the digit multiplied by the base to the exponent of the number of the digit space.
    - Ex. 19 -> 1*10^1 + 9*10^0
    - Ex. 1001-> 1*2^3+ 0*2^2 + 0*2^1 + 1*2^0

- Note that things in computer memory are typically stored in binary. But surely there's a better way to store them than in straight up binary with many digit places.

LIMITATIONS OF FINITE DATA REPRESENTATION

- There are 3 parts to a float: the sign, the exponent, and the fraction.



- The sign indicates the sign of the float, and the exponent is the power of 2 that the entire float is multiplied by. The fraction is a little harder, where each bit is now $1/2^{\text{bitvalue}}$. Other ways to write this include:
  - $(-1)^{b_{31}} \times (1.b_{22}b_{21}\ldots b_0)_2 \times 2^{(b_{30}b_{29}\ldots b_{23})_2 - 127}$
  - $(-1)^{sign} \times (1 + \sum_{i=1}^{23} b_{23-i}2^{-i} \times 2^{e-127})$

EXAM REVIEW: OTHER KINDS OF TESTING

- Integration testing: testing more than one unit together, typically for argument passing errors
- Regression testing: testing a script that once worked and has sense been updated – usually by unit/integration testing each successive version until running into the error in quesiton

EXAM REVIEW: TYPE CONVERSIONS

- Int(), float(), bool(), str()
- Which kinds work together for conversions and which one don't? Which pairings might be the most useful?

EXAM REVIEW: HISTORY OF CS

- We covered a fair bit on this. It might be best to focus on what we spent the most class time on (where did our discussions lead us?) and what's the most relevant to the course (ie. Alan Turing and his tests for AI may be a tad more relevant than who made the first working computer mouse)
- See Lecture 1(last 3 slides), Lecture 2, Lecture 3
  - Note that these also include discussion on environmental affects and the impact on society.

REFERENCES

- https://docs.python.org/3/library/filesys.html
- https://docs.python.org/3/library/functions.html#open
- https://docs.python.org/3/library/argparse.html
- https://docs.python.org/3/tutorial/inputoutput.html