

# Parker, Prokofiev & Python

Programming is musicianship

Dwight Browne

[dwight@dwightjbrowne.com](mailto:dwight@dwightjbrowne.com)

<https://github.com/dbrowne/Presentations>

# Charlie Parker

Jazz Musician

Alto Saxophone



# Sergei Prokofiev

Classical Pianist  
Composer Conductor





# Autodidact & OTJ

(on the Job Training)

3 Years of  
Woodshedding

Daily Practice

Nightly Gigs



# Conservatory Trained

Formal and established  
pedagogy







Mastery of:  
Instrument  
Technique  
Theory

Innovative  
Compositions



# Orthogonal Approach

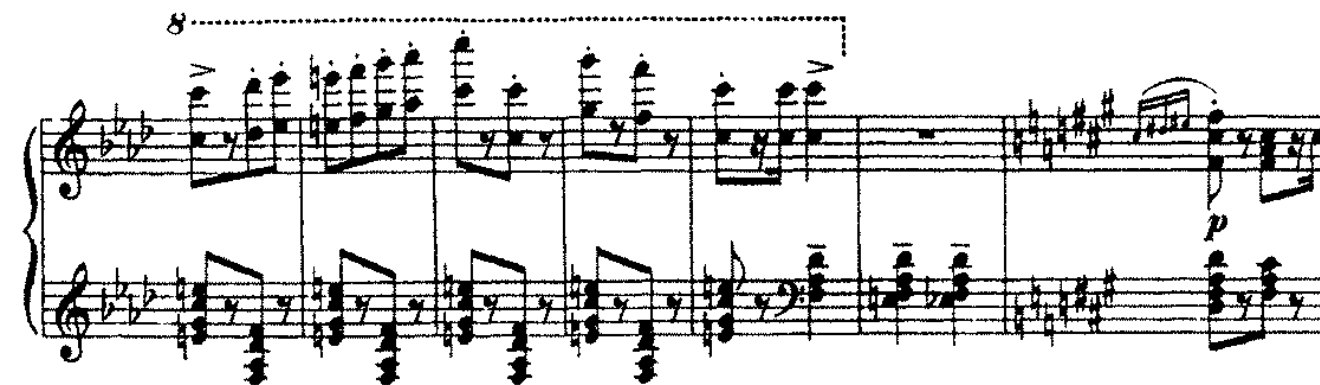
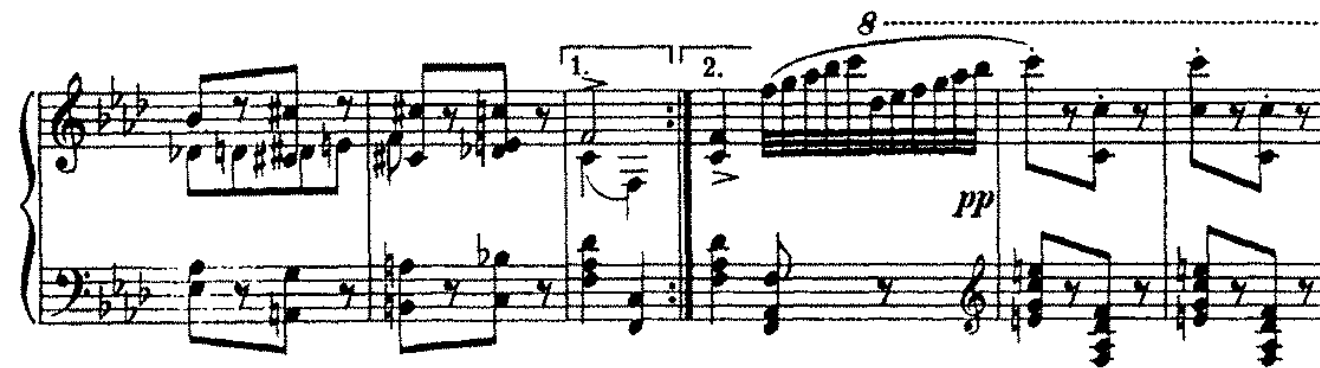
# Mastery of Language



Sergei Prokofiev  
Ten Pieces

Allegro

1. March



Explicit  
Instructions

Sergei Prokofiev  
Ten Pieces  
1. March

Allegro

The image displays a musical score for Sergei Prokofiev's '1. March' from his 'Ten Pieces' for piano. The score is written for piano (p) and includes dynamic markings such as *pp* (pianissimo) and *p* (piano). The tempo is marked 'Allegro'. The score is divided into four systems, each with a treble and bass staff. The first system begins with a *pp* marking. The second system features a first ending (1.) and a second ending (2.) marked with a repeat sign and a first ending bracket. The third system includes a first ending (1.) and a second ending (2.) marked with a repeat sign and a first ending bracket. The fourth system concludes the piece. Red arrows point from the text 'Explicit Instructions' to specific markings in the score: one arrow points to the *pp* marking in the first system, and another arrow points to the first ending bracket in the third system.

Perform as written

# Formal Specification



318

(BOP)

# PASSPORT

-CHARLIE PARKER

Handwritten musical score for "PASSPORT" by Charlie Parker. The score is in B-flat major, 4/4 time, and consists of three staves of music. The first staff contains measures 1-4 with chords Bbmaj7, C-7, F7, D-7, G7#5, C-7, and F7. The second staff contains measures 5-8 with chords F-7, Bb7, Ebmaj7, Eo7, Bbmaj7, G7b9, C-7, and F7. The third staff contains measures 9-12 with chords Bbmaj7, D7, G7, and C7. The score includes various musical notations such as eighth notes, quarter notes, and rests.

# Instructions

# Guidelines

318  
(BOP)

## PASSPORT

-CHARLIE PARKER

The musical score is written in B-flat major (two flats) and 4/4 time. It consists of three staves of music. The first staff begins with a treble clef, a key signature of two flats (B-flat major), and a 4/4 time signature. The melody is written in eighth and quarter notes. The second staff continues the melody, featuring a repeat sign and a second ending. The third staff concludes the piece. Chord symbols are written above the notes: Bbmaj7, C-7, F7, D-7, G7#5, C-7, F7, F-7, Bb7, Ebmaj7, Eo7, Bbmaj7, G7b9, C-7, F7, Bbmaj7, D7, and G7. The title 'PASSPORT' is written in large, bold, capital letters, and the composer's name '-CHARLIE PARKER' is written in smaller capital letters to the right.

Perform as suggested

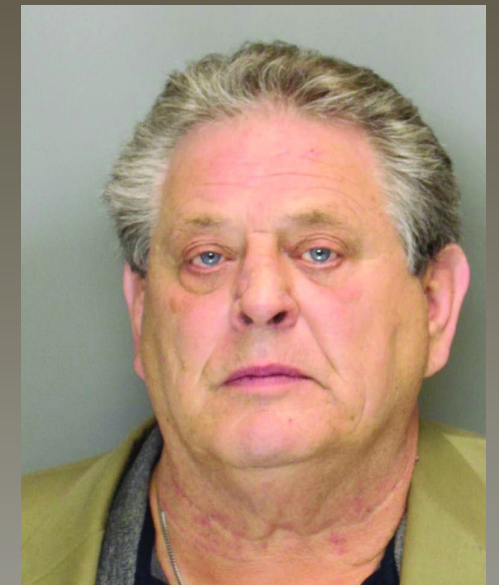
# General Request



# Open to interpretation\*

\* If you know what you are doing

I Want Dwight....



Uh.....



# Classical: Final Class

Jazz: Extensible class



Jazz: S.O.L.I.D

Not Ashford and Simpson\*



\* Solid (as a rock)

S : Single Responsibility

O : Open/Closed

L : Liskov Substitution

I : Interface Segregation

D : Dependency Inversion

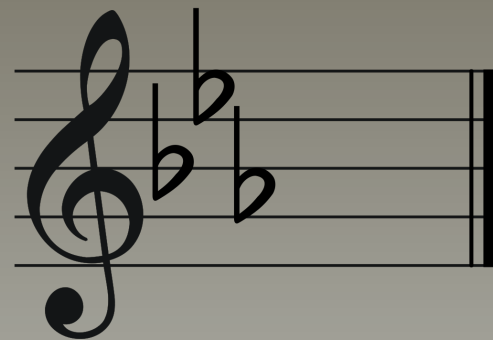
1, b3, 5, 7

@logging

def some function(self):

pass

\_\_slots\_\_





# Contextual Information

# Mastery of Language

# Classical: Etudes

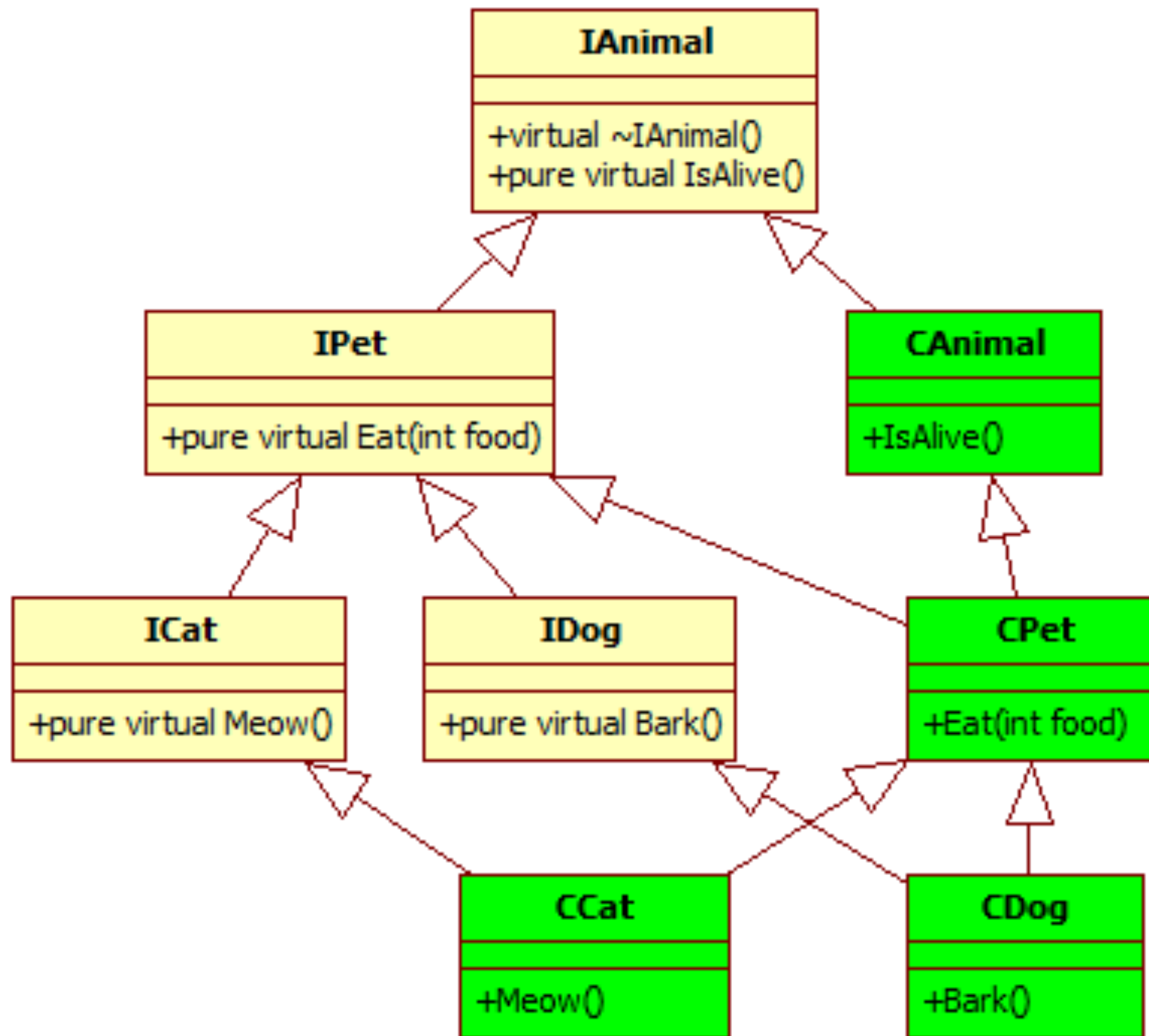
# Jazz: Standards

```
class EmitVisitor(asdl.VisitorBase):
    """Visit that emits lines"""

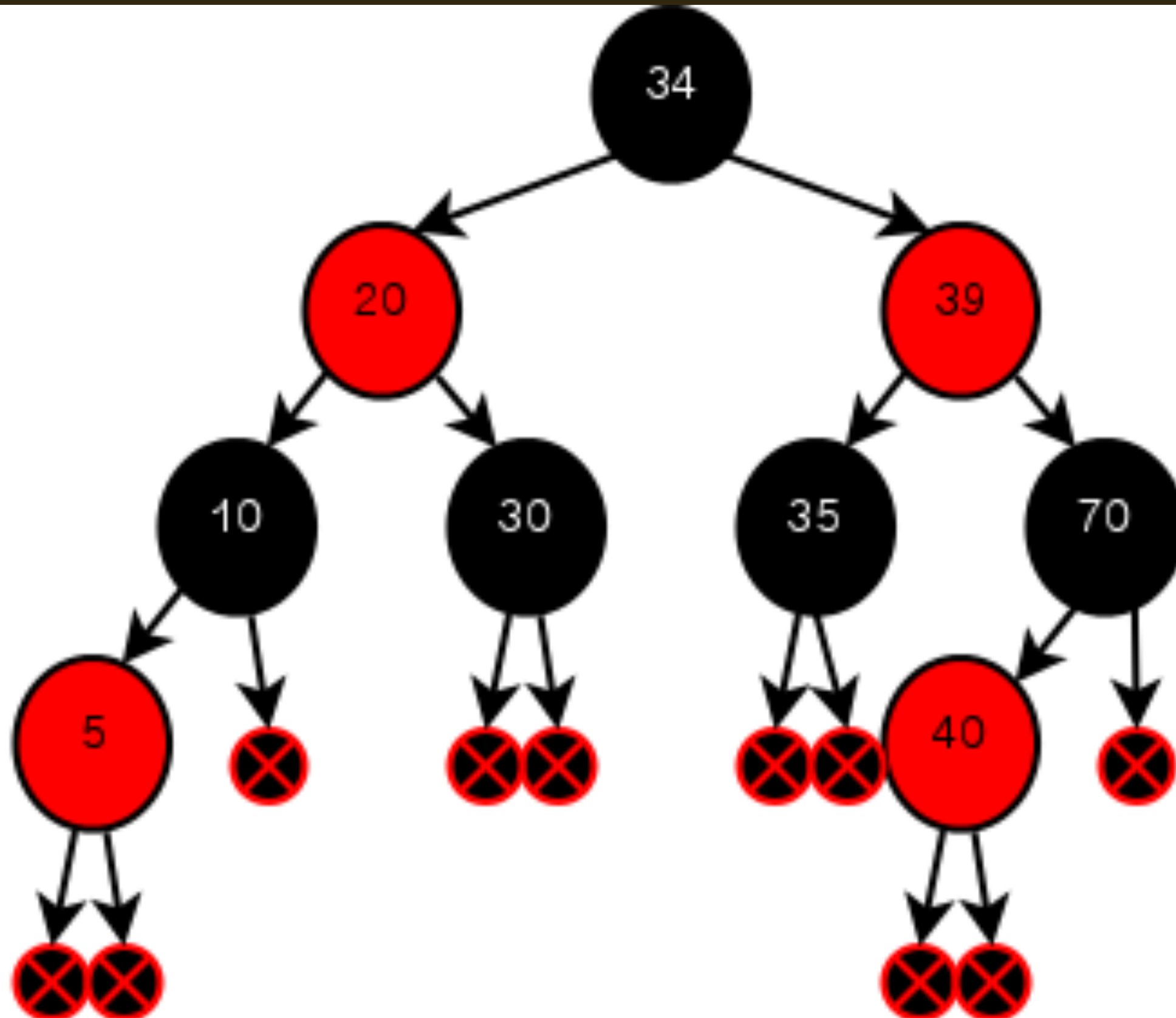
    def __init__(self, file):
        self.file = file
        self.identifiers = set()
        super(EmitVisitor, self).__init__()

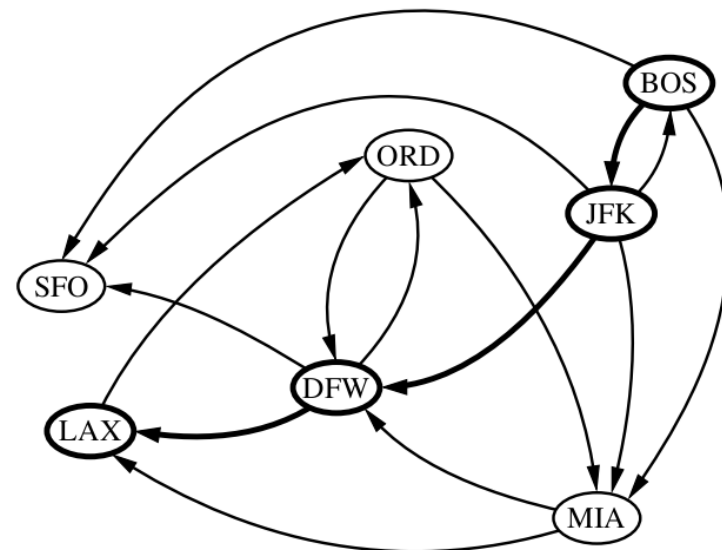
    def emit_identifier(self, name):
        name = str(name)
        if name in self.identifiers:
            return
        self.emit("_Py_IDENTIFIER(%s);" % name, 0)
        self.identifiers.add(name)

    def emit(self, s, depth, reflow=True):
        # XXX reflow long lines?
        if reflow:
            lines = reflow_lines(s, depth)
        else:
            lines = [s]
        for line in lines:
            line = (" " * TABSIZE * depth) + line + "\n"
            self.file.write(line)
```

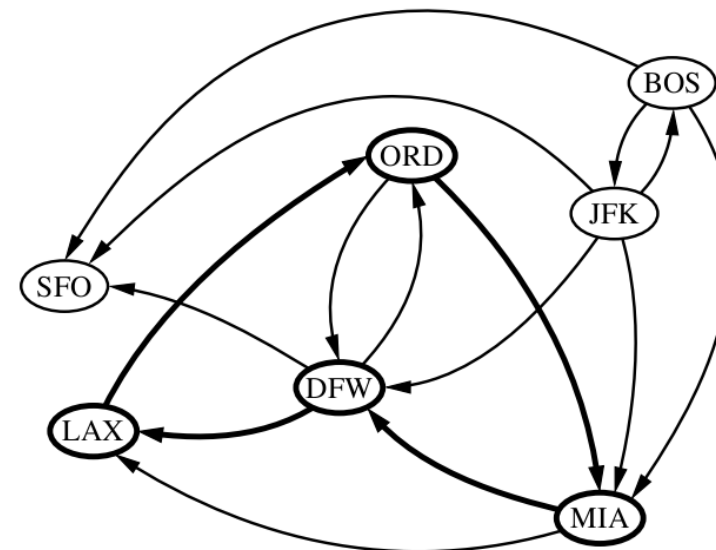




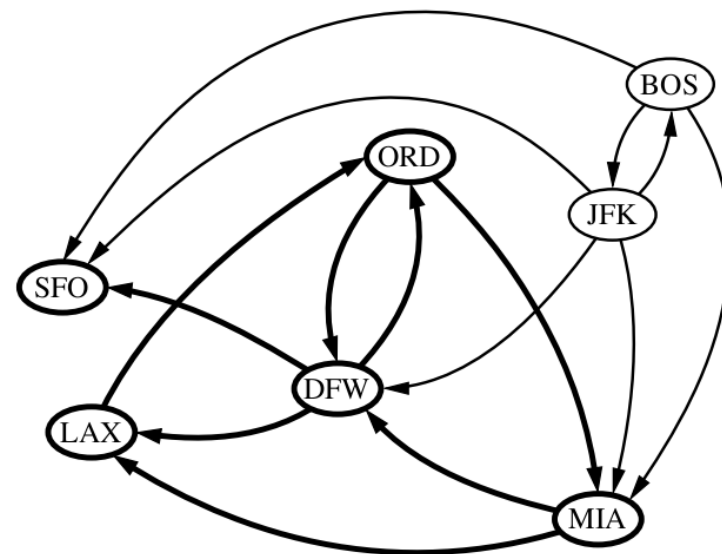




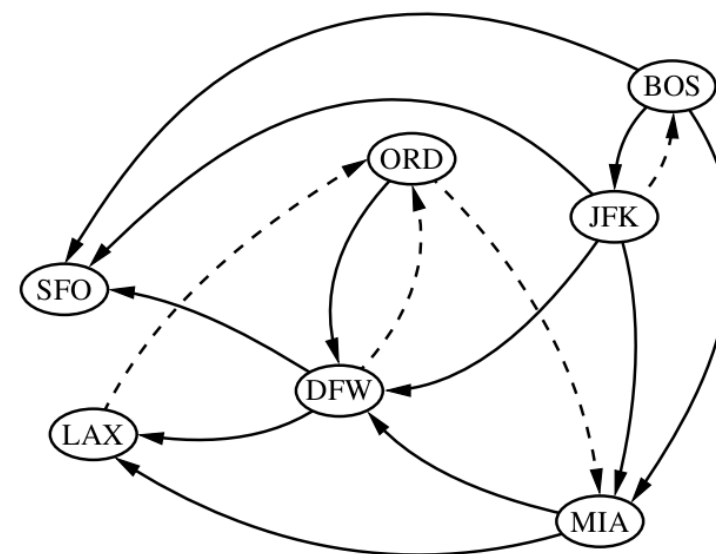
(a)



(b)



(c)



(d)

$$\mathbf{A}^{-1} = \frac{1}{\det(\mathbf{A})} \sum_{s=0}^{n-1} \mathbf{A}^s \sum_{k_1, k_2, \dots, k_{n-1}} \prod_{l=1}^{n-1} \frac{(-1)^{k_l+1}}{l^{k_l} k_l!} \text{tr}(\mathbf{A}^l)^{k_l}$$

# Big O Notation Summary

| Notation       | Type        | Examples   | Description   |
|----------------|-------------|--|---|
| $O(1)$         | Constant    | Hash table access  | Remains constant regardless of the size of the data set   |
| $O(\log n)$    | Logarithmic | Binary search of a sorted table                              | Increases by a constant. If $n$ doubles, the time to perform increases by a constant, smaller than $n$ amount |
| $O(<n)$        | Sublinear   | Search using parallel processing                             | Performs at less than linear and more than logarithmic levels   |
| $O(n)$         | Linear      | Finding an item in an unsorted list                          | Increases in proportion to $n$ . If $n$ doubles, the time to perform doubles                                  |
| $O(n \log(n))$ | $n \log(n)$ | Quicksort, Merge Sort  | Increases at a multiple of a constant   |
| $O(n^2)$       | Quadratic   | Bubble sort  | Increases in proportion to the product of $n*n$   |
| $O(c^n)$       | Exponential | Travelling salesman problem solved using dynamic programming | Increases based on the exponent $n$ of a constant $c$   |
| $O(n!)$        | Factorial   | Travelling salesman problem solved using brute force         | Increases in proportion to the product of all numbers included (e.g., $1*2*3*4...$ )                          |

# Implementation

# Application of Theory



Regular Practice

# Know Your Audience

if x:

vs

if len(x) < 1:

Some will say one is wrong

What to Learn?

Know your business

# Data Structures



# Algorithms

# Computer Architecture

# Python Internals

When to ask questions

You Can Be Incredible Too!

