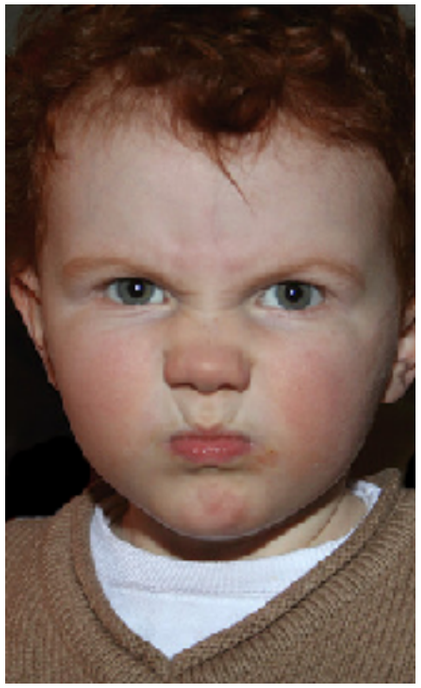
r7k

Unconditional Code

Michael Feathers
R7K Research & Conveyance

$$n + 4$$

Logging



```
}catch MyError.AnError {
    print("AnError")
}catch MyError.AnotherError {
    print("AnotherError")  //AnotherError will be catched and printed
}catch{
    print("Something else happened")
}

do{
    do{
        try throwsError()
    }catch MyError.AnError {
        print("AnError")
    }
}catch MyError.AnotherError {
    print("AnotherError")  //AnotherError will be catched and printed
```

What if noticeable error handling is a symptom of bad design?

Review Your Order

Look this over. If it all looks right, click the "place your order" button to finish buying your stuff.

SHIPPING & PAYMENT

pay **Address Book**

To complete the Amazon Pay checkout you must accept third-party cookies in the options/settings of your internet browser. After you have accepted third-party cookies please refresh this page to continue check out.

Amazon Pay
Privacy

pay **Payment Method**

Your session has expired. Please sign in again by clicking on the Amazon Pay Button.

Amazon Pay
Privacy

Code should just run - unconditionally

Deep dive..

```java
public Item itemForBarcode(String barcode) {
    Item item = items.get(barcode);
    if (item != null)
        return item;
    return null;
}
```

```java
public Item itemForBarcode(String barcode) throws ItemNotFound {
    Item item = items.get(barcode);
    if (item == null)
        throw new ItemNotFound(barcode);
    return item;
}
```

"Use exceptions when you can't know in advance whether a call will succeed or fail."

Bertrand Meyer

```java
public void populateSigns(List<SignProvider> providers) {
    // ...
}
```

```java
public void populateSigns(List<SignProvider> providers) {
    // ...
}
```

```java
public void run() {
    // ...
    populateSigns(null);
    //...
}
```

```java
public void formReport() {
    boolean hasMarkedAccounts = findMarkings();
    boolean inTallyPeriod = CalculationPeriods.hasTallyDate(currentDate);

    // ... 1000 lines here

    if (hasMarkedAccounts && !inTallyPeriod) {
        // ...
    }

}
```

Tunneling

Is interpretation the problem?

The string is a stark data structure and everywhere it is passed there is much duplication of process. It is a perfect vehicle for hiding information.

Alan Perlis

Can We Eliminate Tunnels?

```java
public Item itemForBarcode(String barcode) {
    return items.getOrDefault(barcode, new Item("Item not found", 0));
}
```

```java
public Item itemForBarcode(String barcode) {
    return items.getOrDefault(barcode, new Item("Item not found", 0));
}
```

NaN

[] null object

special case

```java
public interface SaleListener {
    void itemAdded(Item item);
    void saleTotaled(int total);

}
```

```java
public interface SaleListener {
    void itemAdded(Item item);
    void saleTotaled(int total);
    void itemNotFound(String barcode);
}
```

```
data SaleAction = ItemAdded Item
                | SaleTotaled Int
                | ItemNotFound String
```

Tell, Don't Ask

Alec Sharp, in the recent book *Smalltalk by Example* [SHARP], points up a very valuable lesson in few words:

> *Procedural code gets information then makes decisions. Object-oriented code tells objects to do things.*
> — **Alec Sharp**

r7k

```
Person person = source.getPerson(id);

if (person != null) {
    person.reSelect(date);
}
```

r7k

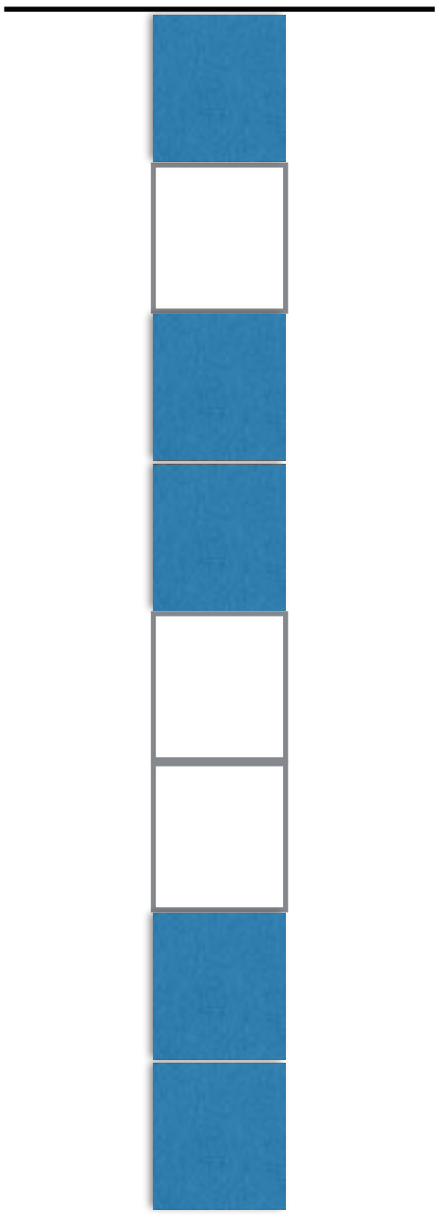
```
source.withPerson(id, (Person p) -> p.reSelect(date));
```

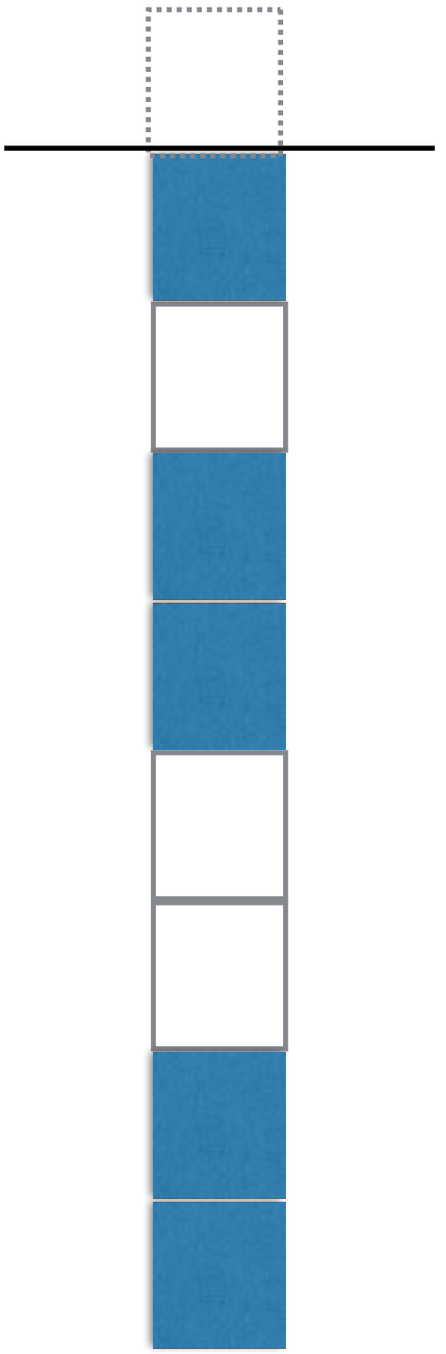
Asking for data can fail.
Giving it when you have it can not.

Extend the Domain

<u>Domain</u> means something in mathematics

```ruby
def span_count ary
  return 0 if ary.size == 0
  count = 0
  if ary[0] > 0
    count = 1
  end
  i = 0
  while i < ary.size - 1
    if ary[i] == 0 && ary[i+1] != 0
      count = count + 1
    end
    i  = i + 1
  end
  return count
end
```





```ruby
# return the number of contiguous spans of non-zeros in an array
#
# :: [Integer] -> Integer
def span_count ary
  ([0] + ary).lazy
             .each_cons(2)
             .count{|c,n| c == 0 && n != 0 }
end
```

```
0--5--3--2--0-----1--0-----------------0-----------------------0--1--0--------------
---------------3-----------------1--3-----------------------------------3--0--------
-----------------------7--2--2-----------2--4--0-----------------------------0--2--
-------------------------------------------------2--0-----------------------------
-------------------------------------------------------1--0----------------------
-----------------------------------------------------------------------------------
```

```
0--5--3--2--0-----1--0-----------------0-----------------------0--1--0--------------
---------------3-----------------1--3-----------------------------------3--0--------
-----------------------7--2--2-----------2--4--0-----------------------------0--2--
-------------------------------------------------2--0-----------------------------
-------------------------------------------------------------1--0-----------------
-------------------------------------------------------------------------------------
```

```
1 0
1 5
1 3
1 2
1 0
2 3
1 1
1 0
3 7
3 2
3 2
2 1
2 3
1 0
3 2
3 4
3 0
```

```
0--5--3--2--0-----1--0-----------------0----------------------0--1--0--------------
---------------3-----------------1--3-----------------------------3--0--------
-----------------------7--2--2-----------2--4--0----------------------------0--2--
-------------------------------------------------2--0-----------------------------
-----------------------------------------------------1--0----------------------
-------------------------------------------------------------------------------
```

1	0
1	5
1	3
1	2
1	0
2	3
1	1
1	0
3	7
3	2
3	2
2	1

```
1 0
1 5
1 3
1 2
1 0
2 3
1 1
1 0
3 7
3 2
3 2
2 1
2 3
1 0
3 2
3 4
3 0
4 2
4 0
5 1
5 0
1 0
1 1
1 0
2 3
2 0
3 0
3 2
```

0. Command line argument for the filename may be missing
1. Unable to open an input file
2. File is empty
3. File contains empty lines
4. Our input file is not a text file
5. A line has more than two numbers
6. A line has less than two numbers
7. A line has fields that can not be parsed as numbers
8. The string number is less than one or more than six
9. The fret number is less than zero or more than twenty-four

```
STRING_COUNT = 6

def tab_column string, fret
  ["---"             ] * (string - 1) +
  [fret.ljust(3,'-')] +
  ["---"             ] * (STRING_COUNT - string)
end

unless File.exist? ARGV[0]
  abort "Unable to open #{ARGV[0]}"
end

File.open(ARGV[0],"r") do |f|
  puts f.each_line
        .map(&:split)
        .map {|string,fret| tab_column(string.to_i, fret) }
        .transpose
        .map(&:join)
        .join($/)
end
```

```
STRING_COUNT = 6

def tab_column string, fret
  ["---"              ] * (string - 1) +
  [fret.ljust(3,'-')] +
  ["---"              ] * (STRING_COUNT - string)
end

unless File.exist? ARGV[0]
  abort "Unable to open #{ARGV[0]}"
end

File.open(ARGV[0],"r") do |f|
  puts f.each_line
        .map(&:split)
        .map {|string,fret| tab_column(string.to_i, fret) }
        .transpose
        .map(&:join)
        .join($/)
end
```

Hmmm..

```
STRING_COUNT = 6

def tab_column string, fret
  ["---"              ] * (string - 1) +
  [fret.ljust(3,'-')] +
  ["---"              ] * (STRING_COUNT - string)
end

puts ARGF.each_line
         .map(&:split)
         .map {|string,fret| tab_column(string.to_i, fret) }
         .transpose
         .map(&:join)
         .join($/)
```

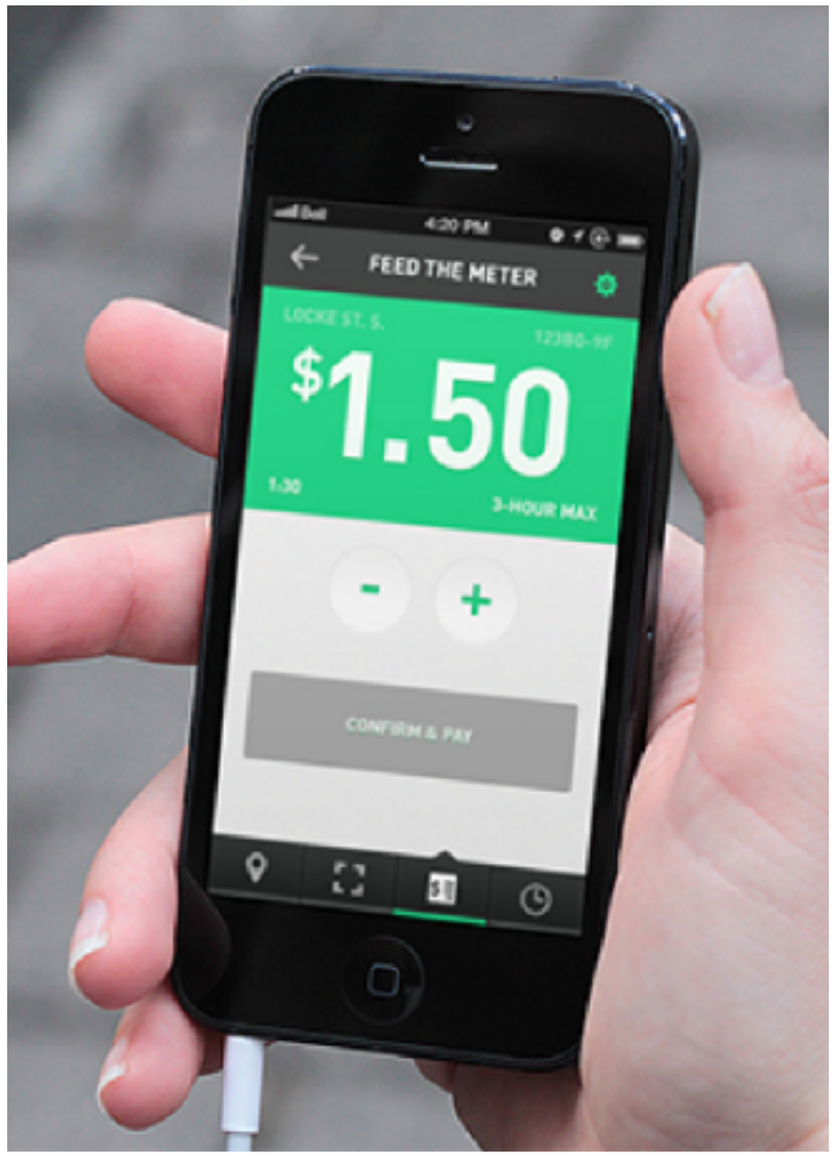
0. Command line argument for the filename may be missing
1. Unable to open an input file
2. File is empty
3. File contains empty lines
4. Our input file is not a text file
5. A line has more than two numbers
6. A line has less than two numbers
7. A line has fields that can not be parsed as numbers
8. The string number is less than one or more than six
9. The fret number is less than zero or more than twenty-four

1	
1	5
1	3
1	2
1	
2	3

```
#include<cmath.h>
double sqrt (double x );
```

Indices may also be negative numbers, to start counting from the right:

```
>>> word[-1]  # last character
'n'
>>> word[-2]  # second-last character
'o'
>>> word[-6]
'P'
```

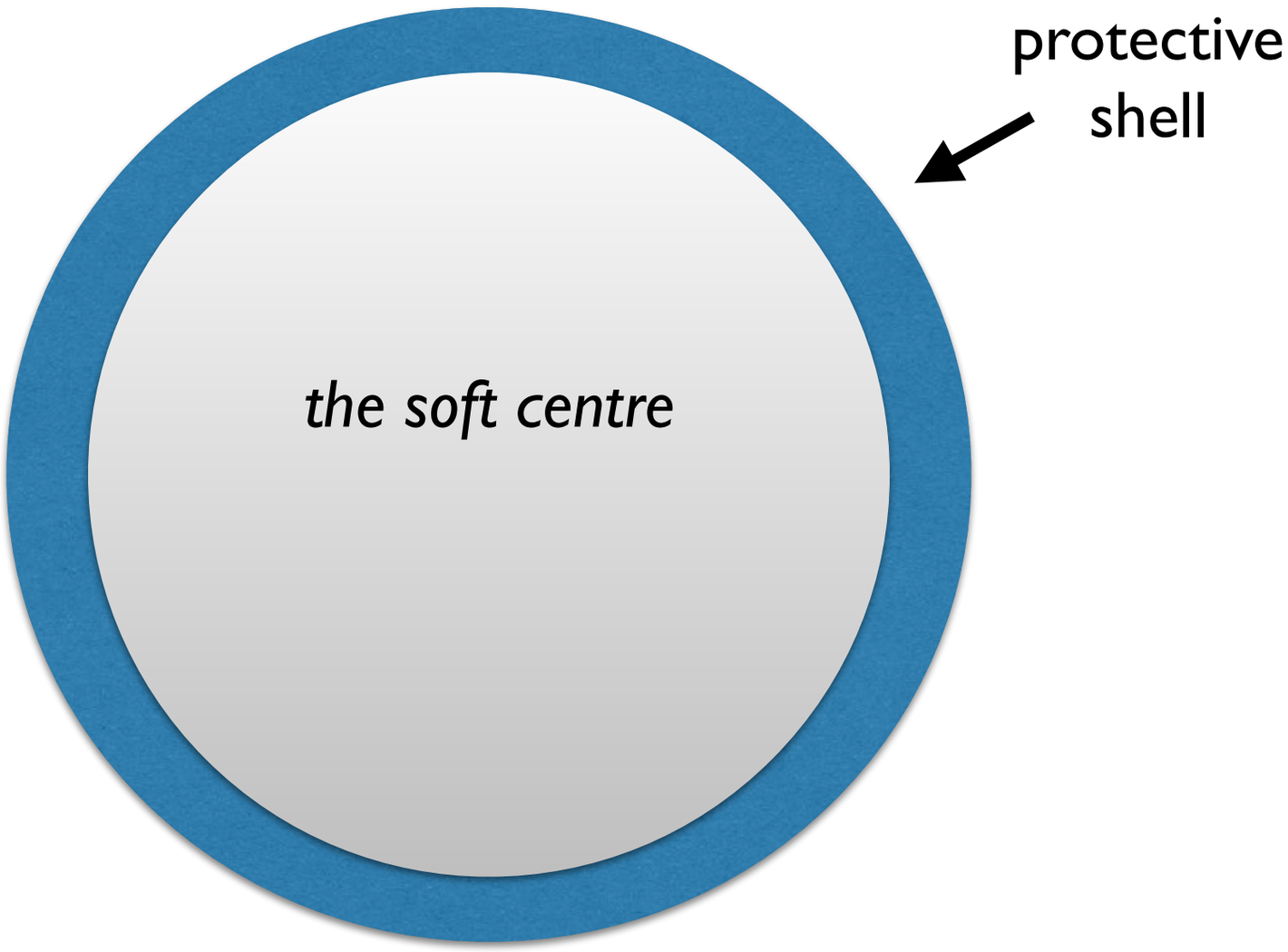

4:20 PM
FEED THE METER
LOCKE ST. S.
$1.50
1:30
3-HOUR MAX
CONFIRM & PAY

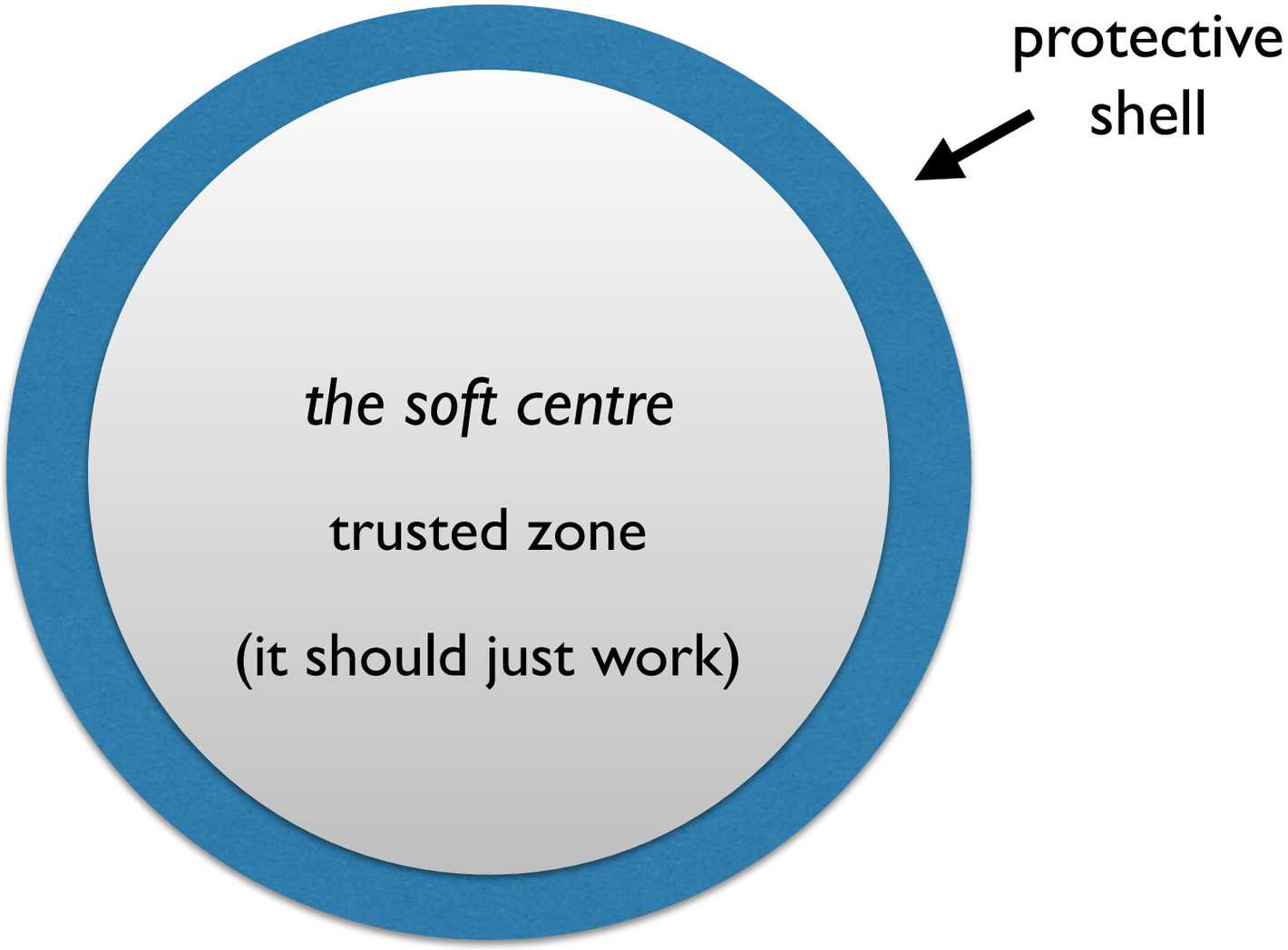
Extend the domain when the extension can be considered the new domain

✗ 0. Command line argument for the filename may be missing
✗ 1. Unable to open an input file
✗ 2. File is empty
3. File contains empty lines
✗ 4. Our input file is not a text file
5. A line has more than two numbers
6. A line has less than two numbers
7. A line has fields that can not be parsed as numbers
8. The string number is less than one or more than six
9. The fret number is less than zero or more than twenty-four

```ruby
STRING_COUNT = 6

def tab_column string, fret
  ["---"            ] * (string - 1) +
  [fret.ljust(3,'-')] +
  ["---"            ] * (STRING_COUNT - string)
end

lines = ARGF.each_line
            .select {|l| l =~ /\S/ }
            .map(&:split)


check("each line should have two fields") do |line_fields|
  line_fields.count == 2
end

check("all fields should be integers") do |string, fret|
  converts_to_int(string) && converts_to_int(fret)
end

check("strings should be in the range 1..6") do |string,_|
  string >= 1 && string <= 6
end

puts lines.each_line
          .map {|string,fret| tab_column(string.to_i, saturate(fret.to_i,(0..99)) }
          .transpose
          .map(&:join)
          .join($/)
```

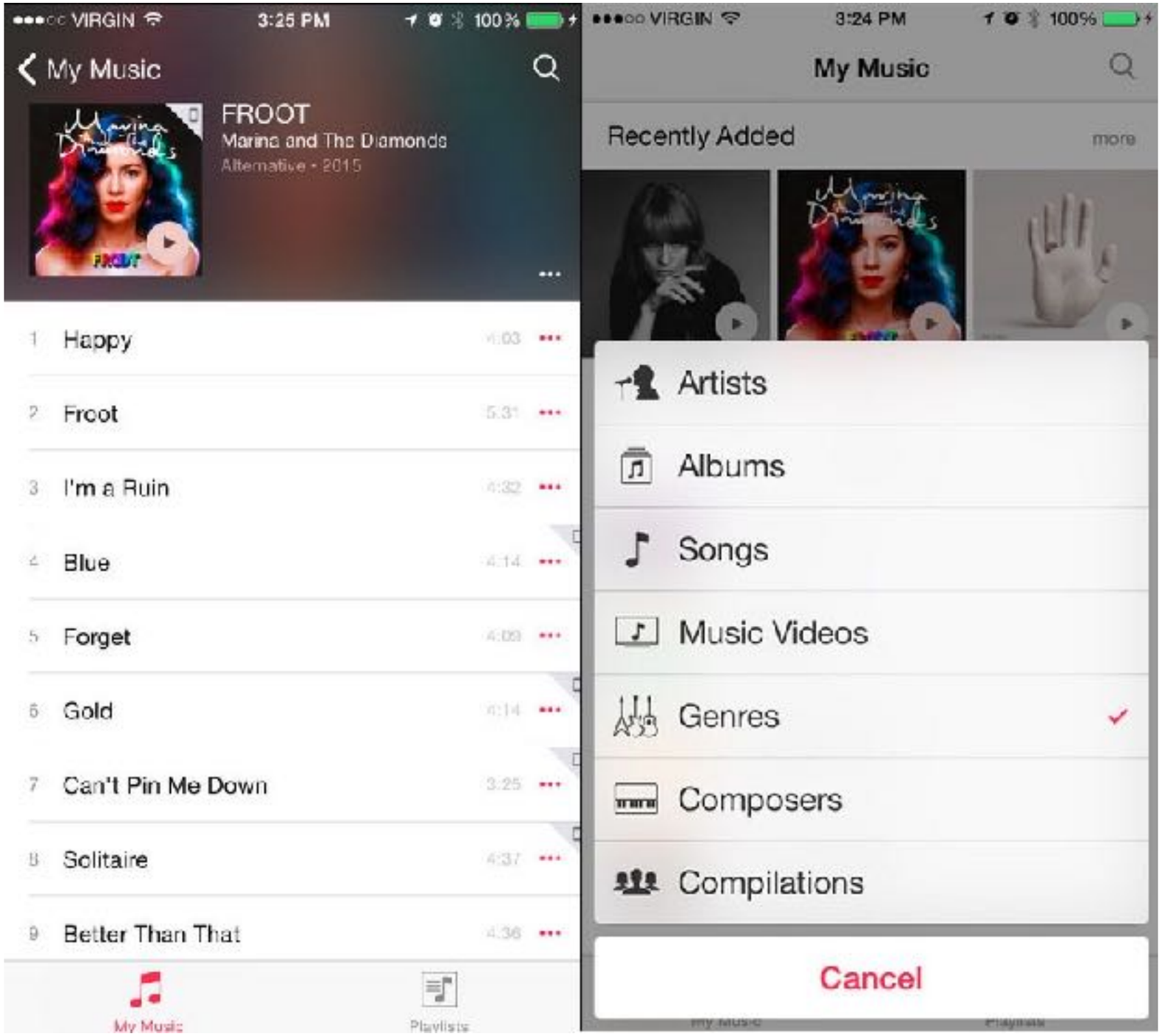
```
STRING_COUNT = 6

def tab_column string, fret
  ["---"            ] * (string - 1) +
  [fret.ljust(3,'-')] +
  ["---"            ] * (STRING_COUNT - string)
end

lines = ARGF.each_line
            .select {|l| l =~ /\S/ }
            .map(&:split)


check("each line should have two fields") do |line_fields|
  line_fields.count == 2
end

check("all fields should be integers") do |string, fret|
  converts_to_int(string) && converts_to_int(fret)
end

check("strings should be in the range 1..6") do |string,_|
  string >= 1 && string <= 6
end

puts lines.map {|string,fret| tab_column(string.to_i, saturate(fret.to_i,(0..99)) }
     .transpose
     .map(&:join)
     .join($/)
```

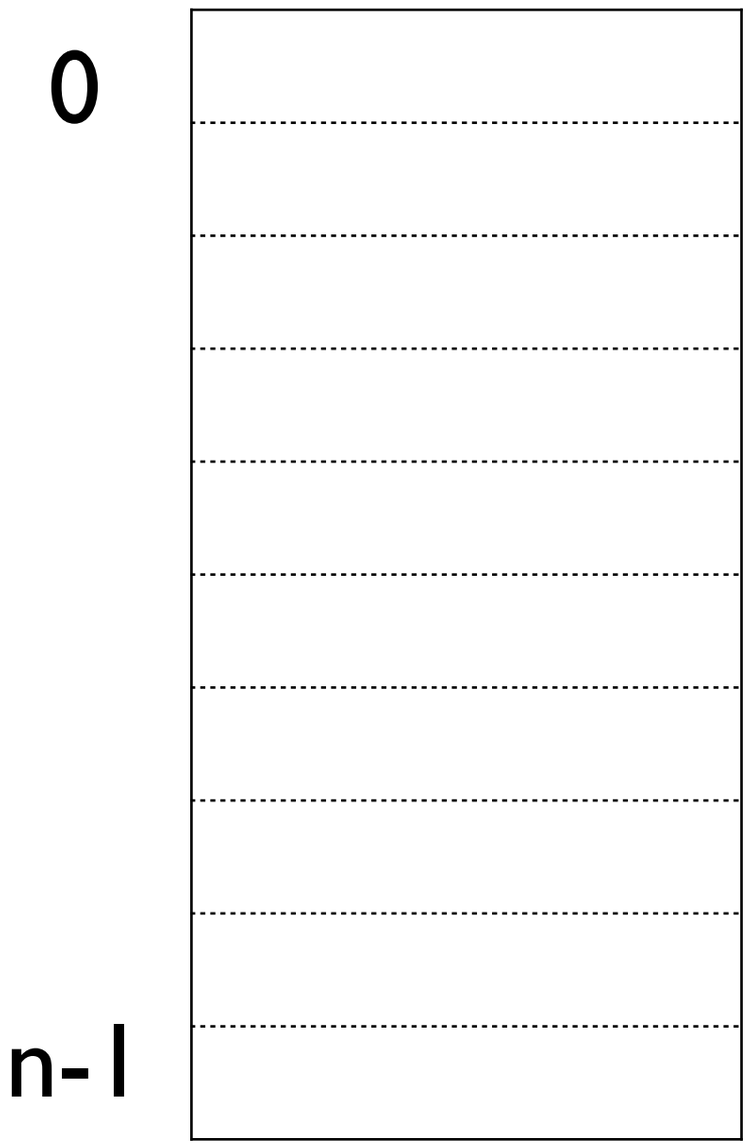

protective
shell
the soft centre

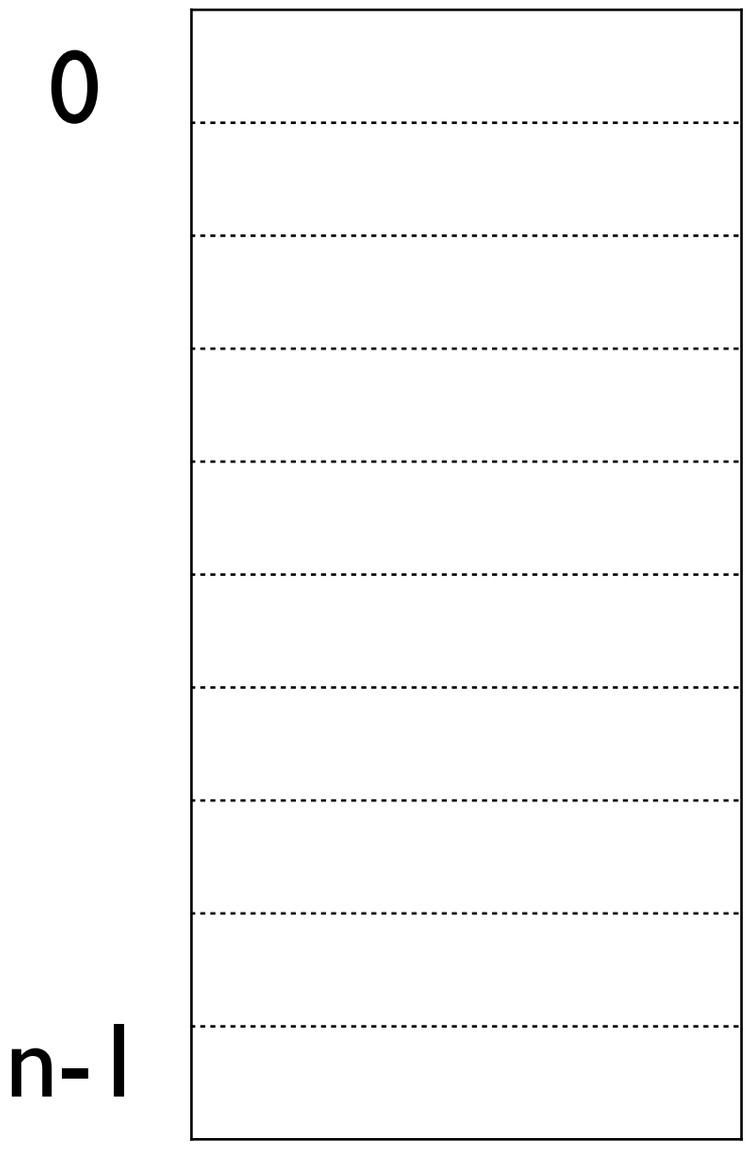

protective
shell
the soft centre
trusted zone
(it should just work)

```ruby
STRING_COUNT = 6

def tab_column string, fret
  ["---"             ] * (string - 1) +
  [fret.ljust(3,'-')] +
  ["---"             ] * (STRING_COUNT - string)
end


puts ARGF.each_line
        .either
        .map(&:split)
        .check("two fields per line") {|fs| fs.count == 2 }
        .check("two ints per line")   {|fs| fs.all? {|f| int?(f) }}
        .check("string # in [1..6]")  {|fs| in_range(1,6,fs[0].to_i) }
        .check("fret # in [0..24]")   {|fs| in_range(1,24,fs[1].to_i) }
        .map {|string,fret| tab_column(string.to_i, fret) }
        .transpose
        .map(&:join)
        .join($/)
```

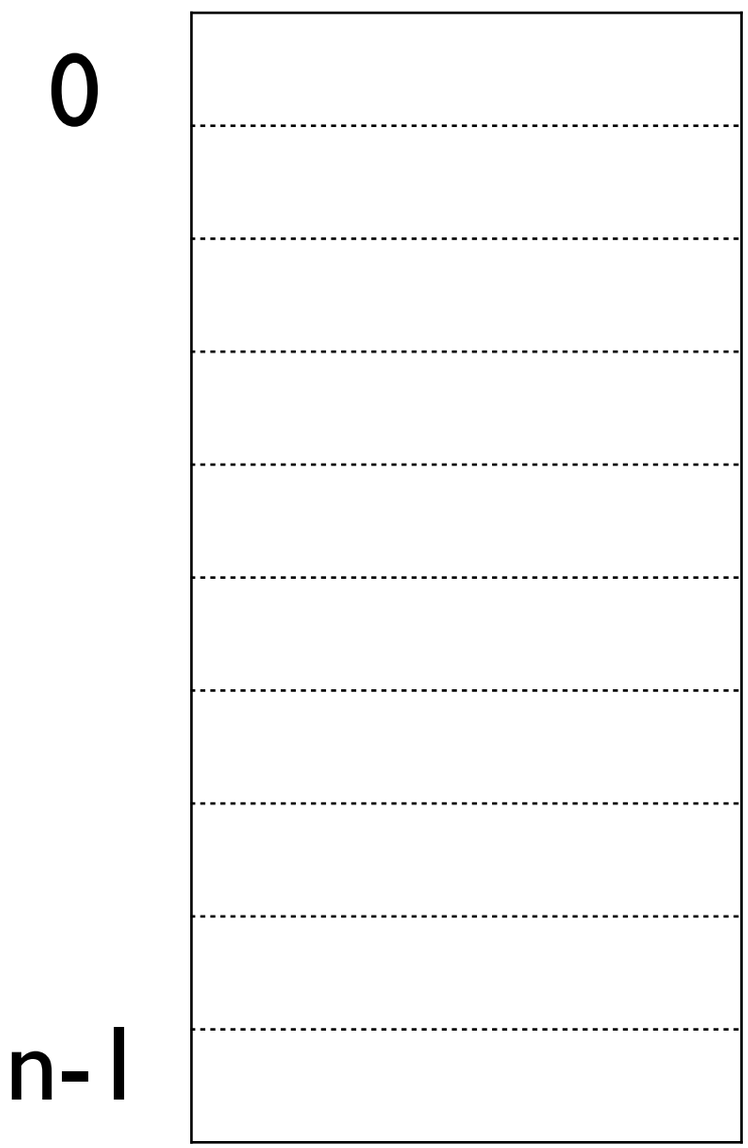
Intentions as a higher model


VIRGIN
3:25 PM
100%
My Music
FROOT
Marina and The Diamonds
Alternative • 2015
1 Happy
2 Froot
3 I'm a Ruin
4 Blue
5 Forget
6 Gold
7 Can't Pin Me Down
8 Solitaire
9 Better Than That
My Music
Playlists
VIRGIN
3:24 PM
100%
My Music
Recently Added
more
Artists
Albums
Songs
Music Videos
Genres
Composers
Compilations
Cancel

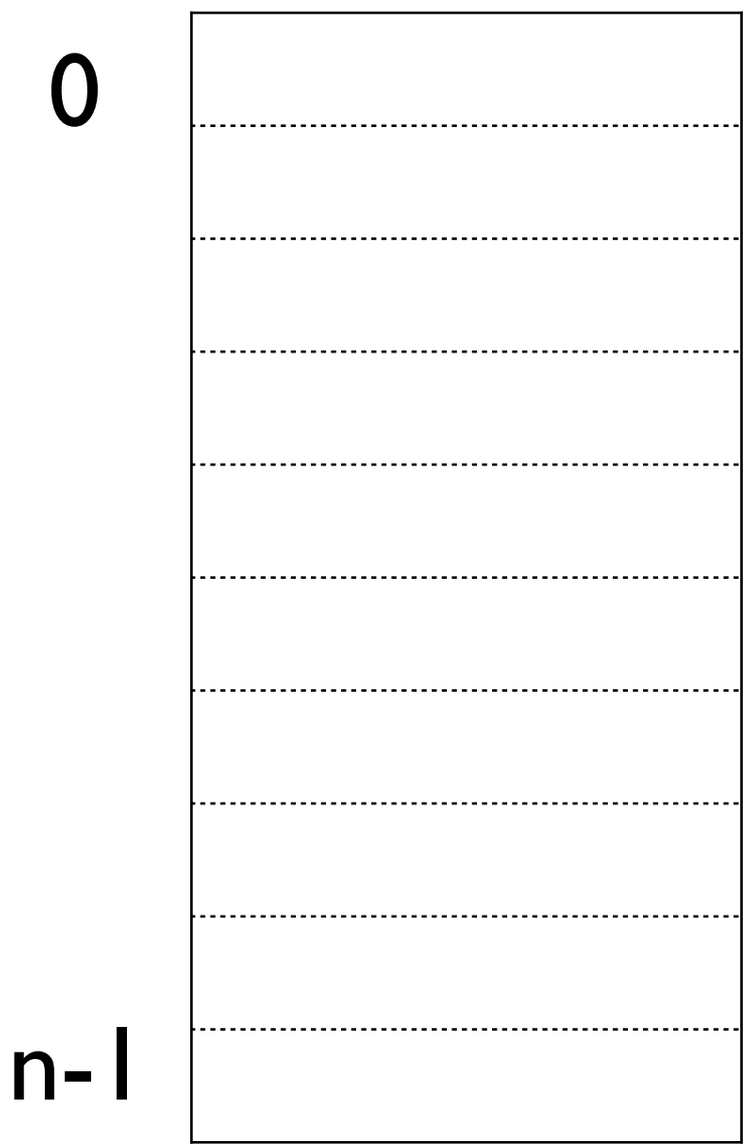

0
n-1



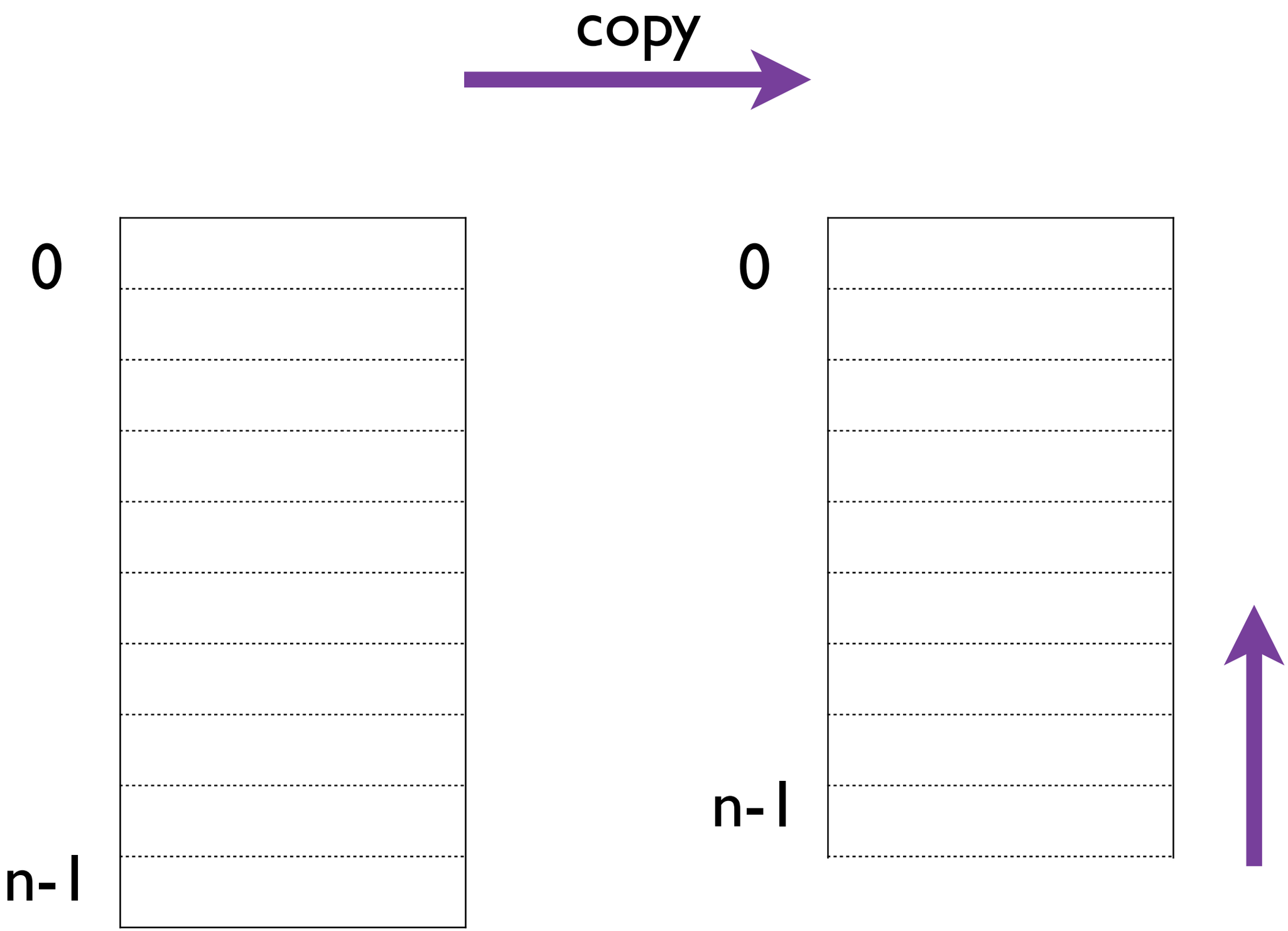
current_rand_n = rand(N)



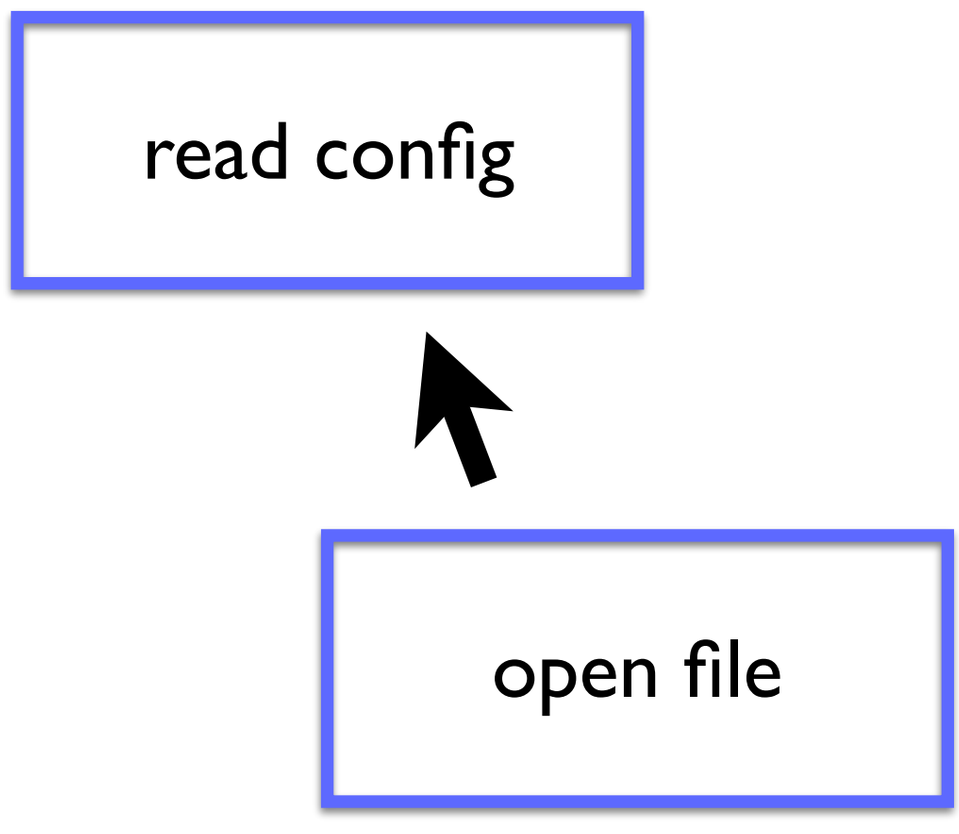

```
do
  current_rand_n = rand(N)
while current_rand_n == last_rand_n
last_rand_n = current_rand_n
```

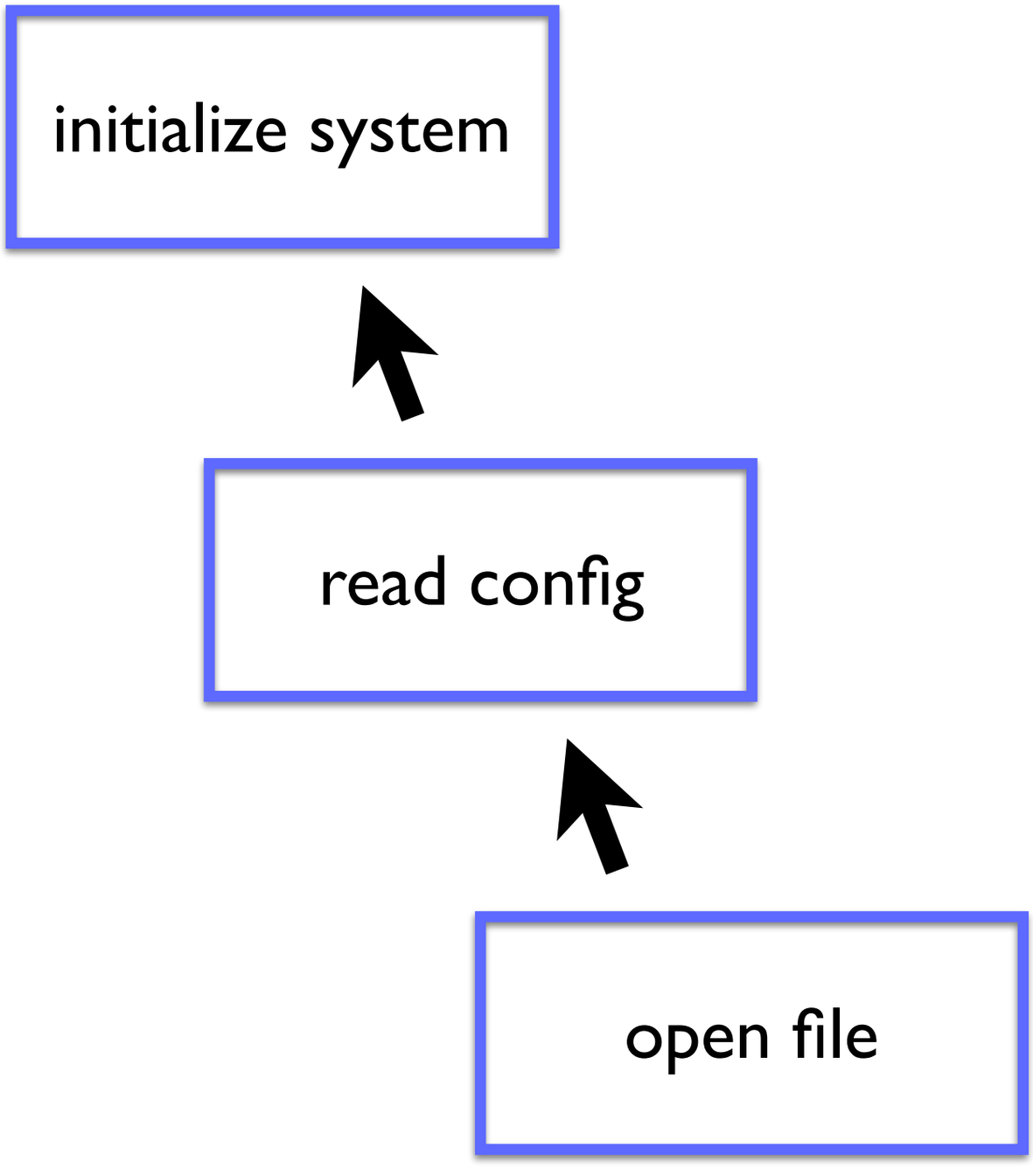



```
current_rand_n = rand(N)
if current_rand_n == last_rand_n
  current_rand_n = (last_rand_n + 1) % N
last_rand_n = current_rand_n
```


copy
0
n- l
0
n- l

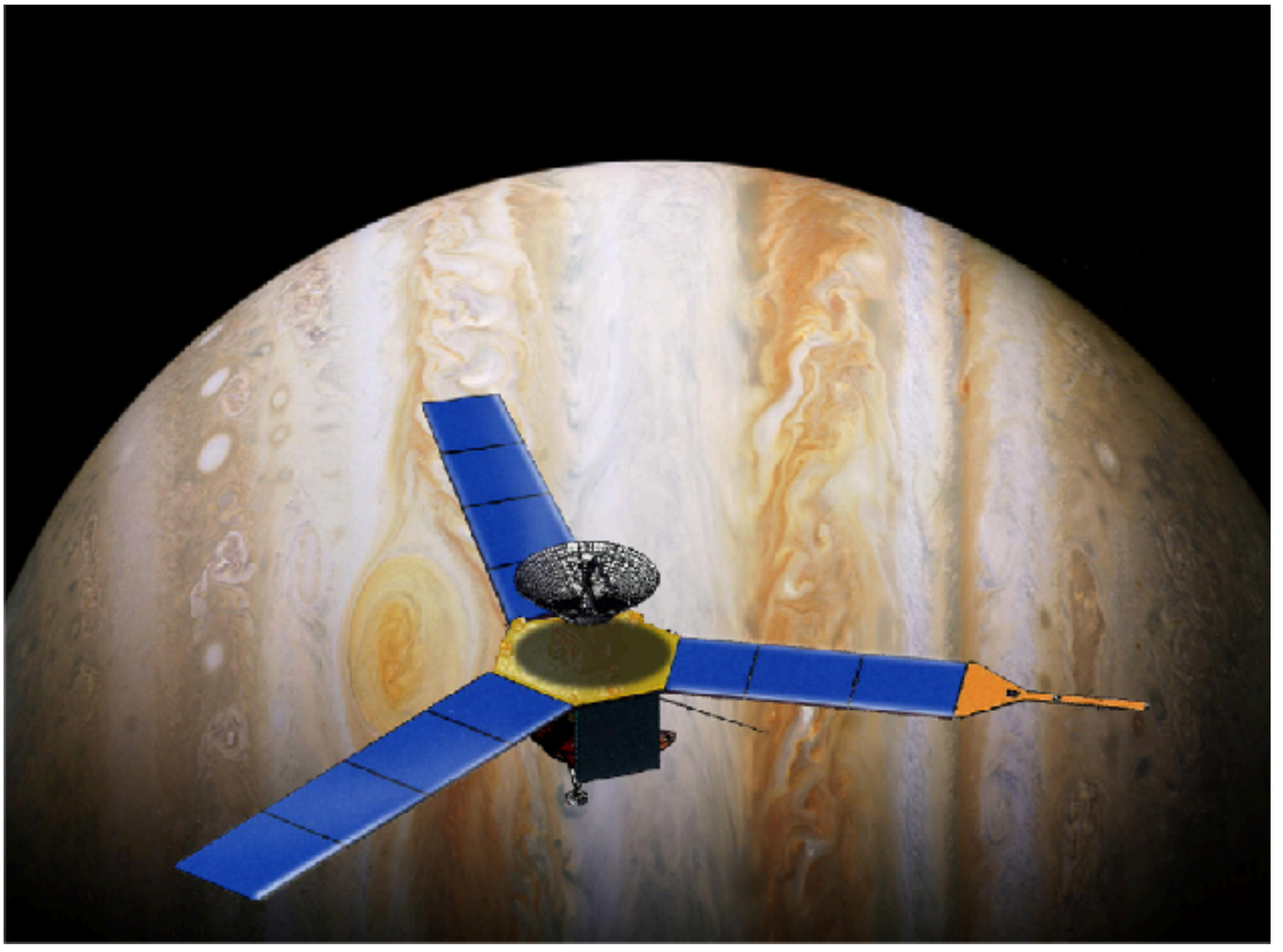
open file


read config
open file


initialize system
read config
open file

Edges


SOURCE
1
2
3
4
5
6
7
8
9
0
PRE-CH
MUTE
VOL
CH LIST
CH
MENU
SMART HUB
GUIDE
TOOLS
INFO
RETURN
EXIT
A
B
C
D
CC
SAMSUNG




Please don't
treat it
like one.

For robustness you need a human in the loop

Why Do Computers Stop and What Can Be Done About It?

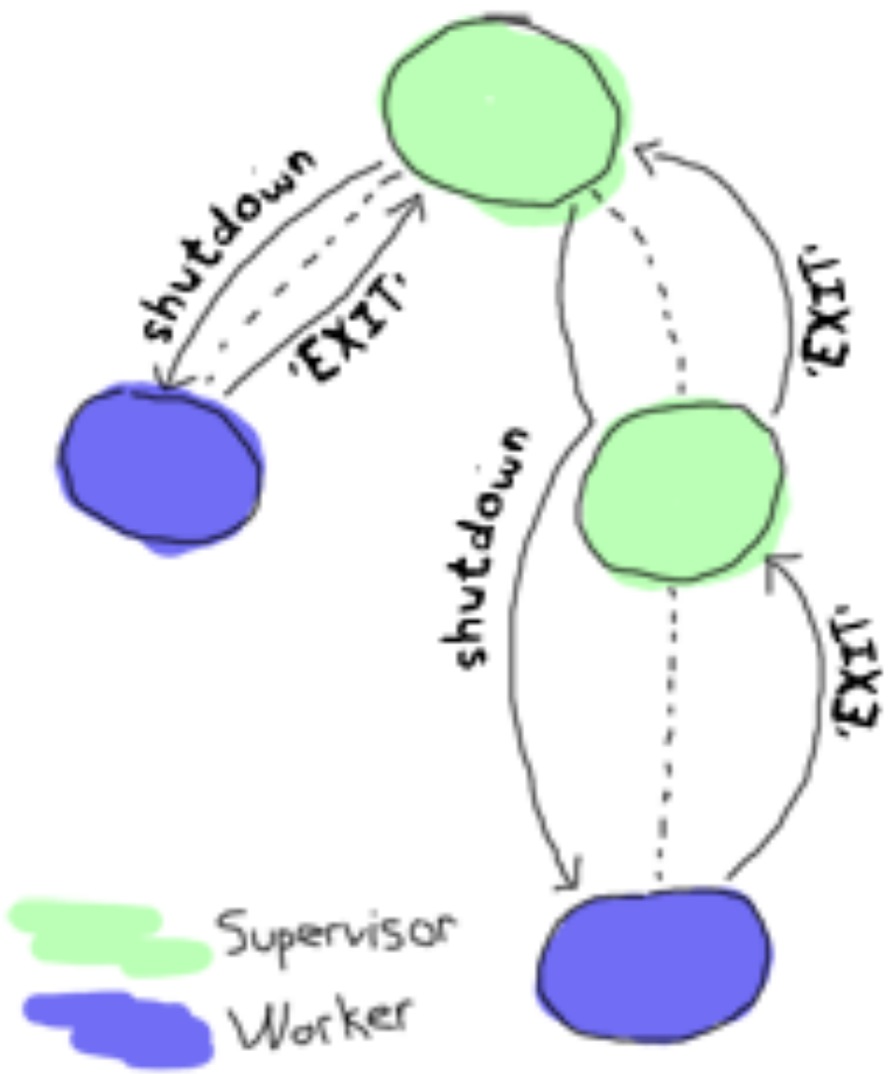
Jim Gray

Why Do Computers Stop and What Can Be Done About It?

> I conjecture that there is a similar phenomenon in software -- most production software faults are soft. If the program state is reinitialized and the failed operation retried, the operation will usually not fail the second time.

Jim Gray


ERLANG
KEEP
CALM
AND
LET IT
CRASH


shutdown
'EXIT'
shutdown
'EXIT'
'EXIT'
Supervisor
Worker

Safety is hard work.
Where do we want it?

Error-proofing is important when there is no supervision

What is the cost of safety?

No traffic signals, no curbs, no rules. Surprisingly, this doesn't lead to road rage and chaos, but to a dramatic drop in road accidents.

Errors are just conditions we refuse to take seriously

When our code works under more conditions
it can run unconditionally