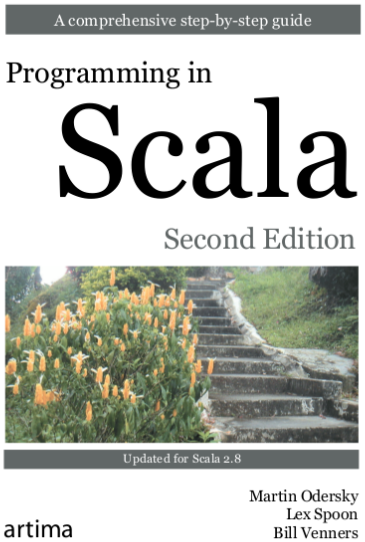
Computer Programming with Scala

Week 4: Sorting Bits

Martin Quinson

December 2015

Introduction

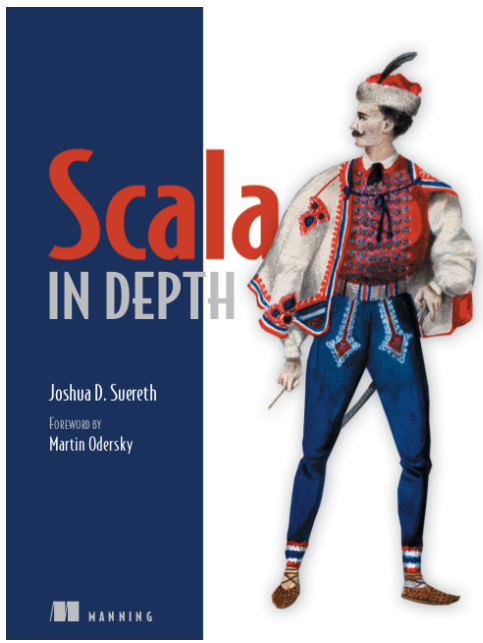


What we saw so far

- We covered the base book (or almost)
- You received the language syntax and content
- Basics, OOP and FP syntax, some stylistic advices
- You know your letters, some know to read/write

Coming next

- How can you become a mature journalist? (~~poet~~)
- Train your programming skill
  - Experience comes from exercising
  - You don't learn swimming or skiing in the books
- Expertise can be (inefficiently) taught
  - Best practices can be presented, not sure if you get it
  - At least, get (exposed to) the vocabulary



Experience is what enables you to recognize a mistake when you make it again.

Design Patterns

What

- Classical patterns that you can find in large programs
  - Seminal work: common patterns from at least 3 large well organised projects
- Good programmer: someone that can detect patterns in someone else code
  - Mean of communication

Why

- Best Practice: Reuse the expertise of the ones before us
- Mean of communication: Put simple words on common concepts

I have this object with some important information and these objects over here need to know when its information changes. These other objects come and go. I'm thinking I should separate out the notification and client registration functionality from the functionality of the object and just let it focus on storing and manipulating its information. Do you agree?

I'm thinking of using the Observer pattern. Do you agree?

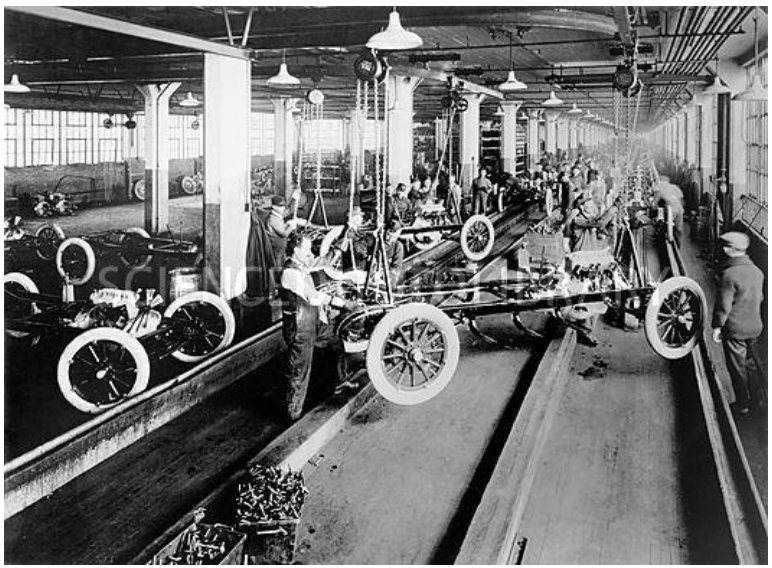
Why Not

- Sometimes boilerplates to circumvent language limitations. Bad idea.
- Lecturing on the 50 Design Patterns is too soporific, let's pick some.

Factory Pattern

What

- Provides an interface for creating an object, encapsulating class instanciation
- Lets subclasses decide which class to instantiate.



```
trait Animal
private class Dog extends Animal
private class Cat extends Animal
object Animal {
  def apply(kind: String): kind match {
    case "dog" => new Dog()
    case "cat" => new Cat()
  }
}
val d = Animal("dog")
```

Why? When?

- Extract complex object creation code
- Select which clase to instanciate
- Cache objects

Discussion

☺ Resembles constructor invocation

☹ Static factory

Singleton Pattern

What:

- Restrict the instantiation to one object
- Provide a global access point to this object

***The Solitary Tree*, Bobbi Jones Jones**

```
object myApp extends App {
  println("Hello World")
}
```

Why? When?

- Have exactly one object of a class

Discussion

- ☺ Integrated into the language
- ☺ Java/C++ $\rightsquigarrow$ boilerplate code

Adapter Pattern

What: Converts an interface into expected interface

Multiple jacks / adaptors, **Mihai Andoni**

```
trait Log {
  def warning(message: String)
  def error(message: String)
}
final class Logger {
  def log(lvl: Level, msg: String) { /* ... */ }
}
implicit class Adapter(logger: Logger) extends Log {
  def warning(msg: String) { logger.log(WARNING, msg) }
  def error(msg: String)   { logger.log(ERROR, msg) }
}

val log: Log = new Logger()
```

Why? When?

- Integrate incompatible classes
- Adapt existing components

Discussion

- ☺ Clear intent, Concise syntax
- ☹ Implicit behavior easily overseen

Decorator Pattern

What: Extends functionality of an object (alternative to subclassing)

Colour Pencils, George Hodan

```
trait OutputStream {
  def write(b: Byte)
  def write(b: Array[Byte])
}
class FileOStream(path: String) extends OutputStream { }
trait Buffering extends OutputStream {
  abstract override def write(b: Byte) {
    // ...
    super.write(buffer)
  }
}
new FileOStream("foo.txt") with Buffering
```

Why? When?

- Extend final classes
- Arbitrarily graph of combination
- Multiple inheritance

Discussion

- ☺ Clear intent, Concise syntax
- ☺ Separation of concern
- ☹ Static binding
- ☹ No constructor parameter

Chain of Responsability

What: Decouples the sender of a request from its receiver

***Things in motion: Colored Domino*, Sigurd Decroos**

```scala
case class Event(source: String)
type EventHandler = PartialFunction[Event, Unit]
val defaultHandler: EventHandler = PartialFunction(_ =>
val keyboardHandler: EventHandler = {
  case Event("keyboard") => /* ... */
}
val mouseHandler(delay: Int) : EventHandler = {
  case Event("Mouse") => /* ... */
}
keyboardHandler orElse mouseHandler(100) orElse defaultHa
```

Why? When?

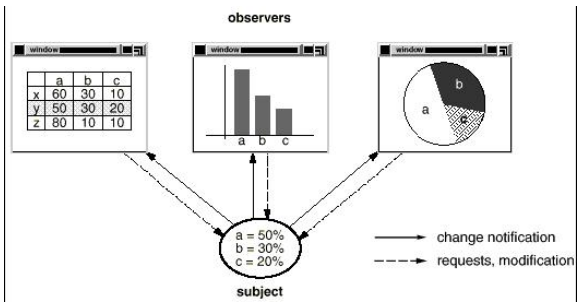
- Request proposed to the chain until handled
- Gives more than one object a chance

Discussion

- ☺ Concise syntax, Built-in logic
- ☹ General-purpose type

Observer Pattern

What: Broadcast object's state changes notifications to dependents




```
trait Observer {
  def notify(changed: Observable)
}
trait Observable {
  private var observers: List[Observer] = Nil
  def addObserver(o:Observer) = observers ::= o
  def fireChange = for (o <- observers) o.notify(this)
}
```

Why? When?

- Set of loosely coupled objects
- Decouple data Model from View

Discussion

☺ (rather) concise syntax

☹ Control flow complex to follow

The Pattern Bestiary (1/3)

Creational Patterns

- Abstract Factory: Factory for building related objects
- Builder: Factory for building complex objects incrementally
- Factory Method: Method in a derived class creates associates
- Prototype: Factory for cloning new instances from a prototype
- Singleton: Factory for a singular (sole) instance

Structural Patterns

- Adapter: Translator adapts a server interface for a client
- Bridge: Abstraction for binding one of many implementations
- Composite: Structure for building recursive aggregations
- Decorator: Extends an object transparently
- Facade: Simplifies the interface for a subsystem
- Flyweight: Many fine-grained objects shared efficiently.
- Proxy: One object approximates another

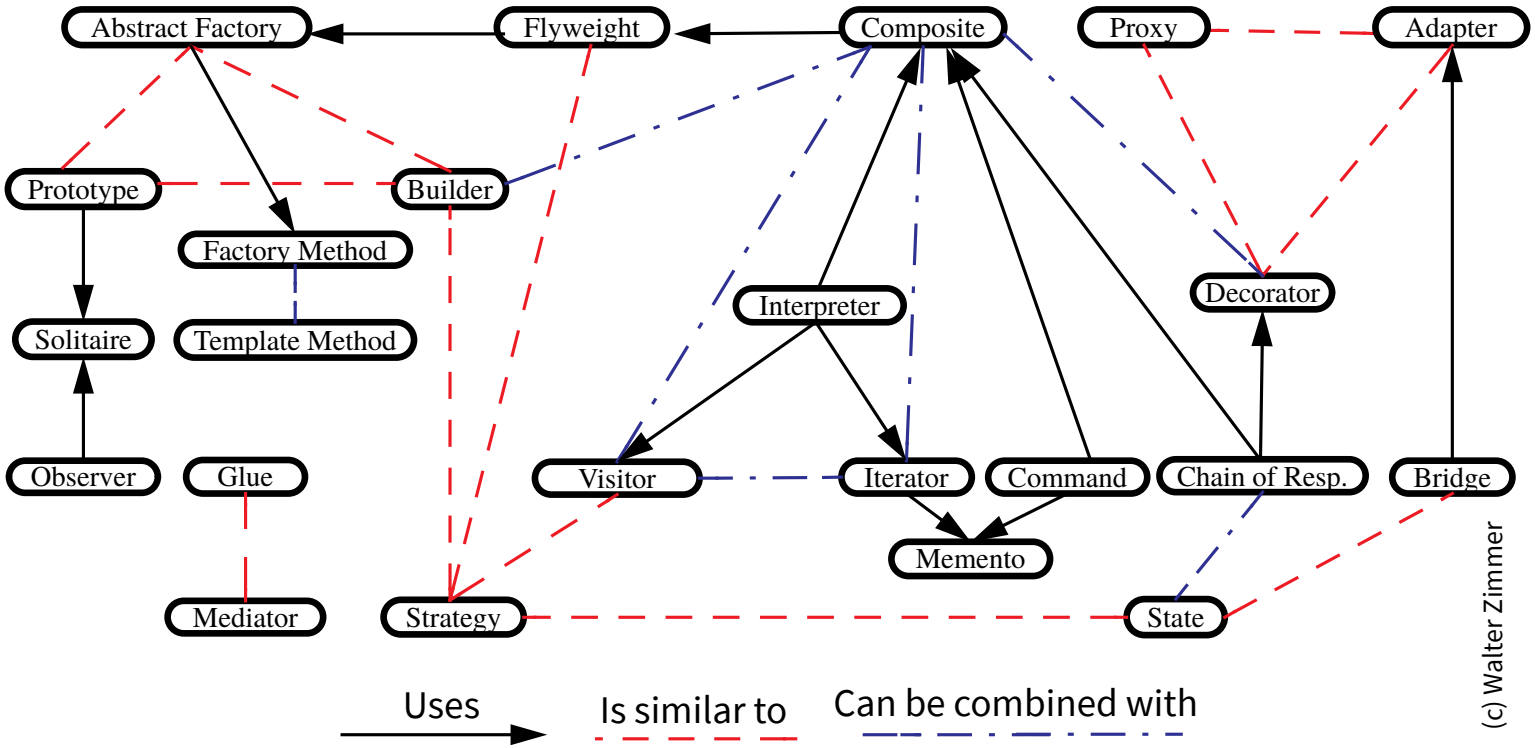
The Pattern Bestiary (2/3)

Behavioral Patterns

- Chain of Responsibility: Request delegated to the responsible service provider
- Command: Request or Action is first-class object, hence storable
- Iterator: Aggregate and access elements sequentially
- Interpreter: Language interpreter for a small grammar
- Mediator: Coordinates interactions between its associates
- Memento: Snapshot captures and restores object states privately
- Observer: Dependents update automatically when subject changes
- State: Object whose behavior depends on its state
- Strategy: Abstraction for selecting one of many algorithms
- Template Method: Algorithm with some steps supplied by a derived class
- Visitor: Operations applied to elements of a heterogeneous object structure

The Pattern Bestiary (3/3)

Patterns Interactions



Conclusion on Design Patterns

- These are only the 23 presented in seminal work. Maaaany more can be found
- Learning them is boring, Knowing them helps building on elder's experience

`https://speakerdeck.com/pavelfatin/design-patterns-in-scala`

Anti-Patterns

- Set of well known bad ideas to common problems
- Bicycle shed: Giving disproportionate weight to trivial issues
- Bleeding edge: Operating with cutting-edge technologies that are still unstable
- Overengineering: Spending time making a project more complex than needed
- Big ball of mud: A system with no recognizable structure
- Interface bloat: Making an interface so powerful that it is difficult to implement
- God object: Concentrating too many functions in a class
- Object orgy: Failed encapsulation permitting unrestricted access to internals
- Sequential coupling: Methods to be called in a particular order
- Accidental complexity: Unneeded complexity due to approach ($\neq$ inherent)
- Hard code: Spread the constant values everywhere in the code
- Repeating yourself: copy paste considered harmful (DRY SPOT)
- Action at a distance, Circular dependency, Premature optimization, Programming by permutation, Reinventing the square wheel, Shotgun surgery

Experience is what enables you to recognize a mistake when you make it again

Code Smell

What

- When you have a bad feeling from reading the code
- Often surface symptomes of deeper issues
- Some problems can be automatically detected

Examples

- Duplicated code: identical or very similar code exists in more than one location
- Contrived complexity: forced usage of overly complicated design patterns
- God class: a class that has grown too large; Lazy class: does too little
- Cyclomatic complexity: too many branches or loops
- Spread literals: should be named constants
- Refused bequest: An overrided method that changes the contract ($\neq$ Liskov)
- Excessively short/long identifiers
- Too many parameters: often denotes an ill-conveived function

`https://en.wikipedia.org/wiki/Code_smell` `https://en.wikipedia.org/wiki/Anti-pattern`

Functional Patterns

OO Pattern	FP Principle
Factory Pattern	Functions
Strategy Pattern	Functions
Decorator Pattern	Functions
Visitor Pattern	Functions
Interface Segregation Principle	Yes, functions

Functional Patterns

OO Pattern

- Factory Pattern
- Strategy Pattern
- Decorator Pattern
- Visitor Pattern
- Interface Segregation Principle

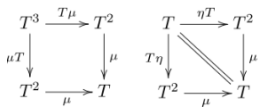
FP Principle

- Functions
- Functions
- Functions
- Functions
- Yes, functions

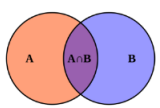
Design Patterns are OO boilerplate, FP is different

- (more on that troll: `http://c2.com/cgi/wiki?AreDesignPatternsMissingLanguageFeatures`)
- Now: quick tour of some FP recuring concepts

- Steal from Mathematics



- Types are not classes



- Functions are things

- Composition everywhere

http://fr.slideshare.net/ScottWlaschin/fp-patterns-buildstufflt

FP Core Principle #1: Steal from Mathematics

In Mathematics we Trust

- Mathematical assertions: unusually precise, general (large class of instances)
- Mathematics: discipline of reasoning $\leadsto$ unusually high confidence level

Some quotes from E. W. Dijkstra

- *Programming is one of the most difficult branch of applied mathematics.*
- *OOP is an exceptionally bad idea which could only have originated in California.*
- *CS is not more related to Computers than Astronomy to Telescopes.*

Why are mathematical functions so cool?

FP Core Principle #1: Steal from Mathematics

In Mathematics we Trust

- Mathematical assertions: unusually precise, general (large class of instances)
- Mathematics: discipline of reasoning $\leadsto$ unusually high confidence level

Some quotes from E. W. Dijkstra

- *Programming is one of the most difficult branch of applied mathematics.*
- *OOP is an exceptionally bad idea which could only have originated in California.*
- *CS is not more related to Computers than Astronomy to Telescopes.*

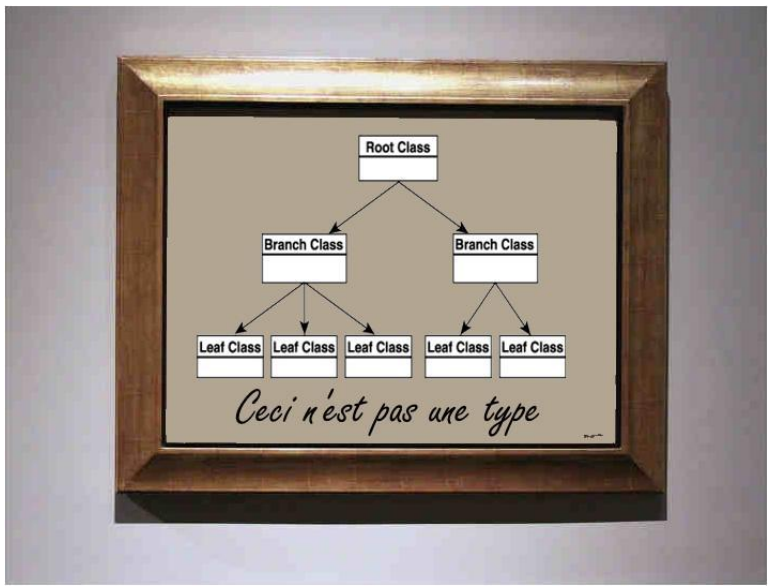
Why are mathematical functions so cool?

- They only map a value to another
- No calculation, just an immutable mapping (no side effect)

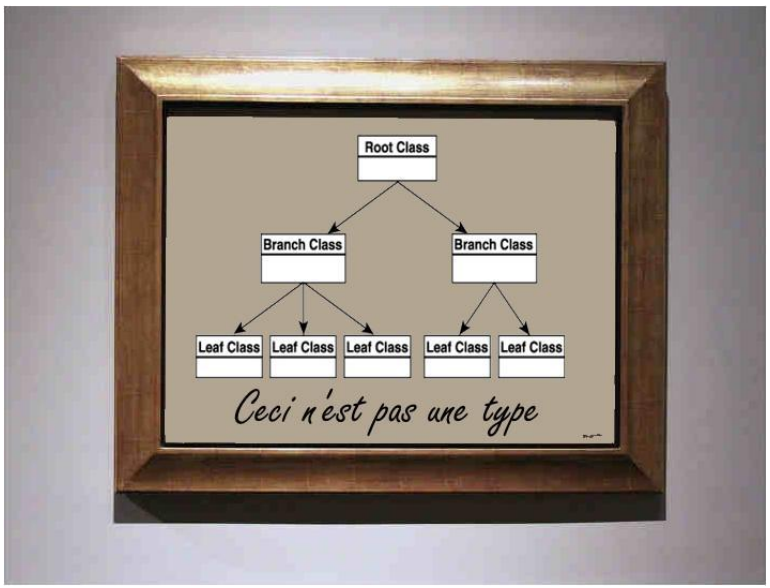
You should strive for Pure Functions

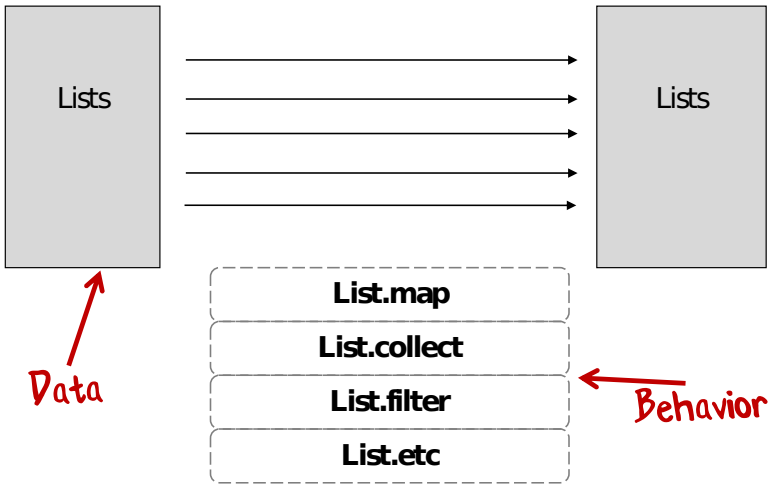
- Easy to understand, to reason about
- Lazily cachable results, Evaluation order does not matter

FP Core Principle #2: Types are not classes

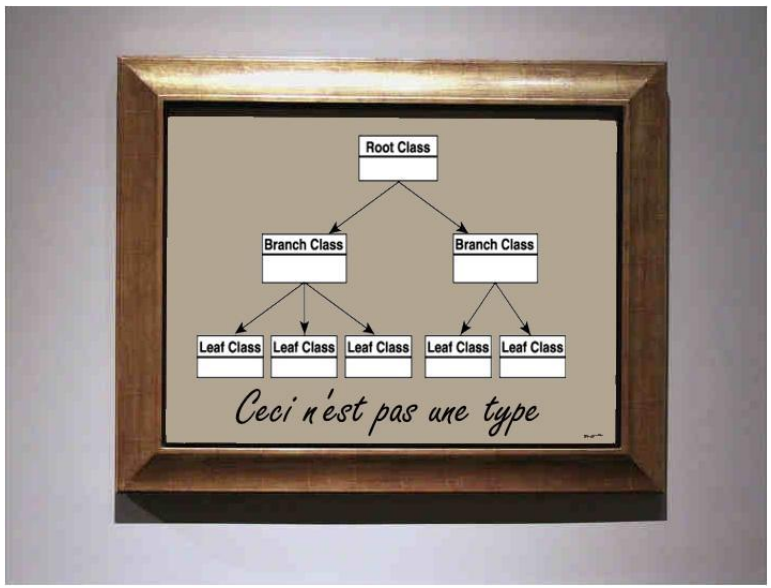


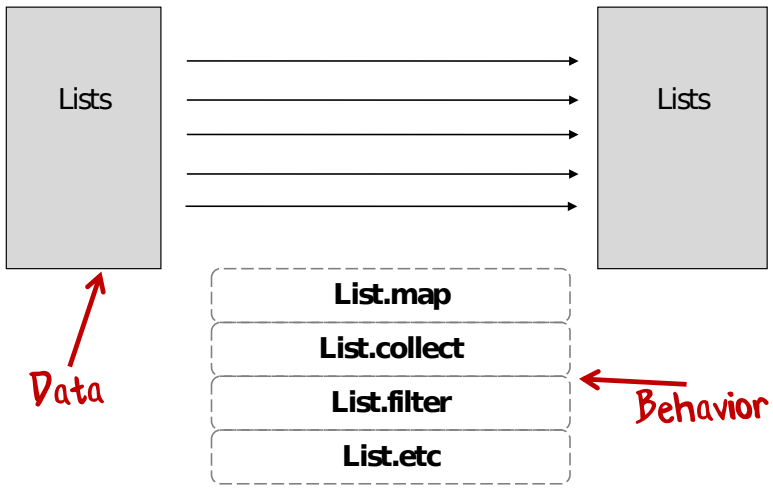
FP Core Principle #2: Types are not classes





FP Core Principle #2: Types are not classes





Types are just Data (set of values) $\neq$ Classes = Data + Behavior

Behavior is within the (pure) functions

FP Core Principle #3: Functions as things

A function is a standalone thing, not attached to a class

- You can manipulate functions as any other values
- Advice: Do as many pure functions as possible; **Avoid mutable objects**
- Advice: Parametrize all the things (Don't repeat yourself)

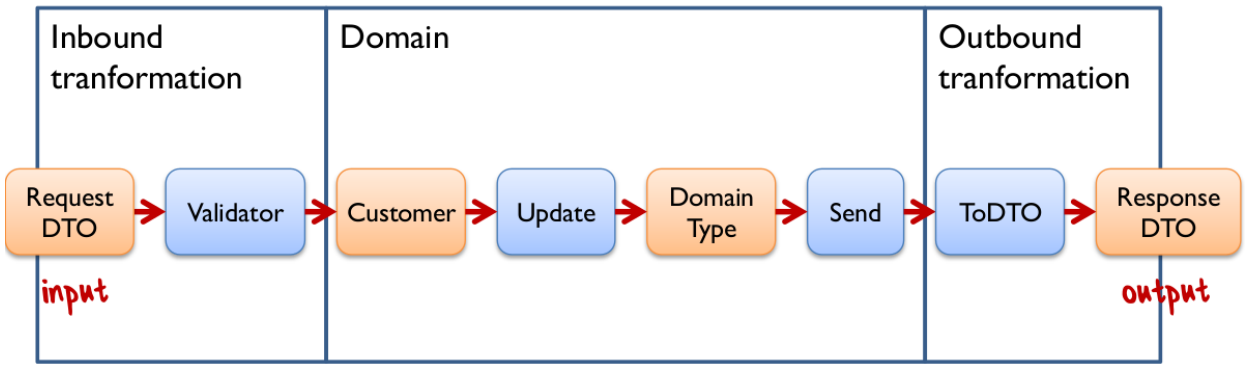
```java
public static int Product(int n) {
  int product = 1;
  for (int i = 1; i <= n; i++) {
    product *= i;
  }
  return product;
}
public static int Sum(int n) {
  int sum = 0;
  for (int i = 1; i <= n; i++) {
    sum += i;
  }
  return sum;
}
```

```scala
def product(n:Int) =
    (1 to n).foldLeft(1)(_ * _)

def product(n:Int) =
    (1 /: (1 to n)) (_ * _)


def product(n:Int) =
    (1 to n).foldLeft(0)(_ + _)

def sum(n:Int)     =
    (0 /: (1 to n)) (_ + _)
```

FP Core Principle #4: Composition everywhere

Functions can naturally be composed

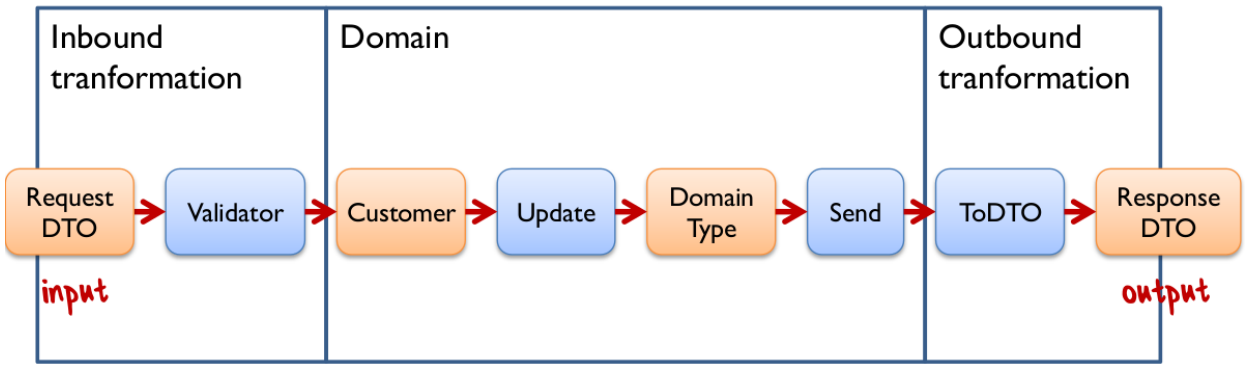
The FP flow is linear



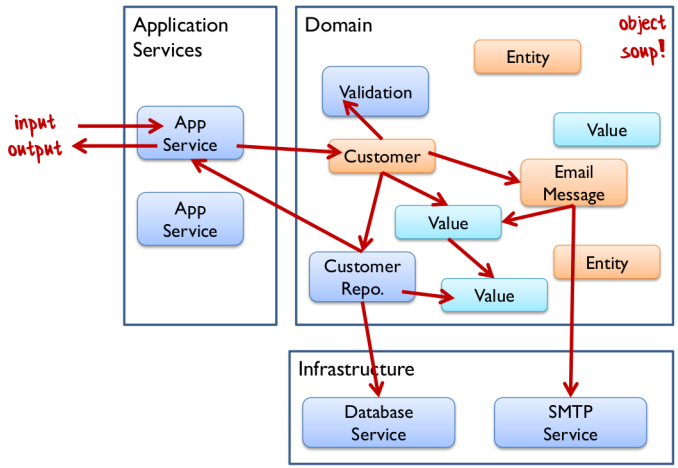
FP Core Principle #4: Composition everywhere

Functions can naturally be composed

The FP flow is linear



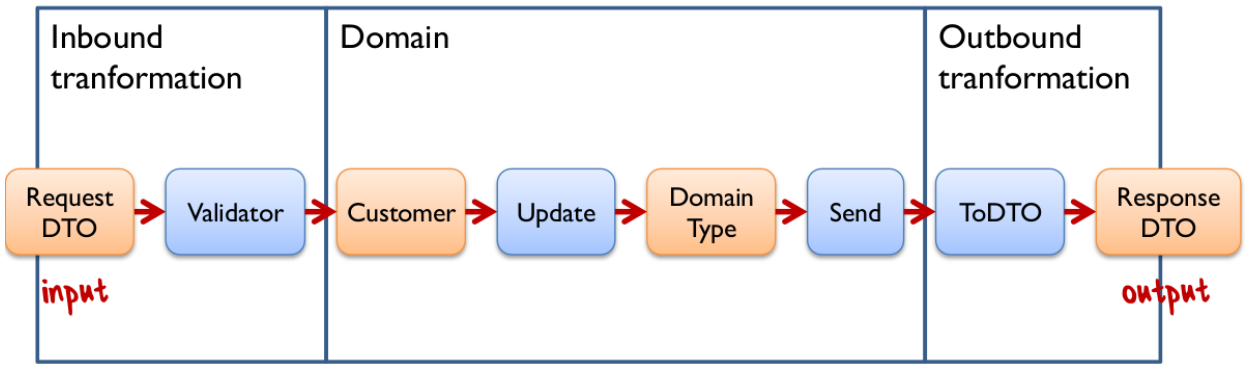
The OO flow soup



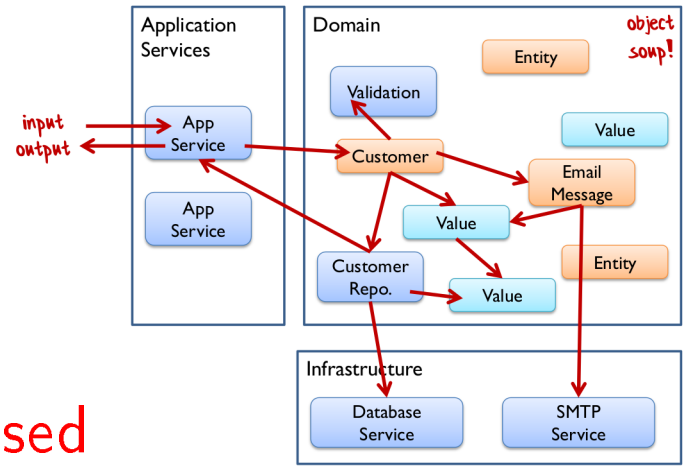
FP Core Principle #4: Composition everywhere

Functions can naturally be composed

The FP flow is linear



The OO flow soup



Types can also be composed

Product Type

- Tuple and type alias

```
val p = (1,2) // new Tuple2(1,2)
type Point = (Int, Int)
val p:Point = (1,2)
```

- Case class

```
case class Point(x: Int, y:Int)
val p = Point(1,2)
```

Sum Type

```
sealed trait Suit
case object Club    extends Suit
case object Diamond extends Suit
case object Spade   extends Suit
case object Heart   extends Suit
```

sealed: no inheritance from other files

FP style guide: `http://fr.slideshare.net/ScottWlaschin/fp-patterns-buildstufflt`

Painting Style

modernism

impressionism

abstract expressionism

realism

surrealism

cubism

photorealism

From: `http://gotocon.com/dl/goto-aar-2013/slides/CristinaVideiraLopes_ExercisesInStyle.pdf`

Writing Style

- Metaphor
- Surprises
- Dream
- Prognostication
- Hesitation
- Precision
- Negativities
- Asides
- Logical analysis
- Past
- Present
- ...
- (99)

What could be something like a programming style?

Exercises in Programming Style*

@cristalopes

github.com/crista/exercises-in-programming-style

*in bookstores Spring 14

Exercises in Programming Style

The story:

Term Frequency
given a text file,
output a list of the 25
most frequently-occurring
non stop, words, ordered by
decreasing frequency

Exercises in Programming Style

The story:

Pride and Prejudice → TF →

mr - 786
elizabeth - 635
very - 488
darcy - 418
such - 395
mrs - 343
much - 329
more - 327
bennet - 323
bingley - 306
jane - 295
miss - 283
one - 275
know - 239
before - 229
herself - 227
though - 226
well - 224
never - 220
…

Term Frequency
given a text file,
output a list of the 25
most frequently-occurring
words, ordered by decreasing
frequency

http://github.com/crista/
exercises-in-programming-style

@cristalopes #style1 *name*

STYLE #1

```
import sys, string
# the global list of [word, frequency] pairs
word_freqs = []
# the list of stop words
with open('../stop_words.txt') as f:
    stop_words = f.read().split(',')
stop_words.extend(list(string.ascii_lowercase))

# iterate through the file one line at a time
for line in open(sys.argv[1]):
    start_char = None
    i = 0
    for c in line:
        if start_char == None:
            if c.isalnum():
                # We found the start of a word
                start_char = i
        else:
            if not c.isalnum():
                # We found the end of a word. Process it
                found = False
                word = line[start_char:i].lower()
                # Ignore stop words
                if word not in stop_words:
                    pair_index = 0
                    # Let's see if it already exists
                    for pair in word_freqs:
                        if word == pair[0]:
                            pair[1] += 1
                            found = True
                            found_at = pair_index
                            break
                        pair_index += 1
                    if not found:
                        word_freqs.append([word, 1])
                    elif len(word_freqs) > 1:
                        # We may need to reorder
                        for n in reversed(range(pair_index)):
                            if word_freqs[pair_index][1] >
                                word_freqs[n][1]:
                                # swap
                                word_freqs[n], word_freqs[
                                    pair_index] = word_freqs[
                                    pair_index], word_freqs[n]
                                pair_index = n
                # Let's reset
                start_char = None
        i += 1

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

```
# the global list of [word, frequency] pairs
word_freqs = []
# the list of stop words
with open('../stop_words.txt') as f:
    stop_words = f.read().split(',')
stop_words.extend(list(string.ascii_lowercase))
```

```
8
9  # iterate through the file one line at a time
10 for line in open(sys.argv[1]):
11     start_char = None
12     i = 0
13     for c in line:
14         if start_char == None:
15             if c.isalnum():
16                 # We found the start of a word
17                 start_char = i
18         else:
19             if not c.isalnum():
20                 # We found the end of a word. Process it
21                 found = False
22                 word = line[start_char:i].lower()
23                 # Ignore stop words
24                 if word not in stop_words:
25                     pair_index = 0
26                     # Let's see if it already exists
27                     for pair in word_freqs:
28                         if word == pair[0]:
29                             pair[1] += 1
30                             found = True
31                             found_at = pair_index
32                             break
33                         pair_index += 1
34                     if not found:
35                         word_freqs.append([word, 1])
36                     elif len(word_freqs) > 1:
37                         # We may need to reorder
38                         for n in reversed(range(pair_index)):
39                             if word_freqs[pair_index][1] >
                                   word_freqs[n][1]:
40                                 # swap
41                                 word_freqs[n], word_freqs[
                                       pair_index] = word_freqs[
                                       pair_index], word_freqs[n]
42                                 pair_index = n
43                 # Let's reset
```

```
import sys, string
# the global list of [word, frequency] pairs
word_freqs = []
# the list of stop words
with open('../stop_words.txt') as f:
    stop_words = f.read().split(',')
stop_words.extend(list(string.ascii_lowercase))

for line in open(sys.argv[1]):
    for c in line:
        if start_char == None:
            if c.isalnum():
                # We found the start of a word
                start_char = i
        else:
            if not c.isalnum():
                # We found the end of a word. Process it
                found = False
                word = line[start_char:i].lower()
                # Ignore stop words
                if word not in stop_words:
                    pair_index = 0
                    # Let's see if it already exists
                    for pair in word_freqs:
                        if word == pair[0]:
                            pair[1] += 1
                            found = True
                            found_at = pair_index
                            break
                        pair_index += 1
                    if not found:
                        word_freqs.append([word, 1])
                    elif len(word_freqs) > 1:
                        # We may need to reorder
                        for n in reversed(range(pair_index)):
                            if word_freqs[pair_index][1] >
                                word_freqs[n][1]:
                                # swap
                                word_freqs[n], word_freqs[
                                    pair_index] = word_freqs[
                                    pair_index], word_freqs[n]
                                pair_index = n
                # Let's reset
                start_char = None
        i += 1

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

Style #1 Main Characteristics

- No abstractions
- No use of libraries

Style #1 Main Characteristics

- No abstractions
- No use of libraries

Monolith

Style #1 Main Characteristics

- No abstractions
- No use of libraries

Brain-dump Style

@cristalopes #style1 *name*

@cristalopes #style2 *name*

STYLE #2

```
import re, string, sys

stops = set(open("../stop_words.txt").read().split(",") + list(string.ascii_lowercase))
words = [x.lower() for x in re.split("[^a-zA-Z]+", open(sys.argv[1]).read()) if len(x) > 0 and x.lower() not in stops]
unique_words = list(set(words))
unique_words.sort(lambda x, y: cmp(words.count(y), words.count(x)))
print "\n".join(["%s - %s" % (x, words.count(x)) for x in unique_words[:25]])
```

Credit: *Laurie Tratt*, Kings College London

```
import re, string, sys

stops = set(open("../stop_words.txt").read().split(",") +
            list(string.ascii_lowercase))
words = [x.lower() for x in re.split("[^a-zA-Z]+",
                                     open(sys.argv[1]).read())
                   if len(x) > 0 and x.lower() not in stops]
unique_words = list(set(words))
unique_words.sort(lambda x, y: cmp(words.count(y),
                                   words.count(x)))
print "\n".join(["%s - %s" % (x, words.count(x))
                                  for x in unique_words[:25]])
```

```
import re, string, sys

stops = set(open("../stop_words.txt").read().split(",") +
            list(string.ascii_lowercase))
words = [x.lower() for x in re.split("[^a-zA-Z]+",
                                     open(sys.argv[1]).read())
                  if len(x) > 0 and x.lower() not in stops]
unique_words = list(set(words))
unique_words.sort(lambda x,y:cmp(words.count(y),
                                 words.count(x)))
print "\n".join(["%s - %s" % (x, words.count(x))
                             for x in unique_words[:25]])
```

Style #2 Main Characteristics

- No [named] abstractions
- Very few [long] lines of code
- Advanced libraries / constructs

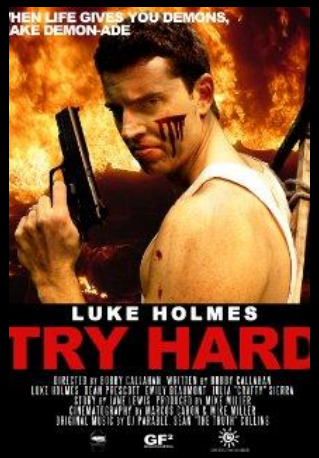
Style #2 Main Characteristics

- No [named] abstractions
- Very few [long] lines of code
- Advanced libraries / constructs

Code Golf Style

Style #2 Main Characteristics

- No [named] abstractions
- Very few [long] lines of code
- Advanced libraries / constructs



Try Hard Style

@cristalopes #style3 *name*

STYLE #3

```
import sys, string

# The shared mutable data
data = []
words = []
word_freqs = []

#
# The functions
#
def read_file(path_to_file):
    """
    Takes a path to a file and assigns the entire
    contents of the file to the global variable data
    """
    global data
    f = open(path_to_file)
    data = data + list(f.read())
    f.close()

def filter_chars_and_normalize():
    """
    Replaces all nonalphanumeric chars in data with white space
    """
    global data
    for i in range(len(data)):
        if not data[i].isalnum():
            data[i] = ' '
        else:
            data[i] = data[i].lower()

def scan():
    """
    Scans data for words, filling the global variable words
    """
    global data
    global words
    data_str = ''.join(data)
    words = words + data_str.split()

def remove_stop_words():
    global words
    f = open('../stop_words.txt')
    stop_words = f.read().split(',')
    f.close()
    # add single-letter words
    stop_words.extend(list(string.ascii_lowercase))
    indeces = []
    for i in range(len(words)):
        if words[i] in stop_words:
            indeces.append(i)
    for i in reversed(indeces):
        words.pop(i)


def frequencies():
    """
    Creates a list of pairs associating
    words with frequencies
    """
    global words
    global word_freqs
    for w in words:
        keys = [wd[0] for wd in word_freqs]
        if w in keys:
            word_freqs[keys.index(w)][1] += 1
        else:
            word_freqs.append([w, 1])

def sort():
    """
    Sorts word_freqs by frequency
    """
    global word_freqs
    word_freqs.sort(lambda x, y: cmp(y[1], x[1]))


#
# The main function
#
read_file(sys.argv[1])
filter_chars_and_normalize()
scan()
remove_stop_words()
frequencies()
sort()

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

```
data=[]
words=[]
freqs=[]
```

```
def read_file(path):
    """
    Takes a path to a file and assigns the entire
    contents of the file to the global variable data
    """
    global data
    f = open(path_to_file)
    data = data + list(f.read())
```

```
def filter_normalize():
    """
    Replaces all nonalphanumeric chars in data with white space
    """
    global data
    for i in range(len(data)):
        if not data[i].isalnum():
            data[i] = ' '
        else:
            data[i] = data[i].lower()
```

```
def scan():
    """
    Scans data for words, filling the global variable words
    """
    global data
    global words
    data_str = ''.join(data)
    words = words + data_str.split()
```

```
def rem_stop_words():
    f = open('../stop_words.txt')
    stop_words = f.read().split(',')
    f.close()
    # add single-letter words
    stop_words.extend(list(string.ascii_lowercase))
    indeces = []
    for i in range(len(words)):
        if words[i] in stop_words:
            indeces.append(i)
    for i in reversed(indeces):
        words.pop(i)
```

```
def frequencies():
    """
    words with frequencies
    """
    global words
    global word_freqs
    for w in words:
        keys = [wd[0] for wd in word_freqs]
        if w in keys:
            word_freqs[keys.index(w)][1] += 1
        else:
            word_freqs.append([w, 1])
```

```
def sort():
    """
    Sorts word_freqs by frequency
    """
    global word_freqs
    word_freqs.sort(lambda x, y: cmp(y[1], x[1]))
```

```
#
# Main
#
read_file(sys.argv[1])
filter_normalize()
scan()
rem_stop_words()
frequencies()
sort()

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

Style #3 Main Characteristics

- Procedural abstractions
  - maybe input, no output
- Shared state
- Commands

Style #3 Main Characteristics

- Procedural abstractions
  - maybe input, no output
- Shared state
- Commands

Cook Book Style

@cristalopes #style4 *name*

STYLE #4

```
import sys, re, operator, string

#
# The functions
#
def read_file(path_to_file):
    """
    Takes a path to a file and returns the entire
    contents of the file as a string
    """
    f = open(path_to_file)
    data = f.read()
    f.close()
    return data

def filter_chars(str_data):
    """
    Takes a string and returns a copy with all nonalphanumeric
    chars replaced by white space
    """
    pattern = re.compile('[\W_]+')
    return pattern.sub(' ', str_data)

def normalize(str_data):
    """
    Takes a string and returns a copy with all chars in lower case
    """
    return str_data.lower()

def scan(str_data):
    """
    Takes a string and scans for words, returning
    a list of words.
    """
    return str_data.split()

def remove_stop_words(word_list):
    """
    Takes a list of words and returns a copy with all stop
    words removed
    """
    f = open('../stop_words.txt')
    stop_words = f.read().split(',')
    f.close()
    # add single-letter words
    stop_words.extend(list(string.ascii_lowercase))
    return [w for w in word_list if not w in stop_words]

def frequencies(word_list):
    """
    Takes a list of words and returns a dictionary associating
    words with frequencies of occurrence
    """
    word_freqs = {}
    for w in word_list:
        if w in word_freqs:
            word_freqs[w] += 1
        else:
            word_freqs[w] = 1
    return word_freqs

def sort(word_freq):
    """
    Takes a dictionary of words and their frequencies
    and returns a list of pairs where the entries are
    sorted by frequency
    """
    return sorted(word_freq.iteritems(), key=operator.itemgetter
        (1), reverse=True)


#
# The main function
#
word_freqs = sort(frequencies(remove_stop_words(scan(normalize(
    filter_chars(read_file(sys.argv[1])))))))

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

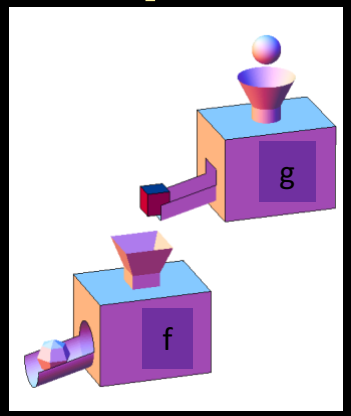
```
import sys, re, operator, string

#
# The functions
def read_file(path):
    """
    Takes a path to a file and returns the entire
    contents of the file as a string
    """
    f = open(path_to_file)
    return ...

def filter(str_data):
    """
    Takes a string and returns a copy with all nonalphanumeric
    chars replaced by white space
    """
    return ...e('[\W_]+')
    ... ', str_data)

def normalize(str_data):
    """
    ... returns a copy with all chars in lower case
    """
    return ...er()

def scan(str_data):
    """
    Takes a string and scans for words, returning
    a list of the words in the string
    """
    return ...
    return str_data.split()

def rem_stop_words(wordl):
    """
    Takes a list of words and returns a copy with all stop
    words removed
    """
    f = open('../stop_words.txt')
    stop_words = f.read().split(',')
    f.close()
    ... words
    return ...ist(string.ascii_lowercase))
    ...word_list if not w in stop_words]

def frequencies(wordl):
    """
    Takes a list of words and returns a dictionary associating
    words with frequencies of occurrence
    """
    word_freqs = {}
    for w in word_list:
        if w in word_freqs:
            word_freqs[w] += 1
        ...
    return ...

    - 1

def sort(word_freqs):
    """
    ...ies
    and returns a list of pairs where the entries are
    sorted by frequency
    """
    return ...eq.iteritems(), key=operator.itemgetter
    ...)

#
# Main
#
wfreqs=st(fq(r(sc(n(fc(rf(sys.argv[1])))))))

for tf in wfreqs[0:25]:
    print tf[0], ' - ', tf[1]
```

Style #4 Main Characteristics

- Function abstractions
  - *f: Input → Output*
- No shared state
- Function composition $f \circ g$

Style #4 Main Characteristics

- Function abstractions
  - *f: Input → Output*
- No shared state
- Function composition *f º g*



Candy Factory Style

@cristalopes #style4 *name*

Image credit: Nykamp DQ, From *Math Insight*. http://mathinsight.org/image/function_machines_composed

@cristalopes #style5 *name*

STYLE #5

```
import sys, re, operator, string

#
# The functions
#
def read_file(path_to_file, func):
    """
    Takes a path to a file and returns the entire
    contents of the file as a string
    """
    f = open(path_to_file)
    data = f.read()
    f.close()
    return func(data, normalize)

def filter_chars(str_data, func):
    """
    Takes a string and returns a copy with all nonalphanumeric
        chars
    replaced by white space
    """
    pattern = re.compile('[\W_]+')
    return func(pattern.sub(' ', str_data), scan)

def normalize(str_data, func):
    """
    Takes a string and returns a copy with all characters in lower
        case
    """
    return func(str_data.lower(), remove_stop_words)

def scan(str_data, func):
    """
    Takes a string and scans for words, returning
    a list of words.
    """
    return func(str_data.split(), frequencies)

def remove_stop_words(word_list, func):
    """ Takes a list of words and returns a copy with all stop
        words removed """
    f = open('../stop_words.txt')
    stop_words = f.read().split(',')
    f.close()
    # add single-letter words
    stop_words.extend(list(string.ascii_lowercase))
    return func([w for w in word_list if not w in stop_words],
        sort)

def frequencies(word_list, func):
    """
    Takes a list of words and returns a dictionary associating
    words with frequencies of occurrence
    """
    word_freqs = {}
    for w in word_list:
        if w in word_freqs:
            word_freqs[w] += 1
        else:
            word_freqs[w] = 1
    return func(word_freqs, no_op)

def sort(word_freq, func):
    """
    Takes a dictionary of words and their frequencies
    and returns a list of pairs where the entries are
    sorted by frequency
    """
    return func(sorted(word_freq.iteritems(), key=operator.
        itemgetter(1), reverse=True), None)

def no_op(a, func):
    return a

#
# The main function
#
word_freqs = read_file(sys.argv[1], filter_chars)

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

```
import sys, re, operator, string

#
# The functions
#
```

```
def read_file(path, func):
    ...
    return func(…, normalize)
```

```
def filter_chars(data, func):
    ...
    return func(…, scan)
def normalize(data, func):
    ...
    return func(…,remove_stops)
def scan(data, func):
    ...
    return func(…, frequencies)
```

```
def remove_stops(data, func):
    ...
    return func(…, sort)
```

```
def frequencies(
    """
    Takes a list
    words with f
    """
```

Etc.

```
    word_freqs = {}
    for w in word_list:
        if w in word_freqs:
            word_freqs[w] += 1
        else:
            word_freqs[w] = 1
    return func(word_freqs, no_op)

def sort(word_freq, func):
    """
    Takes a dictionary of words and their frequencies
    and returns a list of pairs where the entries are
    sorted by frequency
    """
    return func(sorted(word_freq.iteritems(), key=operator.
        itemgetter(1), reverse=True), None)

def no_op(a, func):
    return a
```

dictionary associating

e

```
# Main
w_freqs=read_file(sys.argv[1],
                  filter_chars)

for tf in w_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

Style #5 Main Characteristics

- Functions take one additional parameter, f
  - called at the end
  - given what would normally be the return value plus the next function

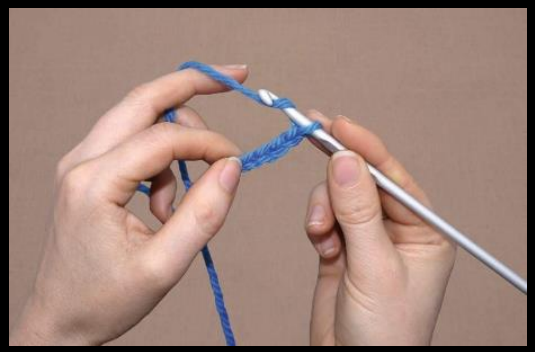
Style #5 Main Characteristics

- Functions take one additional parameter, f
  - called at the end
  - given what would normally be the return value pl[illegible]ext function

Kick teammates

Style #5 Main Characteristics

- Functions take one additional parameter, f
  - called at the end
  - given what would normally be the return value plus the next function



Crochet Style

@cristalopes #style5 *name*

@cristalopes #style6 *name*

STYLE #6

```
import sys, re, operator, string
from abc import ABCMeta

#
# The classes
#
class TFExercise(object):
    __metaclass__ = ABCMeta

    def info(self):
        return self.__class__.__name__ + ": No major data
            structure"

class DataStorageManager(TFExercise):
    """ Models the contents of the file """
    _data = ''
    def __init__(self, path_to_file):
        f = open(path_to_file)
        self._data = f.read()
        f.close()
        self.__filter_chars()
        self.__normalize()

    def __filter_chars(self):
        """
        Takes a string and returns a copy with all nonalphanumeric
             chars
        replaced by white space
        """
        pattern = re.compile('[\W_]+')
        self._data = pattern.sub(' ', self._data)

    def __normalize(self):
        """
        Takes a string and returns a copy with all characters in
            lower case
        """
        self._data = self._data.lower()

    def words(self):
        """
        Returns the list words in storage
        """
        data_str = ''.join(self._data)
        return data_str.split()

    def info(self):
        return self.__class__.__name__ + ": My major data
            structure is a " + self._data.__class__.__name__

class StopWordManager(TFExercise):
    """ Models the stop word filter """
    _stop_words = []
    def __init__(self):
        f = open('../stop_words.txt')
        self._stop_words = f.read().split(',')
        f.close()
        # add single-letter words
        self._stop_words.extend(list(string.ascii_lowercase))

    def is_stop_word(self, word):
        return word in self._stop_words

    def info(self):
        return self.__class__.__name__ + ": My major data
            structure is a " + self._stop_words.__class__.__name__

class WordFrequencyManager(TFExercise):
    """ Keeps the word frequency data """
    _word_freqs = {}

    def increment_count(self, word):
        if word in self._word_freqs:
            self._word_freqs[word] += 1
        else:
            self._word_freqs[word] = 1

    def sorted(self):
        return sorted(self._word_freqs.iteritems(), key=operator.
            itemgetter(1), reverse=True)

    def info(self):
        return self.__class__.__name__ + ": My major data
            structure is a " + self._word_freqs.__class__.__name__


class WordFrequencyController(TFExercise):
    def __init__(self, path_to_file):
        self._storage_manager = DataStorageManager(path_to_file)
        self._stop_word_manager = StopWordManager()
        self._word_freq_manager = WordFrequencyManager()

    def run(self):
        for w in self._storage_manager.words():
            if not self._stop_word_manager.is_stop_word(w):
                self._word_freq_manager.increment_count(w)

        word_freqs = self._word_freq_manager.sorted()
        for tf in word_freqs[0:25]:
            print tf[0], ' - ', tf[1]

#
# The main function
#
WordFrequencyController(sys.argv[1]).run()
```

```
import sys, re, operator, string
from abc import ABCMeta

#
# The classes
```

class TFExercise():

def info(self):

```
        structure"
```

class DataStorageManager(TFExercise):

```
    _data = ''
    def __init__(self, path_to_file):
        f = open(path_to_file)
        self._data = f.read()
        f.close()
        self.__filter_chars()
        self.__normalize()

    def __filter_chars(self):
        """
        Takes a string and returns a copy with all nonalphanumeric
             chars
        replaced by white space
        """
        pattern = re.compile('[\W_]+')
        self._data = pattern.sub(' ', self._data)

    def __normalize(self):
        """
        Takes a string and returns a copy with all characters in
            lower case
        """
        self._data = self._data.lower()
```

def words(self):

```
        Returns the list words in storage
        """
        data_str = ''.join(self._data)
```

def info(self):

```
        return self.__class__.__name__ + ": My major data
```

class StopWordManager(TFExercise):

```
    """ Models the stop word filter """
    _stop_words = []
    def __init__(self):
        f = open('../stop_words.txt')
        self._stop_words = f.read().split(',')
        f.close()
        # add single-letter words
```

def is_stop_word(self, word):

```
        return word in self._stop_words
```

def info(self):

```
        structure is a " + self._stop_words.__class__.__name__
```

class WordFreqManager(TFExercise):

```
    word_freqs = {}
```

def inc_count(self, word):

```
        self._word_freqs[word] += 1
    else:
        self._word_freqs[word] = 1
```

def sorted(self):

```
        itemgetter(1), reverse=True)
```

def info(self):

```
        structure is a " + self._word_freqs.__class__.__name__
```

class WordFreqController(TFExercise):

```
    def __init__(self, path_to_file):
        self._storage_manager = DataStorageManager(path_to_file)
        self._stop_word_manager = StopWordManager()
        self._word_freq_manager = WordFrequencyManager()

    def run(self):
        for w in self._storage_manager.words():
            if not self._stop_word_manager.is_stop_word(w):
                self._word_freq_manager.increment_count(w)

        word_freqs = self._word_freq_manager.sorted()
        for tf in word_freqs[0:25]:
            print tf[0], ' - ', tf[1]
```

```
# Main
WordFreqController(sys.argv[1]).run()
```

Style #6 Main Characteristics

- Things, things and more things!
- Capsules of data and procedures
- Data is never accessed directly
- Capsules say “I do the same things as that one, and more!”

Style #6 Main Characteristics

- Things, things and more things!
- Capsules of data and procedures
- Data is never accessed directly
- Capsules say “I do the same things as that one and more!”

Kingdom of Nouns Style

@cristalopes #style6 *name*

@cristalopes #style7 *name*

STYLE #7

```
import sys, re, operator, string

#
# Functions for map reduce
#
def partition(data_str, nlines):
    """
    Generator function that partitions the input data_str (a big
        string)
    into chunks of nlines.
    """
    lines = data_str.split('\n')
    for i in xrange(0, len(lines), nlines):
        yield '\n'.join(lines[i:i+nlines])

def split_words(data_str):
    """
    Takes a string, filters non alphanumeric characters,
        normalizes to
    lower case, scans for words, and filters the stop words.
    It returns a list of pairs (word, 1), one for each word in the
         input, so
    [(w1, 1), (w2, 1), ..., (wn, 1)]
    """
    def _filter_chars(str_data):
        """
        Takes a string and returns a copy with all nonalphanumeric
             chars
        replaced by white space
        """
        pattern = re.compile('[\W_]+')
        return pattern.sub(' ', str_data)

    def _normalize(str_data):
        """
        Takes a string and returns a copy with all characters in
            lower case
        """
        return str_data.lower()

    def _scan(str_data):
        """
        Takes a string and scans for words, returning
        a list of words.
        """
        return str_data.split()

    def _remove_stop_words(word_list):
        f = open('../stop_words.txt')
        stop_words = f.read().split(',')
        f.close()
        # add single-letter words
        stop_words.extend(list(string.ascii_lowercase))
        return [w for w in word_list if not w in stop_words]

    # The actual work of splitting the input into words
    result = []
    words = _remove_stop_words(_scan(_normalize(_filter_chars(
        data_str))))
    for w in words:
        result.append((w, 1))

    return result

def count_words(pairs_list_1, pairs_list_2):
    """
    Takes a two lists of pairs of the form
    [(w1, 1), ...]
    and returns a list of pairs [(w1, frequency), ...],
    where frequency is the sum of all the reported occurrences
    """
    mapping = dict((k, v) for k, v in pairs_list_1)
    for p in pairs_list_2:
        if p[0] in mapping:
            mapping[p[0]] += p[1]
        else:
            mapping[p[0]] = 1

    return mapping.items()

#
# Auxiliary functions
#

def read_file(path_to_file):
    """
    Takes a path to a file and returns the entire
    contents of the file as a string
    """
    f = open(path_to_file)
    data = f.read()
    f.close()
    return data

def sort(word_freq):
    """
    Takes a collection of words and their frequencies
    and returns a collection of pairs where the entries are
    sorted by frequency
    """
    return sorted(word_freq, key=operator.itemgetter(1), reverse=
        True)


#
# The main function
#
splits = map(split_words, partition(read_file(sys.argv[1]), 200))
splits.insert(0, []) # Normalize input to reduce
word_freqs = sort(reduce(count_words, splits))
```

```
import sys, re, operator, string

#
# Functions for map reduce
#
def partition(data_str, nlines):
    """
    Generator function that partitions the input data_str (a big
        string)
    into chunks of nlines.
    """
    lines = data_str.split('\n')
    for i in xrange(0, len(lines), nlines):
        yield '\n'.join(lines[i:i+nlines])

def split_words(data_str):
    """
    Takes a string, filters non alphanumeric characters,
        normalizes to
    lower case, scans for words, and filters the stop words.
    It returns a list of pairs (word, 1), one for each word in the
        input, so
    [(w1, 1), (w2, 1), ..., (wn, 1)]
    """
    def _filter_chars(str_data):
        """
        Takes a string and returns a copy with all nonalphanumeric
            chars
        replaced by white space
        """
        pattern = re.compile('[\W_]+')
        return pattern.sub(' ', str_data)

    def _normalize(str_data):
        """
        Takes a string and returns a copy with all characters in
            lower case
        """
        return str_data.lower()

    def _scan(str_data):
        """
        Takes a string and scan
        a list of words.
        """
        return str_data.split()

    def _remove_stop_words(word
        f = open('../stop_words
        stop_words = f.read().s
        f.close()
        # add single-letter wo
        stop_words.extend(list(
        return [w for w in word
```

```
    # The actual work of splitting the input into words
    result = []
    words = _remove_stop_words(_scan(_normalize(_filter_chars(
        data_str))))
    for w in words:
        result.append((w, 1))

    return result

def count_words(pairs_list_1, pairs_list_2):
    """
    Takes a two lists of pairs of the form
    [(w1, 1), ...]
    and returns a list of pairs [(w1, frequency), ...],
    where frequency is the sum of all the reported occurrences
    """
    mapping = dict((k, v) for k, v in pairs_list_1)
    for p in pairs_list_2:
        if p[0] in mapping:
            mapping[p[0]] += p[1]
        else:
            mapping[p[0]] = 1

    return mapping.items()

#
# Auxiliary functions
#

def read_file(path_to_file):
    """
    Takes a path to a file and returns the entire
    contents of the file as a string
    """
    f = open(path_to_file)
    data = f.read()
    f.close()
```

```
# Main
splits = map(split_words,
             partition(read_file(sys.argv[1]), 200))
splits.insert(0, [])
word_freqs = sort(reduce(count_words, splits))

for tf in word_freqs[0:25]:
    print tf[0], ' - ', tf[1]
```

```
import sys, re, operator, string

#
# Functions for map reduce
#
def partition(data_str, nlines):
    """
    Generator function that partitions the input data_str (a big
        string)
    into chunks of nlines.
    """
    lines = data_str.split('\n')
    for i in xrange(0, len(lines), nlines):
        yield '\n'.join(lines[i:i+nlines])
```

```
    # The actual work of splitting the input into words
    result = []
    words = _remove_stop_words(_scan(_normalize(_filter_chars(
        data_str))))
    for w in words:
        result.append((w, 1))

    return result

def count_words(pairs_list_1, pairs_list_2):
    """
    Takes a two lists of pairs of the form
    [(w1, 1), ...]
```

```
def split_words(data_str)
    """
    Takes a string (many lines), filters, normalizes to
    lower case, scans for words, and filters the stop words.
    Returns a list of pairs (word, 1), so
    [(w1, 1), (w2, 1), ..., (wn, 1)]
    """
    ...
    result = []
    words = _rem_stop_words(_scan(_normalize(_filter(data_str))))
    for w in words:
        result.append((w, 1))
    return result
```

```
        a list of words.
        """
        return str_data.split()

    def _remove_stop_words(word_list):
        f = open('../stop_words.txt')
        stop_words = f.read().split(',')
        f.close()
        # add single-letter words
        stop_words.extend(list(string.ascii_lowercase))
        return [w for w in word_list if not w in stop_words]
```

```
    sorted by frequency
    """
    return sorted(word_freq, key=operator.itemgetter(1), reverse=
        True)


#
# The main function
#
splits = map(split_words, partition(read_file(sys.argv[1]), 200))
splits.insert(0, []) # Normalize input to reduce
word_freqs = sort(reduce(count_words, splits))
```

```
import sys, re, operator, string

#
# Functions for map reduce
#
def partition(data_str, nlines):
    """
    Generator function that partitions the input data_str (a big
        string)
    into chunks of
    """
    lines = data_
    for i in xran
        yield '\n

def split_words(d
    """
    Takes a strin
        normalize
    lower case, s
    It returns a
         input, s
    [(w1, 1), (w2
    """
    def _filter_c
        """
        Takes a s
             char
        replaced
        """
        pattern =
        return pa

    def _normaliz
        """
        Takes a s
            lower
        """
        return st

    def _scan(str
        """
        Takes a s
        a list of words.
        """
        return str_data.split()

    def _remove_stop_words(word_list):
        f = open('../stop_words.txt')
        stop_words = f.read().split(',')
        f.close()
        # add single-letter words
        stop_words.extend(list(string.ascii_lowercase))
        return [w for w in word_list if not w in stop_words]
```

```
    # The actual work of splitting the input into words
    result = []
    words = _remove_stop_words(_scan(_normalize(_filter_chars(
        data_str))))
    for w in words:
        result.append((w, 1))

    return result
```

```
def count_words(pairs_list_1, pairs_list_2)
    """
    Takes two lists of pairs of the form
    [(w1, 1), ...]
    and returns a list of pairs [(w1, frequency), ...],
    where frequency is the sum of all occurrences
    """
    mapping = dict((k, v) for k, v in pairs_list_1)
    for p in pairs_list_2:
        if p[0] in mapping:
            mapping[p[0]] += p[1]
        else:
            mapping[p[0]] = 1

    return mapping.items()
```

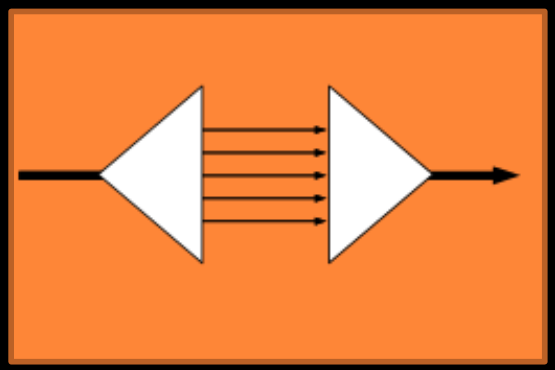
```
    sorted by frequency
    """
    return sorted(word_freq, key=operator.itemgetter(1), reverse=
        True)


#
# The main function
#
splits = map(split_words, partition(read_file(sys.argv[1]), 200))
splits.insert(0, []) # Normalize input to reduce
word_freqs = sort(reduce(count_words, splits))
```

Style #7 Main Characteristics

- Two key abstractions:
  map(f, chunks) and
  reduce(g, results)

Style #7 Main Characteristics

- Two key abstractions:
  map(f, chunks) and
  reduce(g, results)



iMux Style

Programming Style

There is always more than one way to express an idea

- Kiss! Getting things straight forward is very difficult
  Simplicity does not preceeds complexity, but follows it
- One programmer's elegant solution is another one displeasant beast

Improve your style now

- There is a huge literature on good programming style/methodology/buzzwords
- Reading code should be the way to learn, or at least write a lot of code

But don't search for your own style

- You are a journalist, not a poet: express ideas clearly with no stylistic trick
- Programming is a team game $\leadsto$ stick to some programming standards
- Programming tricks and Golf style should remain a game

Bad Style Coding as a Game

The International Obfuscated C Code Contest (www.ioccc.org)

- Yearly contest of intentionally obfuscated codes (in C; exist for other languages)

Example: (arachnid, 2004 entry)

```
#include <ncurses.h>/*****************************************************/
            int                 m[256                   ] [         256   ],a
 ,b   ;;;   ;;;   WINDOW*w;    char*l=""   "\176qxl"    "q"   "q"   "k"   "w\
xm"   "x"   "t"         "j"         "v"         "u"         "n"         ,Q[
 ]=   "Z"   "pt!ftd‘"   "qdc!‘eu"   "dq!$c!nnwf"/**   ***   */"t\040\t";c(
int   u ,         int         v){                     v?m   [u]         [v-
 1]   |=2,m[u][v-1] &   48?W][v-1   ] &   15]]):0:0;u?m[u   -1][v]|=1   ,m[
 u-                1][   v]&         48?               W-1   ][v         ]&
15]   ]):0:0;v<   255   ?m[   u][v+1]|=8,m[u][v+1]&   48?   W][   v+1]&15]]
):0         :0;         u <               255   ?m[   u+1         ][v   ]|=
4,m[u+1][   v]&48?W+1][v]&15]]):0:0;W][   v]&   15]   ]);}cu(char*q){   return
 *q                ?cu   (q+         1)&         1?q   [0]               ++:
q[0   ]--   :1;   }d(   int   u ,   int/**/v,   int/**/x,   int   y){   int
Y=y   -v,   X=x         -u;   int         S,s   ;Y<         0?Y   =-Y   ,s,
s=-   1:(   s=1);X<0?X=-X,S   =-1   :(S=   1);   Y<<=   1;X<<=1;   if(X>Y){
int   f=Y               -(X   >>1   );;               while(u!=         x){
f>=   0?v+=s,f-=X:0;u   +=S   ;f+=   Y;m[u][v]|=32;mvwaddch(w,v   ,u,   m[u
 ][               v]&   64?   60:         46)         ;if         (m[   u][
v]&16){c(u,v);;   ;;;   ;;;   return;}}   }else{int   f=X   -(Y>>1);;   while
 (v   !=y         ){f   >=0         ?u   +=S,               f-=         Y:0
 ;v   +=s   ;f+=X;m[u][v]|=   32;mvwaddch(w,v   ,u,m[u][v]&64?60:46);if(m[u
 ][                    v]&         16)   {c(   u,v                     );
  ;   return;;;}}}Z(   int/**/a,   int   b){   }e(   int/**/y,int/**/   x){
int               i ;         for         (i=         a;i               <=a
+S;i++)d(y,x,i,b),d(y,x,i,b+L);for(i=b;i<=b+L;i++)d(y,x,a,i),d(y,x,a+   S,i
 );                   ;;;         ;;;         ;;;               ;;;   ;
  mvwaddch(w,x,y,64);   ;;;   ;;;   ;;;   prefresh(   w,b,a,0,0   ,L-   1,S-1
);}            main(         int               V ,   char            *C[
  ]   ){FILE*f=   fopen(V==1?"arachnid.c"/**/   :C[   1],"r");int/**/x,y,c,
```

(source code not tested)

Bad Style Coding as a Game

The International Obfuscated C Code Contest (www.ioccc.org)

- Yearly contest of intentionally obfuscated codes (in C; exist for other languages)

Example: Full (interactive) Maze Escape Game (arachnid, 2004 entry)

```
#include <ncurses.h>/*****************************************************/
            int                 m[256                   ] [         256   ],a
 ,b   ;;;   ;;;   WINDOW*w;    char*l=""   "\176qxl"    "q"    "q"    "k"    "w\
xm"   "x"   "t"          "j"          "v"          "u"          "n"          ,Q[
 ]=   "Z"   "pt!ftd‘"    "qdc!‘eu"    "dq!$c!nnwf"/**   ***    */"t\040\t";c(
int   u ,          int          v){                      v?m    [u]          [v-
 1]   |=2,m[u][v-1] &    48?W][v-1    ] &    15]]):0:0;u?m[u    -1][v]|=1    ,m[
 u-                 1][   v]&          48?                 W-1    ][v          ]&
15]   ]):0:0;v<    255    ?m[    u][v+1]|=8,m[u][v+1]&    48?    W][    v+1]&15]]
):0          :0;          u <                 255    ?m[    u+1          ][v    ]|=
4,m[u+1][    v]&48?W+1][v]&15]]):0:0;W][    v]&    15]    ]);}cu(char*q){    return
 *q                 ?cu    (q+          1)&          1?q    [0]                 ++:
q[0    ]--    :1;    }d(    int    u ,    int/**/v,    int/**/x,    int    y){    int
Y=y    -v,    X=x          -u;    int          S,s    ;Y<          0?Y    =-Y    ,s,
s=-    1:(    s=1);X<0?X=-X,S    =-1   :(S=    1);    Y<<=    1;X<<=1;    if(X>Y){
int    f=Y                 -(X    >>1    );;                 while(u!=          x){
f>=    0?v+=s,f-=X:0;u    +=S    ;f+=    Y;m[u][v]|=32;mvwaddch(w,v    ,u,    m[u
 ][                 v]&    64?    60:          46)          ;if          (m[    u][
v]&16){c(u,v);;    ;;;    ;;;    return;}}    }else{int    f=X    -(Y>>1);;    while
 (v    !=y          ){f    >=0          ?u    +=S,                 f-=          Y:0
 ;v    +=s    ;f+=X;m[u][v]|=    32;mvwaddch(w,v    ,u,m[u][v]&64?60:46);if(m[u
 ][                 v]&          16)    {c(    u,v                          );
  ;    return;;;}}}}Z(    int/**/a,    int    b){    }e(    int/**/y,int/**/    x){
int                 i ;          for          (i=          a;i                 <=a
+S;i++)d(y,x,i,b),d(y,x,i,b+L);for(i=b;i<=b+L;i++)d(y,x,a,i),d(y,x,a+    S,i
 );                 ;;;          ;;;          ;;;                 ;;;    ;
  mvwaddch(w,x,y,64);    ;;;    ;;;    ;;;    prefresh(    w,b,a,0,0    ,L-    1,S-1
);}             main(          int                 V ,    char                *C[
  ]    ){FILE*f=    fopen(V==1?"arachnid.c"/**/    :C[    1],"r");int/**/x,y,c,
                    (source code not here)
```

Bad Style Coding as a Game

The International Obfuscated C Code Contest (www.ioccc.org)

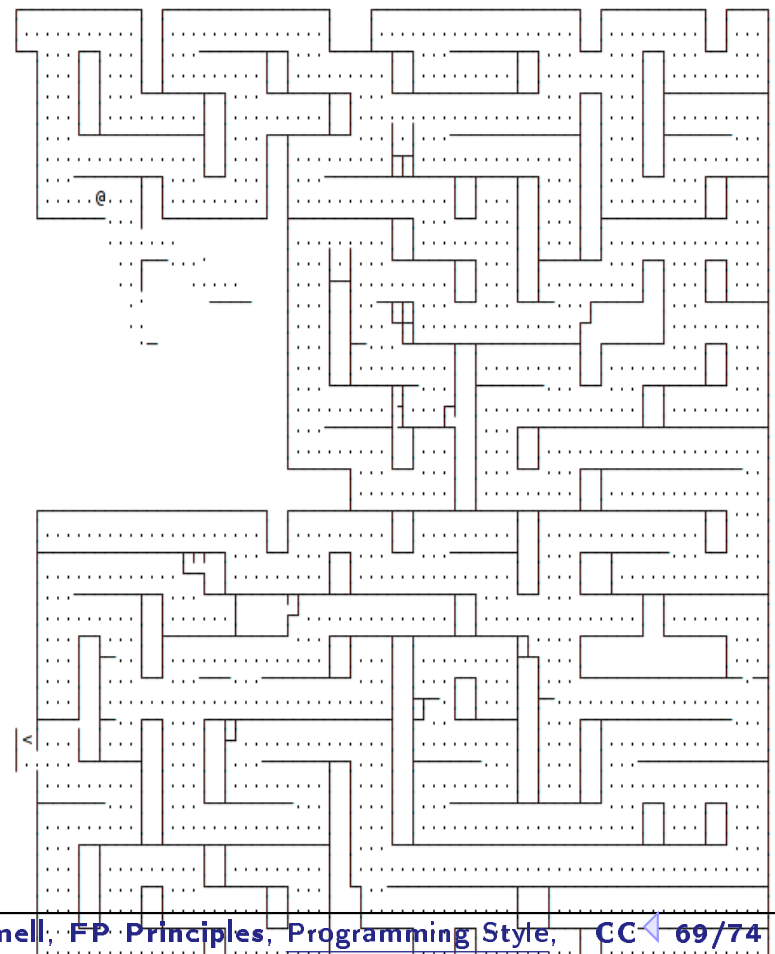
- Yearly contest of intentionally obfuscated codes (in C; exist for other languages)

Example: Full (interactive) Maze Escape Game (arachnid, 2004 entry)

```
#include <ncurses.h>/*****************************************************/
            int                 m[256                       ] [         256   ],a
 ,b   ;;;   ;;;   WINDOW*w;     char*l=""   "\176qxl"   "q"   "q"   "k"   "w\
xm"   "x"   "t"         "j"         "v"         "u"         "n"         ,Q[
 ]=   "Z"   "pt!ftd‘"   "qdc!‘eu"   "dq!$c!nnwf"/**   ***   */"t\040\t";c(
int   u ,         int         v){                         v?m   [u]         [v-
 1]   |=2,m[u][v-1] &   48?W][v-1   ] &   15]]):0:0;u?m[u   -1][v]|=1    ,m[
 u-                1][   v]&         48?                 W-1   ][v         ]&
15]   ]):0:0;v<   255   ?m[   u][v+1]|=8,m[u][v+1]&   48?   W][   v+1]&15]]
):0         :0;         u <               255   ?m[   u+1         ][v   ]|=
4,m[u+1][   v]&48?W+1][v]&15]]):0:0;W][   v]&   15]   ]);}cu(char*q){   return
 *q                 ?cu   (q+         1)&         1?q   [0]                 ++:
q[0   ]--   :1;   }d(   int   u ,   int/**/v,   int/**/x,   int   y){   int
Y=y   -v,   X=x         -u;   int         S,s   ;Y<         0?Y   =-Y   ,s,
s=-   1:(   s=1);X<0?X=-X,S   =-1   :(S=   1);   Y<<=   1;X<<=1;   if(X>Y){
int   f=Y               -(X   >>1   );;               while(u!=         x){
f>=   0?v+=s,f-=X:0;u   +=S   ;f+=   Y;m[u][v]|=32;mvwaddch(w,v   ,u,   m[u
 ][                v]&   64?   60:         46)         ;if         (m[   u][
v]&16){c(u,v);;   ;;;   ;;;   return;}}   }else{int   f=X   -(Y>>1);;   while
 (v   !=y         ){f   >=0         ?u   +=S,                 f-=         Y:0
 ;v   +=s   ;f+=X;m[u][v]|=   32;mvwaddch(w,v   ,u,m[u][v]&64?60:46);if(m[u
 ][                v]&         16)   {c(   u,v                         );
  ;   return;;;}}}}Z(   int/**/a,   int   b){   }e(   int/**/y,int/**/   x){
int               i ;         for         (i=         a;i                 <=a
+S;i++)d(y,x,i,b),d(y,x,i,b+L);for(i=b;i<=b+L;i++)d(y,x,a,i),d(y,x,a+   S,i
 );                ;;;         ;;;         ;;;                 ;;;   ;
  mvwaddch(w,x,y,64);   ;;;   ;;;   ;;;   prefresh(   w,b,a,0,0   ,L-   1,S-1
);}            main(         int               V ,   char              *C[
  ]   ){FILE*f=   fopen(V==1?"arachnid.c"/**/   :C[   1],"r");int/**/x,y,c,
```

(source code not tested)

Screenshoot



Recreational Obfuscation: Phillips entry of IOCCC'8

Program code

```
#include <stdio.h>
main(t,_,a)char *a;{return!0<t?t<3?main(-79,-13,a+main(-87,1-_,
main(-86,0,a+1)+a)):1,t<_?main(t+1,_,a):3,main(-94,-27+t,a)&&t==2?_<13?
main(2,_+1,"%s %d %d\n"):9:16:t<0?t<-72?main(_,t,
"@n'+,#'/*{}w+/w#cdnr/+,{}r/*de}+,/*{*+,/w{%+,/w#q#n+,/#{l,+,/n{n+,/+#n+,/#\
;#q#n+,/+k#;*+,/'r :'d*'3,}{w+K w'K:'+}e#';dq#'l \
q#'+d'K#!/+k#;q#'r}eKK#}w'r}eKK{nl]'/#;#q#n'){)#}w'){){nl]'/+#n';d}rw' i;# \
){nl]!/n{n#'; r{#w'r nc{nl]'/#{l,+'K {rw' iK{;[{nl]'/w#q#n'wk nw' \
iwk{KK{nl]!/w{%'l##w#' i; :{nl]'/*{q#'ld;r'}{nlwb!/*de}'c \
;;{nl'-{}rw]'/+,}##'*}#nc,',#nw]'/+kd'+e}+;#'rdq#w! nr'/ ') }+}{rl#'{n' ')# \
}'+}##(!!/")
:t<-50?_==*a?putchar(31[a]):main(-65,_,a+1):main((*a=='/')+t,_,a+1)
  :0<t?main(2,2,"%s"):*a=='/'||main(0,main(-61,*a,
"!ek;dc i@bK'(q)-[w]*%n+r3#l,{}:\nuwloca-O;m .vpbks,fxntdCeghiry"),a+1);}
```

Output

```
On the first day of Christmas my true love gave to me
a partridge in a pear tree.

On the second day of Christmas my true love gave to me
two turtle doves
and a partridge in a pear tree.

On the third day of Christmas my true love gave to me
three french hens, two turtle doves
and a partridge in a pear tree.

On the fourth day of Christmas my true love gave to me
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the fifth day of Christmas my true love gave to me
```

Output (cont)

```
On the eighth day of Christmas my true love gave to me
eight maids a-milking, seven swans a-swimming,
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the ninth day of Christmas my true love gave to me
nine ladies dancing, eight maids a-milking, seven swans a-swimming,
six geese a-laying, five gold rings;
four calling birds, three french hens, two turtle doves
and a partridge in a pear tree.

On the tenth day of Christmas my true love gave to me
ten lords a-leaping,
nine ladies dancing, eight maids a-milking, seven swans a-swimming,
six geese a-laying, five gold rings;
```

Bad Coding Style as an Art

Another example

```
#include <stdio.h>
int l;int main(int o,char **O,
int I){char c,*D=O[1];if(o>0){
for(l=0;D[l              ];D[l
++]-=10){D   [l++]-=120;D[l]-=
110;while   (!main(0,O,l))D[l]
+=   20;   putchar((D[l]+1032)
/20   )   ;}putchar(10);}else{
c=o+     (D[I]+82)%10-(I>l/2)*
(D[I-l+I]+72)/10-9;D[I]+=I<0?0
:!(o=main(c/10,O,I-1))*((c+999
)%10-(D[I]+92)%10);}return o;}
```

Bad Coding Style as an Art

Another example: Computing Integer Square Roots

```
#include <stdio.h>
int l;int main(int o,char **O,
int I){char c,*D=O[1];if(o>0){
for(l=0;D[l                ];D[l
++]-=10){D   [l++]-=120;D[l]-=
110;while   (!main(0,O,l))D[l]
+=   20;   putchar((D[l]+1032)
/20   )   ;}putchar(10);}else{
c=o+     (D[I]+82)%10-(I>l/2)*
(D[I-l+I]+72)/10-9;D[I]+=I<0?0
:!(o=main(c/10,O,I-1))*((c+999
)%10-(D[I]+92)%10);}return o;}
```

It actually works

```
$ ./cheong 1234
35
```

$(35 \times 35 = 1225;\ 35 \times 36 = 1296)$

```
$ ./cheong 112233445566
335012
```

$335012 \times 335012 = 112233040144$

$335013 \times 335013 = 112233710169$

Bad Coding Style as an Art

Another example: Computing Interger Square Roots

```
#include <stdio.h>
int l;int main(int o,char **O,
int I){char c,*D=O[1];if(o>0){
for(l=0;D[l                ];D[l
++]-=10){D   [l++]-=120;D[l]-=
110;while   (!main(0,O,l))D[l]
+=   20;   putchar((D[l]+1032)
/20   )   ;}putchar(10);}else{
c=o+     (D[I]+82)%10-(I>l/2)*
(D[I-l+I]+72)/10-9;D[I]+=I<0?0
:!(o=main(c/10,O,I-1))*((c+999
)%10-(D[I]+92)%10);}return o;}
```

It actually works

```
$ ./cheong 1234
35
```

$(35 \times 35 = 1225;\ 35 \times 36 = 1296)$

```
$ ./cheong 112233445566
335012
```

$335012 \times 335012 = 112233040144$

$335013 \times 335013 = 112233710169$

Author claim: code self-documented...

```
#include <stdio.h>
int l;int main(int o,char **O,
int I){char c,*D=O[1];if(o>0){
for(l=0;D[l                ];D[l
++]-=10){D   [l++]-=120;D[l]-=
110;while   (!main(0,O,l))D[l]
+=   20;   putchar((D[l]+1032)
/20   )   ;}putchar(10);}else{
c=o+     (D[I]+82)%10-(I>l/2)*
(D[I-l+I]+72)/10-9;D[I]+=I<0?0
:!(o=main(c/10,O,I-1))*((c+999
)%10-(D[I]+92)%10);}return o;}
```

Bad Coding Style as an Art

Another example: Computing Interger Square Roots

```
#include <stdio.h>
int l;int main(int o,char **O,
int I){char c,*D=O[1];if(o>0){
for(l=0;D[l                ];D[l
++]-=10){D   [l++]-=120;D[l]-=
110;while   (!main(0,O,l))D[l]
+=   20;   putchar((D[l]+1032)
/20   )   ;}putchar(10);}else{
c=o+     (D[I]+82)%10-(I>l/2)*
(D[I-l+I]+72)/10-9;D[I]+=I<0?0
:!(o=main(c/10,O,I-1))*((c+999
)%10-(D[I]+92)%10);}return o;}
```

It actually works

```
$ ./cheong 1234
35
```

$(35 \times 35 = 1225;\ 35 \times 36 = 1296)$

```
$ ./cheong 112233445566
335012
```

$335012 \times 335012 = 112233040144$

$335013 \times 335013 = 112233710169$

Author claim: code self-documented...

```
#include <stdio.h>
int l;int main(int o,char **O,
int I){char c,*D=O[1];if(o>0){
for(l=0;D[l                ];D[l
++]-=10){D   [l++]-=120;D[l]-=
110;while   (!main(0,O,l))D[l]
+=   20;   putchar((D[l]+1032)
/20   )   ;}putchar(10);}else{
c=o+     (D[I]+82)%10-(I>l/2)*
(D[I-l+I]+72)/10-9;D[I]+=I<0?0
:!(o=main(c/10,O,I-1))*((c+999
)%10-(D[I]+92)%10);}return o;}
```

It is an old observation that the best writers sometimes disregard the rules of rhetoric. When they do so, however, the reader will usually find in the sentence some compensating merit, attained at the cost of the violation. ***Unless he is certain of doing as well, he will probably do best to follow the rules.***

– William Strunk, Jr. (1918)

Last one, just for fun: dhyang IOCCC'00

Saitou Hajime image

Source code

```
                #define/**/X
               char*d="X0[!4cM,!"
             "4cK`*!4cJc(!4cHg&!4c$j"
           "8f'!&~]9e)!'|:d+!)rAc-!*m*"
         ":d/!4c(b4e0!lr2e2!/t0e4!-y-c6!"
       "+|,c6!)f$b(h*c6!(d'b(i)d5!(b*a``&c"
       ")c5!'b+`&b'c)c4!&b-_$c'd*c3!&a.h'd+"
     "dl!%a/g'e+eO!%b-g(d.d/!&c*h'dld-!(d%g)"
   "d4d+!*l,d7d)!,h-d;c'!.bOc>d%!A`Dc$![7)35E"
  "!'1cA,,!2kE`*!-s@d(!(k(f//g&!)f.e5'f(!+a+)"
  "f%2g*!?f5f,!=f-*e/!<d6el!9eO'f3!6f)-g5!4d*b"
  "+e6!Of%k)d7!+~^'c7!)z/d-+!'n%aO(d5!%cla+/d4"
   "!2)c9e2!9b;e1!8b>e/!     7cAd-!5fAe+!7fBe(!"
 "8hBd&!:iAd$![7S,QO!l     bF 7!lb?'_6!lc,8b4"
  "!2b*a,*d3!2n4f2!${4    f.      '!%y4e5!&f%"
 "d-^-d7!4c+b)d9!4c-a    'd        :!/i(``&d"
 ";!+l'a+d<!)l*b(d=!'   m-        a  &d>!&d'"
 "`O_&c?!$dAc@!$cBc@!$   b         <    ^&d$`"
 ":!$d9_&l++^$!%f3a'    nl              $ !&"
"f/c(o/_%!(f+c)q*c     %!         *       f &d+"
"f$s&!-n,d)n(!Oi-     c-         k)       !  3d"
"/bOh*!H`7a,![7*     i]          5        4   71"
"[=ohr&o*t*q*`*d      *v         *r         ;  O2"
"7*~=h./}tcrsth      &t          :          r   9b"
"]..,b-725-.t--//     #r         [           <   t8-"
"752793?  <.~;b      ].t--+r     /           #    53"
"7-r[/9~X  .v9O      <6/<.v;-52/={            k   goh"
"./}q;   u  vto     hr  `.i*$engt$            $    ,b"
";$/     =t ;v;     6     =`it.`;7=`          :    ,b-"
"725    = / o`.     .d       ;b]`--[/+       55/    }o"
"`.d   :   - ?5    /           }o`.'     v/i]q      - "
"-[;   5  2  =`  it            .        o;53-       . "
"v96   <7 /      =o            :           d        =o"
"--/i  ]q--      [;           h.           /        = "
"i]q--[  ;v      9h           ./           <        - "
"52={cj   u      c&`          i   t       . o       ; "
"?4=o:d=         o--          /  i        ]q        - "
"-[;54={  cj     uc&          i]q          -         -"
"[;76=i]q[;6     =vsr        u.i           /         ={"
"=),BihY_gha     ,)\0         "            ,         o [
 3217];int i,   r,w,f         ,             b        ,x ,
 p;n(){return   r  <X         X              X       X  X
 768?d[X(143+   X  r++        +              *d      )  %
  768]:r>2659   ?  59:        (               x      =  d
  [(r++-768)%   X  947       +             768]      ) ?
  x^(p?6:0):(p  =   34       X           X           X )
  ;}s(){for(x=  n   ();      (           x^          ( p
 ?6:0))==32;x=  n    ()      )   ;return x           ; }
 void/**/main X      ()      {           r          =  p
 =0;w=sprintf  (X     X     X         X X          X o
 ,"char*d=");  for          (    f=1;f <           * d
 +143;)if(33-(  b=d         [      f++ X          ]  )
 ){if(b<93){if   X(!        p          )            o
  [w++]=34;for    X(i       =         35            +
   (p?0:1);i<b;    i++      )         o
   [w++]=s();o[     w++               ]
    =p?s():34;}     else              X
      {for(i=92;     i<b;            i
       ++)o[w++]=     32;}           }
            else o     [w++         ]
                       =10;o        [
                         w]=0      ;
                          puts(o);}
```

Last one, just for fun: dhyang IOCCC'00

Saitou Hajime image that prints a prog

Source code

```
                #define/**/X
              char*d="X0[!4cM,!"
            "4cK`*!4cJc(!4cHg&!4c$j"
          "8f'!&~]9e)!']:d+!)rAc-!*m*"
        ":d/!4c(b4e0!1r2e2!/t0e4!-y-c6!"
      "+|,c6!)f$b(h*c6!(d'b(i)d5!(b*a``&c"
      ")c5!'b+`&b'c)c4!&b-_$c'd*c3!&a.h'd+"
    "dl!%a/g'e+eO!%b-g(d.d/!&c*h'd1d-!(d%g)"
   "d4d+!*l,d7d)!,h-d;c'!.b0c>d%!A`Dc$![7)35E"
  "!'1cA,,!2kE`*!-s@d(!(k(f//g&!)f.e5'f(!+a+)"
  "f%2g*!?f5f,!=f-*e/!<d6e1!9e0'f3!6f)-g5!4d*b"
  "+e6!0f%k)d7!+~^'c7!)z/d-+!'n%aO(d5!%c1a+/d4"
  "!2)c9e2!9b;e1!8b>e/!     7cAd-!5fAe+!7fBe(!"
 "8hBd&!:iAd$![7S,QO!1     bF 7!1b?'_6!1c,8b4"
 "!2b*a,*d3!2n4f2!${4    f.      '!%y4e5!&f%"
 "d-^-d7!4c+b)d9!4c-a     'd        :!/i(``&d"
 ";!+l'a+d<!)l*b(d=!'   m-        a  &d>!&d'"
 "`O_&c?!$dAc@!$cBc@!$   b         <    ^&d$`"
 ":!$d9_&l++^$!%f3a'    n1               $ !&"
 "f/c(o/_%!(f+c)q*c     %!         *       f &d+"
 "f$s&!-n,d)n(!Oi-     c-          k)      !  3d"
 "/bOh*!H`7a,![7*     i]          5       4   71"
"[=ohr&o*t*q*`*d      *v          *r       ;  O2"
"7*~=h./}tcrsth      &t           :        r   9b"
"]..b-725-.t--//     #r           [        <   t8-"
"752793?  <.~;b      ].t--+r      /        #    53"
"7-r[/9~X  .v90      <6/<.v;-52/={          k   goh"
"./}q;   u  vto     hr  `.i*$engt$          $    ,b"
";$/     =t ;v;     6     =`it.`;7=`        :    ,b-"
"725    = / o`.     .d       ;b]`--[/+      55/    }o"
"`.d   :   - ?5    /           }o`.'     v/i]q      - "
"-[;   5  2  =`  it             .       o;53-       . "
"v96   <7 /      =o             :          d        =o"
"--/i  ]q--      [;            h.          /        = "
"i]q--[  ;v      9h            ./          <        - "
"52={cj   u      c&`           i   t       . o       ; "
"?4=o:d=         o--           /  i       ]q        - "
"-[;54={  cj     uc&           i]q          -        -"
"[;76=i]q[;6     =vsr         u.i           /        ={"
"=),BihY_gha     ,)\0          "            ,         o [
 3217];int i,   r,w,f          ,             b        ,x ,
 p;n(){return   r  <X         X              X       X  X
 768?d[X(143+   X  r++        +              *d      )  %
  768]:r>2659   ?  59:        (               x       =  d
  [(r++-768)%   X  947       +             768]       ) ?
  x^(p?6:0):(p  =   34      X             X           X )
  ;}s(){for(x=  n   ();     (            x^           ( p
 ?6:0))==32;x=  n    ()     )   ;return x             ; }
 void/**/main X      ()     {            r           =  p
 =0;w=sprintf  (X     X     X         X X           X o
 ,"char*d=");  for          (    f=1;f <             * d
 +143;)if(33-(  b=d         [      f++ X           ]  )
 ){if(b<93){if   X(!        p           )            o
  [w++]=34;for    X(i       =          35            +
   (p?0:1);i<b;    i++      )          o
   [w++]=s();o[     w++               ]
    =p?s():34;}     else              X
      {for(i=92;     i<b;             i
       ++)o[w++]=     32;}            }
            else o     [w++          ]
                       =10;o         [
                         w]=0      ;
                          puts(o);}
```

Output 1

```
char*d=")35E!'lcA,,!""2kE`*!-s@d(!(k(f//g&!)f.e5'f(!+a+)f%2g*!?f5f,!=f-*e/!<d"
"6e1!9e0'f3!6f)-g"      "5!4d*b+e6!0f%k)d7!+~^'c7!)z/d-+!'n%aO(d5!""%c1a+/d4"
"!2)c9e2!9b;e1!8b"      ">e/!7cAd-!5fAe+!7fBe(!8hBd&!:iAd$![7S,Q"    "O!1bF7"
"!1b?'_6!1c,8b4!2"      "b*a,*d3!2n4f2!${4f.'!%y4e5!&f%d-^-d7!4"       "c+b)"
"d9!4c-a'd:!/i(``"      "&d;!+l'a+d<!)l*b(d=!'m-a&d>!&d'`0_&c"          "?!"
"$dAc@!$cBc@!$b<^"      ""              "&d$`:!$d9_&l++^$!%f3"          "a'n"
"1_"                                  "$!&f/c(o/_%!(f+c)q*c%"          "!*f&d"
"+f$"                                "s&!-n,d)n(!Oi-c-k)!3d/"        "bOh*!H`"
"7a,![                          "7XO[!4cM,!4cK`*!4cJc(!4cHg&!4"       "c$j8f'!&~"
"]9e)!'"                "|:d+!)""rAc-!*m*:d/!4c(b4e0!1r"        "2e2!/t0e4!-"
"y-c6!+|,c6!)f$b("      "h*c6"      "!(d'b(i)d5!(b*a'"          "`&c)c5!'b+`&"
"b'c)c4!&b-_$c"                     "'d*c3!&a.h'd+d"          "1!%a/g'e+eO!%b"
"-g(d.d/!&c*"                       "h'd1d-!(d%g)"        "d4d+!*l,d7d)!,h-"
"d;c'!.b0c"                            ">d%!A`Dc$"       "![7*i]5471[=ohr&o*"
"t*q*`*d"                               "*v*r;027"       "*~=h./}tcrsth&t:r9"
"b]..b"          ""      "-725"           "-.t--/"       "/#r[<t8-752793?<.~"
";b]."        "t--"      "+r/#"           "537-r"         "[/9~X.v90<6/<.v;-"
"52/="      "{kgoh."     "/}q"    ";u"      "vtohr"       "`.i*$engt$$,b;$/="
"t;v"       ";6=`it."    ";`"     "7=`"      ":,b-7"      "25=/o`..d;b]`--["
"/+"      "55/}o`.d:"    ""       "-?5"       "/}o`.'"      "v/i]q--[;52=`it"
".o"     ";53-.v96<7"             "/=o"       ":d=o--/"       "i]q--[;h./=i]"
"q"     "--[;v9h./<-"           "52="         "{cjuc&`"        "it.o;?4=o:d="
"o"      "--/i]q--["           ";54="        "{cjuc&i]q-"       "-[;76=i]q[;"
"6="       "vsru.i"            "/={"        "=),BihY_gha,)"       "",o[3217];
int i,         r,w          ,f,b,p,        t=641,x;n(){return         r<t?d[(*
d+143+(r              ++))%t]:r>       +1341?59:(x=d[(r++-t)       %351+t]
)?x^(p?6:0          ):(p=+34);}       main(){w=sprintf(o,"char"       "*d="
);r=p=0;for(      f=1;f<*d+143;)      if((b=d[f++])-33){if(b<+93){       if(
!p)o[w++]=34;for(i=35+(p?0:1);i    <b;i++)o[w++]=n();o[w++]=p?n():+34      ;}
else for(i=92;i<b;i++)o[w++]=32;}else o[w++]=10;o[w]=0;puts(o);};/*Don_Yang*/;
```

Last one, just for fun: dhyang IOCCC'00

Saitou Hajime image that prints a prog that prints a prog

Source code

```
                #define/**/X
              char*d="X0[!4cM,!"
            "4cK`*!4cJc(!4cHg&!4c$j"
          "8f'!&~]9e)!']:d+!)rAc-!*m*"
        ":d/!4c(b4e0!1r2e2!/t0e4!-y-c6!"
      "+|,c6!)f$b(h*c6!(d'b(i)d5!(b*a``&c"
      ")c5!'b+`&b'c)c4!&b-_$c'd*c3!&a.h'd+"
    "dl!%a/g'e+e0!%b-g(d.d/!&c*h'dld-!(d%g)"
   "d4d+!*l,d7d)!,h-d;c'!.b0c>d%!A`Dc$![7)35E"
  "!'1cA,,!2kE`*!-s@d(!(k(f//g&!)f.e5'f(!+a+)"
  "f%2g*!?f5f,!=f-*e/!<d6e1!9e0'f3!6f)-g5!4d*b"
  "+e6!0f%k)d7!+~^'c7!)z/d-+!'n%a0(d5!%c1a+/d4"
  "!2)c9e2!9b;el!8b>e/!     7cAd-!5fAe+!7fBe(!"
 "8hBd&!:iAd$![7S,QO!1     bF 7!1b?'_6!1c,8b4"
 "!2b*a,*d3!2n4f2!${4    f.      '!%y4e5!&f%"
 "d-^-d7!4c+b)d9!4c-a     'd        :!/i(``&d"
 ";!+l'a+d<!)l*b(d=!'   m-        a  &d>!&d'"
 "`O_&c?!$dAc@!$cBc@!$   b         <    ^&d$`"
 ":!$d9_&l++^$!%f3a'     n1               $ !&"
 "f/c(o/_%!(f+c)q*c      %!               f &d+"
 "f$s&!-n,d)n(!0i-      c-          k)      !  3d"
 "/bOh*!H`7a,![7*     i]            5       4   71"
"[=ohr&o*t*q*`*d      *v           *r        ;  O2"
"7*~=h./}tcrsth       &t           :         r   9b"
"]..b-725-.t--//      #r           [         <   t8-"
"752793?  <.~;b       ].t--+r      /         #    53"
"7-r[/9~X  .v90      <6/<.v;-52/={              k   goh"
"./}q;    u  vto     hr  `.i*$engt$             $    ,b"
";$/     =t ;v;      6     =`it.`;7=`           :    ,b-"
"725     = / o`.     .d       ;b]`--[/+       55/     }o"
"`.d   :    - ?5    /           }o`.'     v/i]q       - "
"-[;   5  2  =`  it              .        o;53-       . "
"v96   <7 /      =o              :          d         =o"
"--/i  ]q--      [;              h.         /         = "
"i]q--[  ;v      9h              ./         <         - "
"52={cj   u      c&`             i   t      . o       ; "
"?4=o:d=         o--             /  i       ]q        - "
"-[;54={  cj     uc&             i]q         -         -"
"[;76=i]q[;6     =vsr           u.i          /         ={"
"=),BihY_gha     ,)\0            "           ,          o [
 3217];int i,   r,w,f            ,            b        ,x ,
 p;n(){return   r  <X           X             X       X  X
 768?d[X(143+   X  r++          +              *d      )  %
  768]:r>2659   ?  59:          (                x      =  d
  [(r++-768)%   X  947         +              768]      ) ?
  x^(p?6:0):(p  =   34         X                X       X )
  ;}s(){for(x=  n   ();        (               x^       ( p
 ?6:0))==32;x=  n    ()       )   ;return x              ; }
 void/**/main X      ()       {               r        =  p
 =0;w=sprintf  (X     X       X          X X           X o
 ,"char*d=");  for            (    f=1;f <              * d
 +143;)if(33-(  b=d           [      f++ X             ]  )
 ){if(b<93){if   X(!          p           )              o
  [w++]=34;for    X(i         =          35             +
   (p?0:1);i<b;    i++        )          o
   [w++]=s();o[     w++                  ]
    =p?s():34;}     else                X
      {for(i=92;     i<b;               i
       ++)o[w++]=     32;}              }
           else o     [w++             ]
                      =10;o            [
                        w]=0      ;
                         puts(o);}
```

Output 1

```
char*d=")35E!'lcA,,!""2kE`*!-s@d(!(k(f//g&!)f.e5'f(!+a+)f%2g*!?f5f,!=f-*e/!<d"
"6e!9e0'f3!6f)-g"       "5!4d*b+e6!0f%k)d7!+~^'c7!)z/d-+!'n%a0(d5!""%c1a+/d4"
"!2)c9e2!9b;el!8b"      ">e/!7cAd-!5fAe+!7fBe(!8hBd&!:iAd$![7S,Q"    "0!1bF7"
"!1b?'_6!1c,8b4!2"      "b*a,*d3!2n4f2!${4f.'!%y4e5!&f%d-^-d7!4"       "c+b)"
"d9!4c-a'd:!/i(``"      "&d;!+l'a+d<!)l*b(d=!'m-a&d>!&d'`0_&c"          "?!"
"$dAc@!$cBc@!$b<^"      ""                "&d$`:!$d9_&l++^$!%f3"          "a'n"
"1_"                                    "$!&f/c(o/_%!(f+c)q*c%"          "!*f&d"
"+f$"                                   "s&!-n,d)n(!0i-c-k)!3d/"        "b0h*!H`"
"7a,![                          "7X0[!4cM,!4cK`*!4cJc(!4cHg&!4"         "c$j8f'!&~"
"]9e)!'"                        "|:d+!)""rAc-!*m*:d/!4c(b4e0!1r"        "2e2!/t0e4!-"
"y-c6!+|,c6!)f$b("              "h*c6"      "!(d'b(i)d5!(b*a'"           "&)c5!'b+`&"
"b'c)c4!&b-_$c"                             "`d*c3!&a.h'd+d"        "1!%a/g'e+e0!%b"
"-g(d./!&c*"                                "h'dld-!(d%g)"       "d4d+!*l,d7d)!,h-"
"d;c'!.b0c"                                    ">d%!A`Dc$"       "![7*i]5471[=ohr&o*"
"t*q*`*d"                                      "*v*r;027"        "*~=h./}tcrsth&t:r9"
"b]..b"          ""      "-725"              "-.t--/"        "/#r[<t8-752793?<.~"
";b]."          "t--"     "+r/#"             "537-r"        "[/9~X.v90<6/<.v;-"
"52/="      "{kgoh."     "/}q"     ";u"       "vtohr"       "`.i*$engt$$,b;$/=["
"t;v"       ";6=`it."   "`;"      "7=`"       ":,b-7"      "25=/o`..d;b]`--["
"/+"      "55/}o`.d:"   ""        "-?5"       "/}o`.'"       "v/i]q--[;52=`it"
".o"      ";53-.v96<7"              "/=o"       ":d=o--/"       "i]q--[;h./=i]"
"q"     "--[;v9h./<-"           "52="         "{cjuc&`"        "it.o;?4=o:d="
"o"       "--/i]q--["           ";54="        "{cjuc&i]q-"      "-[;76=i]q[;"
"6="       "vsru.i"             "/={"         "=),BihY_gha,)"       "",o[3217];
int i,        r,w          ,f,b,p,        t=641,x;n(){return        r<t?d[(*
d+143+(r              ++))%t];r>       +1341?59:(x=d[(r++-t)       %351+t]
)?x^(p?6:0          ):(p=+34);}       main(){w=sprintf(o,"char"       "*d="
);r=p=0;for(      f=1;f<*d+143;)      if((b=d[f++])-33){if(b<+93){       if(
!p)o[w++]=34;for(i=35+(p?0:1);i    <b;i++)o[w++]=n();o[w++]=p?n():+34      ;}
else for(i=92;i<b;i++)o[w++]=32;}else o[w++]=10;o[w]=0;puts(o);};/*Don_Yang*/;
```

Output 2

```
char*d="S,Q0!1bF7!1b?'_""6!1c,8b4!2b*a,*d3""!2n4f2!${4f.'!%y4e5!&f%d-^-d7!4c+"
"b)d""9!4c-a'd:!/i("      "'`&d;!+l'a+d<!)l*b(d=!'m-a&d>""!&d``O_&""c?!$dAc@"
"!$cBc@!$b<^&d$"              "`:!$d9_&l++^$!%f3a'nl_$!&f/c(o/_%"    "!(f+c)"
"q*c%!*f&d"                          "+f$s&!-n,d)n(!0i-c-k)!3d/b0h"          "*!H`"
"7a,!"                    "[!4cM,!4cK`""*!4cJc(!4cH"                  "g&"
"!4c$j"          "8f'!&~]9e)"         "!']:d+!)rAc-!*m*:""d/!"          "4c(b"
"4e0!1r2"     "e2!/t0e""4!-y-"          "c""6!+|,c6!)f$b(h"           "*c6!(d"
"'b(i)d5!(b*a'`&c)c5!'b+`&b'"          "c)c4!&b-_$c'd*c3!"          "&a.h'd+d"
"l!%a/g'e+e0!%b-g(d.d/!&c*"          "h'dld-!(d""%g)d4d"         "+!*l,d7d)!,"
"h-d;c'!.b0c>d%!A`Dc$![7)"        "35E!'lcA,,!2kE`*!-"         "s@d(!(k(f//g&"
"!)f.e5'f(!+a+)f%2g*!?"          "f5f,!=f-*e/!""<d6"          "el!9e0'f3!6f)-g"
"5!4d*b+e6!0f%k)d7!"           "+~^'c""7!)z/d-+!"           "'n%a0(d5!%c1a+/d4"
"!2)c9e2!9b;el!8b"        ">e/!7c"       "Ad-!5fA"          "e+!7fBe(!8hBd&!:iA"
"d$![7*i]5471"          "["              "=ohr&"          "o*t*q*`*d*v*r;027*~"
"=h./}tc"                                  "rst"        "h&t:r9b]..b-725-.t-"
"-//#r"                          "[<t8-752793"        "?<.~;b].t""--+r/#5"
"37-"                  "r"     "[/9~X.v90<6/""<.v;"        "-52/={kgoh./}q;uv"
"t"        "ohr`.i*$engt$"     "$,b;$/=""t;v;6=`it.`"        ";7=`:,b-725=/o`."
".d;b]`--[/+55/""}o`.d"       ":-?5/}o`.'v/i]q--[;52"         "=`it.o;53-.v96"
"<7/=o:d=o--/i]q--[;h."      "/=i]q--[;v9h./<-52={cju"         "c&`it.o;?4=o:"
"d=o--/i]q--[;54={cju"      "c&i]q--[;76=i]q[;6=vsru.i/"         "={=),BihY_g"
"ha,)",o[3217];int i,       r,w,f,b,p,t=641,x;n(){return r<        t?d[(*d+143
+(r++))%t]:r>+1341?          59:(x=d[(r++-t)%351+t])?x^(p?6:          0):(p=+34
);}main(){w=sprintf(o          ,"char""*d=");r=p=0;for(f=1;f<*d          +143;)
if((b=d[f++])-33){if(b           <+93){if(!p)o[w++]=34;for(i=35+(         p?0:
1);i<b;i++)o[w++]=n();o[            w++]=p?n():+34;}else for(i=92;i          <b
;i++)o[w++]=32;}else o[w++]=10;o[w]=0;puts(o);};/*Don_Yang*/;;;;;;;;;;;;;;;;;
```

Last one, just for fun: dhyang IOCCC'00

Saitou Hajime image that prints a prog that prints a prog that prints a prog

Source code

```
                #define/**/X
              char*d="X0[!4cM,!"
            "4cK`*!4cJc(!4cHg&!4c$j"
          "8f'!&~]9e)!']:d+!)rAc-!*m*"
        ":d/!4c(b4e0!1r2e2!/t0e4!-y-c6!"
      "+|,c6!)f$b(h*c6!(d'b(i)d5!(b*a`&c"
      ")c5!'b+`&b'c)c4!&b-_$c'd*c3!&a.h'd+"
    "dl!%a/g'e+e0!%b-g(d.d/!&c*h'd1d-!(d%g)"
   "d4d+!*l,d7d)!,h-d;c'!.b0c>d%!A`Dc$![7)35E"
  "!'1cA,,!2kE`*!-s@d(!(k(f//g&!)f.e5'f(!+a+)"
  "f%2g*!?f5f,!=f-*e/!<d6e1!9e0'f3!6f)-g5!4d*b"
  "+e6!0f%k)d7!+~^'c7!)z/d-+!'n%a0(d5!%c1a+/d4"
  "!2)c9e2!9b;el!8b>e/!     7cAd-!5fAe+!7fBe(!"
 "8hBd&!:iAd$![7S,QO!l     bF 7!lb?'_6!1c,8b4"
 "!2b*a,*d3!2n4f2!${4    f.      '!%y4e5!&f%"
 "d-^-d7!4c+b)d9!4c-a    'd       :!/i(``&d"
 ";!+l'a+d<!)l*b(d=!'   m-        a  &d>!&d'"
 "`O_&c?!$dAc@!$cBc@!$   b         <    ^&d$`"
 ":!$d9_&l++^$!%f3a'     n1               $ !&"
 "f/c(o/_%!(f+c)q*c     %!              f &d+"
 "f$s&!-n,d)n(!Oi-      c-          k)      !  3d"
 "/bOh*!H`7a,![7*     i]            5       4   71"
"[=ohr&o*t*q*`*d      *v           *r         ;  O2"
"7*~=h./}tcrsth       &t           :          r   9b"
"],.b-725-.t--//      #r           [          <   t8-"
"752793?  <.~;b       ].t--+r      /          #    53"
"7-r[/9~X  .v90      <6/<.v;-52/={              k   goh"
"./}q;    u  vto     hr   `.i*$engt$            $    ,b"
";$/     =t ;v;     6     =`it.`;7=`            :    ,b-"
"725     = / o`.    .d       ;b]`--[/+        55/     }o"
"`.d   :    - ?5    /           }o`.'      v/i]q       - "
"-[;   5  2  =`  it              .         o;53-        . "
"v96   <7 /      =o              :           d          =o"
"--/i  ]q--      [;              h.          /          = "
"i]q--[  ;v      9h              ./          <          - "
"52={cj   u      c&`             i   t       . o        ; "
"?4=o:d=         o--             /  i        ]q         - "
"-[;54={  cj     uc&             i]q          -          -"
"[;76=i]q[;6     =vsr            u.i          /          ={"
"=),BihY_gha     ,)\0            "            ,          o [
 3217];int i,    r,w,f           ,            b         ,x ,
 p;n(){return    r  <X           X            X        X  X
 768?d[X(143+    X  r++          +            *d       )  %
  768];r>2659    ?  59:          (            x        =  d
  [(r++-768)%    X  947        +                768]     ) ?
  x^(p?6:0):(p   =   34       X                 X       X )
  ;}s(){for(x=   n   ();      (              x^         ( p
 ?6:0))==32;x=   n    ()     )    ;return x              ; }
 void/**/main X       ()     {              r          =  p
 =0;w=sprintf   (X      X     X          X X            X o
 ,"char*d=");   for           (    f=1;f <              * d
 +143;)if(33-(  b=d           [      f++ X             ]  )
 ){if(b<93){if    X(!         p           )              o
  [w++]=34;for     X(i        =           35             +
   (p?0:1);i<b;    i++        )           o
   [w++]=s():o[     w++                    ]
    =p?s():34;}     else                   X
     {for(i=92;      i<b;                  i
      ++)o[w++]=      32;}                 }
          else o      [w++                 ]
                      =10;o                [
                        w]=0      ;
                         puts(o);}
```

Output 1

```
char*d=")35E!'lcA,,!""2kE`*!-s@d(!(k(f//g&!)f.e5'f(!+a+)f%2g*!?f5f,!=f-*e/!<d"
"6e1!9e0'f3!6f)-g"      "5!4d*b+e6!0f%k)d7!+~^'c7!)z/d-+!'n%a0(d5!""%c1a+/d4"
"!2)c9e2!9b;el!8b"      ">e/!7cAd-!5fAe+!7fBe(!8hBd&!:iAd$![7S,Q"    "0!lbF7"
"!lb?'_6!1c,8b4!2"      "b*a,*d3!2n4f2!${4f.'!%y4e5!&f%d-^-d7!4"       "c+b)"
"d9!4c-a'd:!/i(``"      "&d;!+l'a+d<!)l*b(d=!'m-a&d>!&d'`O_&c"          "?!"
"$dAc@!$cBc@!$b<^"      ""          "&d$`:!$d9_&l++^$!%f3"          "a'n"
"1_"                                "$!&f/c(o/_%!(f+c)q*c%"          "!*f&d"
"+f$"                               "s&!-n,d)n(!Oi-c-k)!3d/"       "b0h*!H`"
"7a,![                    "7X0[!4cM,!4cK`*!4cJc(!4cHg&!4"       "c$j8f'!&~"
"]9e)!'"                "|;d+!)""rAc-!*m*:d/!4c(b4e0!1r"       "2e2!/t0e4!-"
"y-c6!+|,c6!)f$b("      "h*c6"      "!(d'b(i)d5!(b*a'"          "&c)c5!'b+`&"
"b'c)c4!&b-_$c"                     "'d*c3!&a.h'd+d"        "1!%a/g'e+e0!%b"
"-g(d./!&c*"                        "h'd1d-!(d%g)"       "d4d+!*l,d7d)!,h-"
"d;c'!.b0c                             ">d%!A`Dc$"      "![7*i]5471[=ohr&o*"
"t*q*`*d                               "*v*r;027"      "*~=h./}tcrsth&t:r9"
"b]..b"          ""     "-725"          "-.t--/"      "/#r[<t8-752793?<.~"
";b]."          "t--"    "+r/#"          "537-r"       "[/9~X.v90<6/<.v;-"
"52/="      "{kgoh."     "/}q"    ";u"      "vtohr"      "`.i*$engt$$,b;$/="
"t;v"        ";6=`it."   ";"      "7=`"      ":,b-7"     "25=/o`..d;b]`--["
"/+"       "55/}o`.d:"   ""       "-?5"       "/}o`.'"     "v/i]q--[;52=`it"
".o"       ";53-.v96<7"            "/=0"         ":d=o--/"      "i]q--[;h./=i]"
"q"      "--[;v9h./<-"           "52="          "{cjuc&`"       "it.o;?4=o:d="
"o"        "--/i]q--["           ";54="         "{cjuc&i]q-"      "-[;76=i]q[;"
"6="       "vsru.i"              "/={"          "=),BihY_gha,)"       "",o[3217];
int i,        r,w          ,f,b,p,       t=641,x;n(){return       r<t?d[(*
d+143+(r              ++))%t]:r>       +1341?59:(x=d[(r++-t)       %351+t]
)?x^(p?6:0          ):(p=+34);}       main(){w=sprintf(o,"char"       "*d="
);r=p=0;for(      f=1;f<*d+143;)       if((b=d[f++])-33){if(b<+93){       if(
!p)o[w++]=34;for(i=35+(p?0:1);i    <b;i++)o[w++]=n();o[w++]=p?n():+34      ;}
else for(i=92;i<b;i++)o[w++]=32;}else o[w++]=10;o[w]=0;puts(o);};/*Don_Yang*/;
```

Output 2

```
char*d="S,QO!lbF7!lb?'_""6!1c,8b4!2b*a,*d3""!2n4f2!${4f.'!%y4e5!&f%d-^-d7!4c+"
"b)d""9!4c-a'd:!/i("      "'`&d;!+l'a+d<!)l*b(d=!'m-a&d>""!&d``O_&""c?!$dAc@"
"!$cBc@!$b<^&d$"            "`:!$d9_&l++^$!%f3a'n1_$!&f/c(o/_%"    "!(f+c)"
"q*c%!*f&d"                          "+f$s&!-n,d)n(!Oi-c-k)!3d/b0h"          "*!H`"
"7a,!"                    "[!4cM,!4cK`""*!4cJc(!4cH"          "g&"
"!4c$j"          "8f'!&~]9e)"        "!']:d+!)rAc-!*m*:""d/!"          "4c(b"
"4e0!1r2"     "e2!/t0e""4!-y-"          "c""6!+|,c6!)f$b(h"          "*c6!(d"
"'b(i)d5!(b*a'`&c)c5!'b+`&b'"          "c)c4!&b-_$c'd*c3!"          "&a.h'd+d"
"!!%a/g'e+e0!%b-g(d.d/!&c*"          "h'd1d-!(d""%g)d4d"          "+!*l,d7d)!,"
"h-d;c'!.b0c>d%!A`Dc$![7)"          "35E!'lcA,,!2kE`*!-"          "s@d(!(k(f//g&"
"!)f.e5'f(!+a+)f%2g*!?"          "f5f,!=f-*e/!""<d6"          "el!9e0'f3!6f)-g"
"5!4d*b+e6!0f%k)d7!"          "+~^'c""7!)z/d-+!"          "'n%a0(d5!%c1a+/d4"
"!2)c9e2!9b;el!8b"          ">e/!7c"       "Ad-!5fA"          "e+!7fBe(!8hBd&!:iA"
"d$![7*i]5471"          "["          "=ohr&"          "o*t*q*`*d*v*r;027*~"
"=h./}tc"                          "rst"          "h&t:r9b]..b-725-.t-"
"-//#r"                    "[<t8-752793"          "?<.~;b].t""--+r/#5"
"37-"                "r"     "[/9~X.v90<6/""<.v;"          "-52/={kgoh./}q;uv"
"t"          "ohr`.i*$engt$"     "$,b;$/=""t;v;6=`it.`"          ";7=`:,b-725=/o`."
".d;b]`--[/+55/""}o`.d"          ":-?5/}o`.'v/i]q--[;52"          "=`it.o;53-.v96"
"<7/=o:d=o--/i]q--[;h."          "/=i]q--[;v9h./<-52={cju"          "c&`it.o;?4=o:"
"d=o--/i]q--[;54={cju"          "c&i]q--[;76=i]q[;6=vsru.i/"          "={=),BihY_g"
"ha,)",o[3217];int i,       r,w,f,b,p,t=641,x;n(){return r<          t?d[(*d+143
+(r++))%t]:r>+1341?          59:(x=d[(r++-t)%351+t])?x^(p?6:          0):(p=+34
);}main(){w=sprintf(o          ,"char""*d=");r=p=0;for(f=1;f<*d          +143;)
if((b=d[f++])-33){if(b          <+93){if(!p)o[w++]=34;for(i=35+(          p?0:
1);i<b;i++)o[w++]=n();o[          w++]=p?n():+34;}else for(i=92;i          <b
;i++)o[w++]=32;}else o[w++]=10;o[w]=0;puts(o);};/*Don_Yang*/;;;;;;;;;;;;;;;;;
```

Output 3

```
char*d="X0[!4cM,""!4cK`*!4cJc(!4cHg&!4c$j8f'!&~]9e)!']:d+!)rAc-""!*m*:d/!4c(b"
"4e0!1r2e2!/t0"      "e4!-y-c6!+|,c6!)f$b(h*c6!(d'b(i)d5""!(b*a'`&c)c5!'b+`&b"
"'c)c4!&b-_$c'"      "d*c3!&a.h'd+d1!%a/g'e+e0!%b""-g("    "d./!&c*h'd1d-!(d%"
"g)d4d+!*l,d7d"      ")!,h-d;c""'!.b0c>d%!A`Dc$![7)3"      "5E!'lcA,,!2kE`*!"
"-s@d(!(k(f//g&"      "!)f.e5"      "'f(!+a+)""f%2g*!"          "?f5f,!=f-*e/!<d"
"6e1!9e0'f3!6f)"                  "-g5!4d*b+e6!0f%k"          ")d7!+~^'c7!)z/"
""                              "d-+!'n%a0(d5!%c1"          "a+/d4!2)c9""e2!"
"9"                           "b;el!8b>e/!7cAd-"          "!5fAe+!7fBe(!8hBd"
"&!"          ":"       "iAd$![7S,"   "QO!lbF7!1"          "b?'_6!1c,8b4!2b*a,*"
"d3!2n4f2!${4f.'!"      "%y4e5!&"      "f%d-^"          "-d7!4c+b)d9!4c-a'd:!/"
"i(``&d;!+l'a+d<!"      ")l*b(d=!'"    "m-a"          "&d>!&d'`O_&c?!$dAc@!$c"
"Bc@!$b<^&d$"            "`:!$""d9_"    "&l"          "++^$!%f3a'n1_$!&f/c(o/_"
"%!(f+c)"                "q*c"      "%!*f&d+"          "f$s&!-n,d)n(!Oi-c-k)!3d/"
"b0h*!"                "H`7a,!"      "[7*i"          "]5471[=ohr&o*t*q*`*d*v*r;"
"027"                "*~=h./}tc"     "rs"          "th&t:r9b]..b-725-.t--//#r["
"<t"          "8-7"     "527937<.~;b]"    ".t"          "--+r/#537-r[/9~X.v90<6/<.v;"
""          "-52/={kgoh./}q;uvtohr`.i*$eng"          "t$$,b;$/=t;v;6=`it.`;7=`:,b-"
""        "725=/o`..d;b]`--[/+55/}o`.d:-?"          "5/}o`.'v/i]q--[;52=`it.o;53-"
""      ".v96<7/=o:d=o--/i]q--[;h"    "./"          ""      "=i]q--[;v9h./<-52={cju"
""        "c&`it.o;?4=o:d=o--/i]"    "q-"          "-[;54={""cjuc&i]"    ""
"q"            "--[;76=i]q[;6=v"      "sru"          ".i/={=),BihY_"    ""
"gh"           "a,)",o[3217]      ;int i          ,r,w,f,b,p,t=    641
,x;n()          {return r       <t?d[(*          d+143+(r       ++)
)%t]:r>+        1341       ?59:(x=d[          (r          ++-t
)%351+t])?x          ^(p?6:0):(       p=+34);          }main(
){w=sprintf(o,          "char""*d="       );r=p=0;for          (f=1;f<
*d+143;)if((b=d[f       ++])-33){if(b      <+93){if(!p)o[          w++]=34;
for(i=35+(p?0:1);i<b;i++)o[w++]=n();o[    w++]=p?n():+34;}else     for(i=92;i
<b;i++)o[w++]=32;}else o[w++]=10;o[w]=0;puts(o);};/*Don_Yang*/;;;;;;;;;;;;;;;;;
```

Last one, just for fun: dhyang IOCCC'00

Saitou Hajime image that prints a prog that prints a prog that prints a prog ...
Repeating endlessly "aku soku zan", Hajime's mode meaning *slay evil imediatly*.

Source code

Output 1

Output 2

Output 3

Output 4 (=1)

Conclusion

Computer Science is the Science of Abstraction

- Computer Scientists are engineers terraforming ideas and concepts
- Write code to communicate with humans, and accidentally to execute it
- Programming is not about technology for its own sake. It's about being able to express your ideas precisely and efficiently. `http://prog21.dadgum.com/`

FP vs. OOP: how you prefer to state your ideas?

- OOP is all about nouns, FP is all about verbs
- Please keep the troll level low: It's the mutable state that is evil, not the object

Don't hope to get it right on the first time

- To a great extent the act of coding is one of organization. Refactoring. Simplifying. Figuring out how to remove extraneous manipulations here and there.
- Write it. Rewrite it correct. Rewrite it efficient. Rewrite it modifiable / elegant.

This course is now over

- I really hope that you will like your long journey on the programmer path