Name: ________________________

Class: ______________________________________________________

BOOTSTRAP: 2

www.bootstrapworld.org
## Lesson 1

<table>
<thead>
<tr>
<th>Racket Code</th>
<th>Pyret Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>(define AGE 14)</td>
<td>AGE = 14</td>
</tr>
<tr>
<td>(define A-NUMBER 0.6)</td>
<td>A-NUMBER = 0.6</td>
</tr>
<tr>
<td>(define SPEED -90)</td>
<td>SPEED = -90</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Numbers</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>(define CLASS &quot;Bootstrap&quot;)</td>
<td>CLASS = “Bootstrap”</td>
</tr>
<tr>
<td>(define PHRASE &quot;Coding is fun!&quot;)</td>
<td>PHRASE = “Coding is fun!”</td>
</tr>
<tr>
<td>(define A-STRING &quot;2500&quot;)</td>
<td>A-STRING = “2500”</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Strings</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Two of your own:</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
(define SHAPE
  (triangle 40 "outline" "red"))

(define OUTLINE
  (star 80 "solid" "green"))

(define SQUARE
  (rectangle 50 50 "solid" "blue"))

SHAPE = triangle(40, "outline", "red")
OUTLINE = star(80, "solid", "green")
SQUARE = rectangle(50, 50, "solid", "blue")

One of your own:

(define BOOL true)
(define BOOL2 false)

BOOL = true

One of your own:

; double : Number -> Number
; Given a number, multiply by
; 2 to double it
(EXAMPLE (double 5) (* 2 5))
(EXAMPLE (double 7) (* 2 7))
(define (double n) (* 2 n))

# double : Number -> Number
# Given a number, multiply by
# 2 to double it
examples:
  double(5) is 2 * 5
  double(7) is 2 * 7
end

fun double(n):
  2 * n
end
Fast Functions!
Fill out the contract for each function, then try to write two examples and the definition by yourself.

<table>
<thead>
<tr>
<th>name</th>
<th>domain</th>
<th>range</th>
</tr>
</thead>
<tbody>
<tr>
<td>double</td>
<td>Number</td>
<td>Number</td>
</tr>
</tbody>
</table>

examples:
- double (5) is 2 * 5
- double (7) is 2 * 7

end

fun double (n):
  2 * n

end

<table>
<thead>
<tr>
<th>name</th>
<th>domain</th>
<th>range</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

examples:
- (____) is ____________________
- (____) is ____________________

end

fun _________(_______________):


end
Fast Functions!

Fill out the contract for each function, then try to write two examples and the definition by yourself.

<table>
<thead>
<tr>
<th># <em><strong><strong><strong><strong><strong><strong><strong><strong>:</strong></strong></strong></strong></strong></strong></strong></strong></em></th>
<th>domain</th>
<th>range</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

examples:

________(____) is ________________

________(____) is ________________

end

fun __________(__________):

______________________________________________________

end

<table>
<thead>
<tr>
<th># <em><strong><strong><strong><strong><strong><strong><strong><strong>:</strong></strong></strong></strong></strong></strong></strong></strong></em></th>
<th>domain</th>
<th>range</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

examples:

________(____) is ________________

________(____) is ________________

end

fun __________(__________):

______________________________________________________

end
Fast Functions!
Fill out the contract for each function, then try to write two examples and the definition by yourself.

<table>
<thead>
<tr>
<th># name</th>
<th>domain</th>
<th>range</th>
</tr>
</thead>
</table>

examples:

_______(______) is __________________

_______(______) is __________________

end

fun ________(__________):

_______________________________________________________

end
#1
SECONDS = (7)
STRING = my string

#2
SHAPE1 = circle(50 “solid” “blue”)
SHAPE2 = triangle(75, outline, yellow)

#3
# triple : Number -> Number
# Multiply a given number by # 3 to triple it
equations:
   triple(5) = 3 * 5
   triple(7) = 3 * 7
end

#4
fun triple(n):
   3 * n

#5
# ys : Number -> Number
# Given a number, create a solid # yellow star of the given size
equations:
   ys(99) is star(99, “solid”, “yellow”)
   ys(33) is star(99, “solid”, “yellow”)
ys(size):
   star(size “solid” “yellow”)
end
Write a function `double-radius`, which takes in a radius and a color. It produces an outlined circle of whatever color was passed in, whose radius is twice as big as the input.

**Contract+Purpose Statement**

Every contract has three parts:

```
# ___________ : ___________________ --> ________
        name                     Domain             Range

# ________________________________________________________________

What does the function do?
```

**Give Examples**

Write examples of your function in action

```
examples:

________(______________) is
    the user types...

_______________________________________________________________

...which should become

________(______________) is
    the user types...

_______________________________________________________________

...which should become

end
```

**Function**

Circle the changes in the examples, and name the variables. Write the code, copying everything that isn't circled, and using names where you find variables!

```
fun ____________ (______________):

______________________________________________________________

end
```
Word Problem: double-width

Write a function `double-width`, which takes in a number (the length of a rectangle) and produces a rectangle whose width is twice the given length.

**Contract+Purpose Statement**
Every contract has three parts:

```
# n : l -> w
```

name | Domain     | Range
--- | ----------- | ---

What does the function do?

**Give Examples**
Write examples of your function in action

```
examples:
_________________________ is the user types...
...
...which should become

_________________________ is the user types...
...
...which should become
```

end

**Function**
Circle the changes in the examples, and name the variables.
Write the code, copying everything that isn't circled, and using names where you find variables!

```
fun _______________(______________) :

______________________________________________

end
```
Word Problem: next-position

Write a function `next-position`, which takes in two numbers (an x and y-coordinate) and returns a Coord, increasing the x-coordinate by 5 and decreasing the y-coordinate by 5.

Contract+Purpose Statement
Every contract has three parts:

```
# __________ : ____________________________ --> _________
  name                  Domain            Range
#
```

What does the function do?

Give Examples
Write examples of your function in action

examples:

```
    (___________________)        is
        the user types...
```

```
    ...which should become
```

```
    (___________________)        is
        the user types...
```

```
    ...which should become
```

d

Function
Circle the changes in the examples, and name the variables.
Write the code, copying everything that isn't circled, and using names where you find variables!

```
fun ________________(_______________):

```

end
# a Cake is a flavor, color, message, layers, & is-iceCream

data Cake:
    | cake(__________________________________________________
    | ___________________________________________________
    | ___________________________________________________
    | ___________________________________________________
    | ___________________________________________________
end

To make examples of this structure, I would write:

cake1 = ____________________________________________

cake2 = ____________________________________________

To access the fields of cake2, I would write:

_______________________________________
_______________________________________
_______________________________________
_______________________________________
_______________________________________
# a Party is a location, theme, and number of guests
data Party:
    | party(___________________________________________
    | _____________________________________________
    | _____________________________________________
end

To make examples of this structure, I would write:

```
party1 = _____________________________________________

party2 = _____________________________________________
```

To access the fields of party2, I would write:

```
_______________________________________
_______________________________________
_______________________________________
```
Word Problem: change-flavor

Write a function called change-flavor, which takes in a Cake and a flavor, and returns a new Cake that is almost the same as the original, but is now the given flavor.

Contract+Purpose Statement

# ____________ : ___________________________ -> __________
# ____________________________________________

Give Examples

examples:

_________(___________) is

______________________________________

______________________________________

______________________________________

______________________________________

_____________(___________) is

______________________________________

______________________________________

______________________________________

______________________________________

end

Function

fun ______________(______________) :

______________________________________

______________________________________

______________________________________

______________________________________

end
Word Problem: will-melt
Write a function called will-melt, which takes in a Cake and a temperature, and returns true if the temperature is greater than 32 degrees, AND the Cake is an ice cream cake.

Contract+Purpose Statement

# ______________ : ___________________________ -> __________
# ____________________________________________

Give Examples

examples:

________________________(____________________) is

________________________________________

________________________(____________________) is

________________________________________

end

Function

fun ______________(____________________):

________________________________________

end
Word Problem: keypress (Ninja World)

State the Problem
For each keypress in Ninja World, show how (keypress <world> <key>) should change the world.

Contract+Purpose Statement

# keypress : World String -> World

# Given a world and a key, produce a new world with NinjaCat’s position moved by 10 pixels, depending on which arrow key was pressed

Give Examples

examples:

keypress(worldA, “up”) is
world(worldA.dogX, worldA.coinX, worldA.catX, worldA.catY + 10)

keypress(worldB, “down”) is
world(worldB.dogX, worldB.coinX, worldB.catX, worldB.catY - 10)

keypress(worldA, “left”) is
world(worldA.dogX, worldA.coinX, worldA.catX - 10, worldA.catY)

keypress(worldB, “right”) is
world(worldB.dogX, worldB.coinX, worldB.catX + 10, worldB.catY)

end
fun keypress(current-world, key) :

    ask:
    | string-equal(key, “up”) then:
    | string-equal(key, “down”) then:
    | string-equal(key, “left”) then:
    | string-equal(key, “right”) then:
    | otherwise: current-world

end
Word Problem: next-world (Ninja World)

Given a world, return the next world by adding 10 to dogX, subtracting 5 from coinX, and subtracting 5 from catY only when the cat’s y-coordinate is greater than 75.

Contract+Purpose Statement

# __________________ : ___________________________ -> ____________
# ____________________________

Give Examples

examples:

_________(__________ ) is

________________________________________________________________________________________
________________________________________________________________________________________
________________________________________________________________________________________
_______________________________________________________________________________________

_________(__________ ) is

________________________________________________________________________________________
________________________________________________________________________________________
________________________________________________________________________________________
_______________________________________________________________________________________

end
fun __________________________(____________________): 

  ask:
  | ________________________________ then:
  | ________________________________
  | ________________________________
  | ________________________________
  | ________________________________
  | ________________________________
  | ________________________________
  | ________________________________
  | ________________________________
  | otherwise:
  | ________________________________
  | ________________________________
  | ________________________________
  | ________________________________
  | ________________________________
  | ________________________________
  end

end
Word Problem: red-shape

Write a function red-shape, which takes in the name of a shape (such as “circle”, “triangle”, “star”, or “rectangle”), and draws that solid, red shape. Use 50 as the radius of the circle and star, and side-length of the triangle. Make the rectangle 99 pixels long by 9 wide.

# : ->

#

Give Examples

examples:

____________(____________) is _______________________

____________(____________) is _______________________

____________(____________) is _______________________

____________(____________) is _______________________

end

Function

fun __________(______________):

  ask:
  | ______________ then:
  | ______________ then:
  | ______________ then:
  | ______________ then:

end
end
Word Problem: strong-password

Websites have strict password requirements. Write a function strong-password, which takes in a username and password, and checks to make sure they aren’t the same, and then checks the string-length of the password to make sure it is greater than 8 characters. The function should return a message to the user letting them know if their password is strong enough.

# : ->

#

Give Examples

examples:

________________(________________________) is __________________________

________________(________________________) is __________________________

________________(________________________) is __________________________

________________(________________________) is __________________________

end

Function

fun __________(______________):
  ask:
    | ____________________________ then:
    | ____________________________
    | ____________________________ then:
    | ____________________________
    | otherwise: ______________________________
  end
end
# is-off-right : _______________ -> _______________

examples:

____________________(______) is

______________________________

____________________(______) is

______________________________

end

fun ______________(______________):

______________________________

end

# is-off-left : _______________ -> _______________

examples:

____________________(______) is

______________________________

____________________(______) is

______________________________

end

fun ______________(______________):

______________________________

end
examples:

___________(______) is ________________________________

___________(______) is ________________________________

end

fun __________(__________):

______________________________

end
Write a function called `line-length`, which takes in two numbers and returns the difference between them. It should always subtract the smaller number from the bigger one.

**Contract Purpose Statement**

```plaintext
# ___________: _______________________________ -> ___________
# ____________________________________________
```

**Give Examples**

```plaintext
examples:

_________(________)    is

______________________________________________

_________(________)    is

______________________________________________
```

end

**Function Header**

```plaintext
fun ___________(__________): 
    function name  variable names

    _____:

    __________________________________

    __________________________________

    __________________________________

    __________________________________

    __________________________________

    __________________________________

    end
```

end
Distance:

The Player is at \((4, 2)\) and the Target is at \((0, 5)\).
Distance takes in the player’s \(x\), player’s \(y\), character’s \(x\) and character’s \(y\).

Use the formula below to fill in the EXAMPLE:

\[
\sqrt{(line - length \ 4 \ 0)^2 + (line - length \ 2 \ 5)^2}
\]

Convert it into a Circle of Evaluation. (We’ve already gotten you started!)

Convert it into Pyret code:
Word Problem: distance

Write a function distance, which takes FOUR inputs:
- px: The x-coordinate of the player
- py: The y-coordinate of the player
- cx: The x-coordinate of another game character
- cy: The y-coordinate of another game character

It should return the distance between the two, using the Distance formula:

\[
\text{Distance}^2 = (\text{line-length} \ px \ cx)^2 + (\text{line-length} \ py \ cy)^2
\]

Contract+Purpose Statement

# ___________ : ___________________________ => __________
# ____________________________

Give Examples

Write examples of your function in action

examples:

__________ (__________) is ________________________________

__________ (__________) is ________________________________

end

Function

fun ___________ (___________):

________________________

________________________

end
Word Problem: is-collision

Write a function `is-collision`, which takes FOUR inputs:
- `px`: The x-coordinate of the player
- `py`: The y-coordinate of the player
- `cx`: The x-coordinate of another game character
- `cy`: The y-coordinate of another game character

It should return `true` if the coordinates of the player are within **50 pixels** of the coordinates of the other character. Otherwise, `false`.

Contract+Purpose Statement

```
# ______________ : ___________________________ -> ______

# _____________________________
```

Give Examples

Write examples of your function in action

```
examples:

________________________(________________)    is

________________________(________________)

________________________(________________)    is

________________________(________________)

end
```

Function

```
fun _____________(________________): 

___________________________

end
```
GAME DESIGN
“Start Simple, Get Complex”

Draw a rough sketch of your game when it begins, and another sketch just a moment later

A sketch at the START of the game…  A sketch for the very NEXT moment…

What images will you need for your game? Name them in the 1st column, and describe them in the 2nd column.

<table>
<thead>
<tr>
<th>BACKGROUND</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

List everything that has changed from one sketch to the other. What datatype will represent it?

<table>
<thead>
<tr>
<th>Changed (position, score, color, costume…)</th>
<th>Datatype (Number, String, Image, Boolean…)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# a world is a

data World:

| world(_____________________________
| _______________________________________________________________________
| _______________________________________________________________________
| _______________________________________________________________________
| _______________________________________________________________________
| _______________________________________________________________________)

end

To make example worlds that represent my sketches from page 31, I would write...

worldA = ___________________________________________________________________

worldB = ___________________________________________________________________

To access the fields of worldA, I would write:

__________________________________________________________________________
__________________________________________________________________________
__________________________________________________________________________
__________________________________________________________________________
__________________________________________________________________________
Word Problem: draw-world  (My game)

Contract

#____________:_____________________________ -> _______

Definition

fun __________(_______________________________):

   put-image(___________________________)

   ______________________________________
   ______________________________________
   ______________________________________
   ______________________________________
   ______________________________________
   ______________________________________
   ______________________________________
   ______________________________________
   ______________________________________

end
Word Problem: next-world (My game)

State the problem (What changes?):

Contract+Purpose Statement

# ___________: ________________________ -> ___________
# _______________________________________

Give Examples
examples:

_________ (______________) is

___________________________________________
___________________________________________
___________________________________________
___________________________________________
___________________________________________

_________ (______________) is

___________________________________________
___________________________________________
___________________________________________
___________________________________________
___________________________________________

end

Function

fun _______________(______________):

___________________________________________
___________________________________________
___________________________________________
___________________________________________
___________________________________________

end
Lesson 9

<table>
<thead>
<tr>
<th>When this key is pressed...</th>
<th>...this field of the new world...</th>
<th>...changes by...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Word Problem: keypress (My game)

For each keypress in your game, show how keypress(worldA, <key>) should change your world.

# : ->

#

Give Examples

examples:

keypress(worldA, ________) is

___________________________________________
___________________________________________
___________________________________________
___________________________________________

keypress(worldA, ________) is

___________________________________________
___________________________________________
___________________________________________

keypress(worldA, ________) is

___________________________________________
___________________________________________
___________________________________________
___________________________________________

end
fun funfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfunfun
ask:
| ________________ then:
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
| ______________________________
end
end
Building Your Helper Functions

# is-off-right : ________________ -> ________________

examples:

________________________(______) is

______________________________

________________________(______) is

______________________________

end

fun __________(______________):

________________________________________________________________________

end

# is-off-left : ________________ -> ________________

examples:

________________________(______) is

______________________________

________________________(______) is

______________________________

end

fun __________(______________):

________________________________________________________________________
end

# _________________:_________________-> ____________

examples:

________________(____) is

___________________________

________________(____) is

___________________________

end

fun __________(_____________):

___________________________

end

# _________________:_________________-> ____________

examples:

________________(____) is

___________________________

________________(____) is

___________________________

end

fun __________(_____________):

___________________________

end
Using Helpers inside `next-world`:

How does the World structure change when....?

<table>
<thead>
<tr>
<th>TEST</th>
<th>RESULT</th>
</tr>
</thead>
</table>
|      | `world(___________________________________)
|      |        |
|      | `world(___________________________________)
|      |        |
|      | `world(___________________________________)
|      |        |
|      | `world(___________________________________)
|      |        |
|      | `world(___________________________________)
|      |        |
|      | `world(___________________________________)
<p>| | |
|      |        |</p>
<table>
<thead>
<tr>
<th>TEST</th>
<th>RESULT</th>
</tr>
</thead>
</table>
|      | world(___________________________________
|      | ________________________________________
|      | ________________________________________
|      | ________________________________________
|      | ________________________________________
|      | ________________________________________ |
Using Helpers inside `draw-world`:

What changes the appearance of your game?

<table>
<thead>
<tr>
<th>TEST</th>
<th>RESULT</th>
</tr>
</thead>
</table>
| put-image(____________________)  
put-image(____________________)  
put-image(____________________)  
put-image(____________________)  
put-image(____________________) |        |
| put-image(____________________)  
put-image(____________________)  
put-image(____________________)  
put-image(____________________)  
put-image(____________________) |        |
| put-image(____________________)  
put-image(____________________)  
put-image(____________________)  
put-image(____________________)  
put-image(____________________) |        |
| put-image(____________________)  
put-image(____________________)  
put-image(____________________)  
put-image(____________________)  
put-image(____________________) |        |
<table>
<thead>
<tr>
<th>TEST</th>
<th>RESULT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
<tr>
<td></td>
<td>put-image(____________________________)</td>
</tr>
</tbody>
</table>
Contract+Purpose Statement
Every contract has three parts:

# __________________________: ___________________________________ -> __________
	name                      Domain                      Range

# __________________________

What does the function do?

Give Examples
Write examples of your function in action

examples:

________________________(__________) is
	the user types...

________________________________________

...which should become

________________________(__________) is
	the user types...

________________________________________

...which should become

end

Function
Circle the changes in the examples, and name the variables.

fun __________________________(___________________):


end
**Design Recipe**

**Contract+Purpose Statement**
Every contract has three parts:

# ___________________ : ___________________ -> ___________________
                   name                                      Domain         Range

# ___________________

What does the function do?

**Give Examples**
Write examples of your function in action

examples:

____________(__________)  is
the user types...

____________________________________________________

...which should become

____________(__________)  is
the user types...

____________________________________________________

...which should become

end

**Function**
Circle the changes in the examples, and name the variables.

fun _________________(_______________):

________________________________________________________________________

end
<table>
<thead>
<tr>
<th>Name</th>
<th>Domain</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>..</td>
<td>......</td>
</tr>
</tbody>
</table>

*Example contracts table with placeholders for specific values.*
<table>
<thead>
<tr>
<th>Contracts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
</tr>
<tr>
<td>Name</td>
</tr>
<tr>
<td>Range</td>
</tr>
<tr>
<td>example</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
<tr>
<td>..</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
<tr>
<td>#</td>
</tr>
</tbody>
</table>