

# Unit 1

|                | Racket Code   | Pyret Code   |
|----------------|---|--|
| <b>Numbers</b> | <pre>(define AGE 14) (define A-NUMBER 0.6) (define SPEED -90)</pre>                             | AGE = 14<br>A-NUMBER = 0.6<br>SPEED = -90<br>Two of your own:<br><hr/> <hr/>                             |
| <b>Strings</b> | <pre>(define CLASS "Bootstrap") (define PHRASE "Coding is fun!") (define A-STRING "2500")</pre> | CLASS = "Bootstrap"<br>PHRASE = "Coding is fun!"<br>A-STRING = "2500"<br>Two of your own:<br><hr/> <hr/> |

|                  |  |  |
|------------------|--|--|
| <i>Images</i>    | <pre>(define SHAPE   (triangle 40 "outline" "red"))  (define OUTLINE   (star 80 "solid" "green"))  (define SQUARE   (rectangle 50 50 "solid" "blue"))</pre>                      | <pre>SHAPE = triangle(40, "outline", "red")  OUTLINE = star(80, "solid", "green")  SQUARE = rectangle(50, 50, "solid", "blue")</pre> <p>One of your own:</p> <hr/>             |
| <i>Booleans</i>  | <pre>(define BOOL true)  (define BOOL2 false)</pre>  | <pre>BOOL = true</pre> <p>One of your own:</p> <hr/>   |
| <i>Functions</i> | <pre>; double : Number -&gt; 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))</pre> | <pre># double :: Number -&gt; 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</pre> |

# Fast Functions!

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

# double :: Number → Number  
name domain range

examples:

double ( 5 ) is 2 \* 5  
double ( 7 ) is 2 \* 7

end

fun double ( n ) :

2 \* n

end

#                    ::                    ->                     
name domain range

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.

# \_\_\_\_\_ :: \_\_\_\_\_ → \_\_\_\_\_  
name domain range

examples:

\_\_\_\_\_ (\_\_\_\_\_ ) is \_\_\_\_\_

\_\_\_\_\_ (\_\_\_\_\_ ) is \_\_\_\_\_

end

fun \_\_\_\_\_ (\_\_\_\_\_ ) :

\_\_\_\_\_

end

# \_\_\_\_\_ :: \_\_\_\_\_ -> \_\_\_\_\_  
name domain range

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.

# \_\_\_\_\_ :: \_\_\_\_\_ → \_\_\_\_\_  
name domain range

**examples:**

\_\_\_\_\_ (\_\_\_\_\_ ) is \_\_\_\_\_

\_\_\_\_\_ (\_\_\_\_\_ ) is \_\_\_\_\_

end

fun \_\_\_\_\_ (\_\_\_\_\_ ) :

\_\_\_\_\_

end

# \_\_\_\_\_ :: \_\_\_\_\_ -> \_\_\_\_\_  
name domain range

**examples:**

\_\_\_\_\_ (\_\_\_\_\_ ) is \_\_\_\_\_

\_\_\_\_\_ (\_\_\_\_\_ ) is \_\_\_\_\_

end

fun \_\_\_\_\_ (\_\_\_\_\_ ) :

\_\_\_\_\_

end

# Syntax and Style Bug Hunting: Pyret Edition

|    |   |  |
|----|---|--|
| #1 | <pre>SECONDS = (7)  STRING = my string</pre>  |  |
| #2 | <pre>SHAPE1 = circle(50 "solid" "blue")  SHAPE2 = triangle(75, outline, yellow)</pre>   |  |
| #3 | <pre># triple :: Number -&gt; Number # Multiply a given number by # 3 to triple it  examples:   triple(5) = 3 * 5   triple(7) = 3 * 7 end</pre>   |  |
| #4 | <pre>fun triple(n):   3 * n</pre>   |  |
| #5 | <pre># ys :: Number -&gt; Number # Given a number, create a solid # yellow star of the given size  examples:   ys(99) is star(99, "solid", "yellow")   ys(33) is star(99, "solid", "yellow")  ys(size):   star(size "solid" "yellow") end</pre> |  |

## Unit 2

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

## Word Problem: double-radius

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 :**

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

is \_\_\_\_\_  
...which should become

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

is \_\_\_\_\_  
end ...which should become

### 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:

# \_\_\_\_\_ :: \_\_\_\_\_ → \_\_\_\_\_  
name Domain Range

# \_\_\_\_\_

What does the function do?

## Give Examples

Write examples of your function in action

**examples :**

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

**is** \_\_\_\_\_

...which should become

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

**is** \_\_\_\_\_

...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 DeliveryState, 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:**

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

is \_\_\_\_\_  
...which should become

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

is \_\_\_\_\_  
end ...which should become

### 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

# Data Structure

# A CakeType is a **flavor**, **layers**, & **is-iceCream**

data **CakeType**:

| **cake**( \_\_\_\_\_  
| \_\_\_\_\_  
| \_\_\_\_\_ )

end

To make instances of this structure, I would write:

**cake1** = \_\_\_\_\_

**cake2** = \_\_\_\_\_

To access the fields of **cake2**, I would write:

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

## Word Problem: taller-than

Write a function called *taller-than*, which consumes two CakeTypes, and produces true if the number of layers in the first CakeType is greater than the number of layers in the second.

Contract+Purpose Statement

# \_\_\_\_\_ :: \_\_\_\_\_ → \_\_\_\_\_

# \_\_\_\_\_

### Give Examples

Write examples of your function in action

**examples :**

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

**is** \_\_\_\_\_

...which should become

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

**is** \_\_\_\_\_

**end** ...which should become

### 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: will-melt

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

Contract+Purpose Statement

# \_\_\_\_\_ :: \_\_\_\_\_ → \_\_\_\_\_

# \_\_\_\_\_

### Give Examples

Write examples of your function in action

**examples:**

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

is \_\_\_\_\_

...which should become

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

is \_\_\_\_\_

...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**

## Vocabulary Practice

Below is a new structure definition:

**data** MediaType:

```
| book(  
|   title :: String,  
|   author :: String,  
|   pubyear :: Number)
```

**end**

# an example book:

```
book1 = book("1984", "Orwell", 1949)
```

Fill in the blanks below with the vocabulary term that applies to each name. Here are the terms to choose from:

- |               |              |
|---------------|--------------|
| - contract    | - example    |
| - header      | - field      |
| - datatype    | - instance   |
| - constructor | - data block |
| - name        | - purpose    |

**author** is a \_\_\_\_\_

**book** is a \_\_\_\_\_

**MediaType** is a \_\_\_\_\_

**book1** is a \_\_\_\_\_

**title** is a \_\_\_\_\_

**data ... end** is a \_\_\_\_\_

## Unit 3

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

# Identifying Animation Data Worksheet: Sunset

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

(worksheet continues on the next page)

Define the Data Structure

# a \_\_\_\_\_ **State** is \_\_\_\_\_

data \_\_\_\_\_ **State**:

| \_\_\_\_\_ ( \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ )

end

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

## Word Problem: draw-state

Write a function called `draw-state`, which takes in a `SunsetState` and returns an image in which the sun (a circle) appears at the position given in the `SunsetState`. The sun should be behind the horizon (the ground) once it is low in the sky.

Contract+Purpose Statement

# `draw-state` :: \_\_\_\_\_ → Image

# \_\_\_\_\_

Write an expression for each piece of your final image

|          |  |
|----------|--|
| SUN =    |  |
| GROUND = |  |
| SKY =    |  |

Write the `draw-state` function, using `put-image` to combine your pieces

fun \_\_\_\_\_( \_\_\_\_\_ ) :

---

---

---

end

## Word Problem: next-state-tick

Write a function called *next-state-tick*, which takes in a *SunsetState* and returns a *SunsetState* in which the new x-coordinate is 8 pixels larger than in the given *SunsetState* and the y-coordinate is 4 pixels smaller than in the given *SunsetState*.

Contract+Purpose Statement

# \_\_\_\_\_ :: \_\_\_\_\_ → \_\_\_\_\_

# \_\_\_\_\_

Give Examples

Write examples of your function in action

**examples:**

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

**is** \_\_\_\_\_  
...which should become

\_\_\_\_\_ ( \_\_\_\_\_ )

the user types...

**is** \_\_\_\_\_  
**end** ...which should become

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**

# Identifying Animation Data Worksheet

Draw a sketch for three distinct moments of the animation

|  |  |  |
|--|--|--|
|  |  |  |
|--|--|--|

Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

(worksheet continues on the next page)

Define the Data Structure

```
# a _____ State is _____  
  
data _____ State:  
| _____ ( _____  
| _____  
| _____  
| _____ )  
end
```

Make a sample instance for each sketch from the previous page:

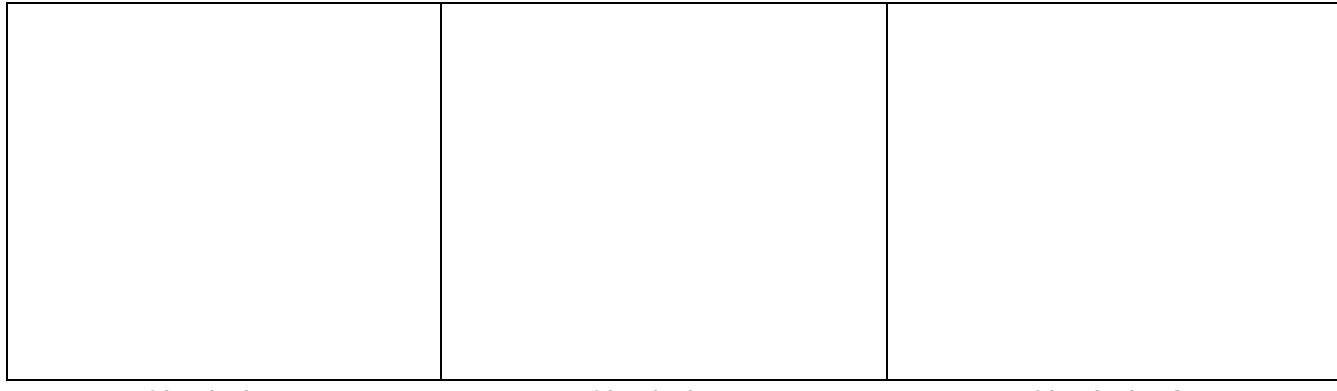
\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

# Identifying Animation Data Worksheet

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

(worksheet continues on the next page)

Define the Data Structure

```
# a _____ State is _____  
  
data _____ State:  
| _____ ( _____  
| _____  
| _____  
| _____ )  
end
```

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

# Identifying Animation Data Worksheet

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

(worksheet continues on the next page)

Define the Data Structure

```
# a _____ State is _____  
  
data _____ State:  
| _____ ( _____  
| _____  
| _____  
| _____ )  
end
```

Make a sample instance for each sketch from the previous page:

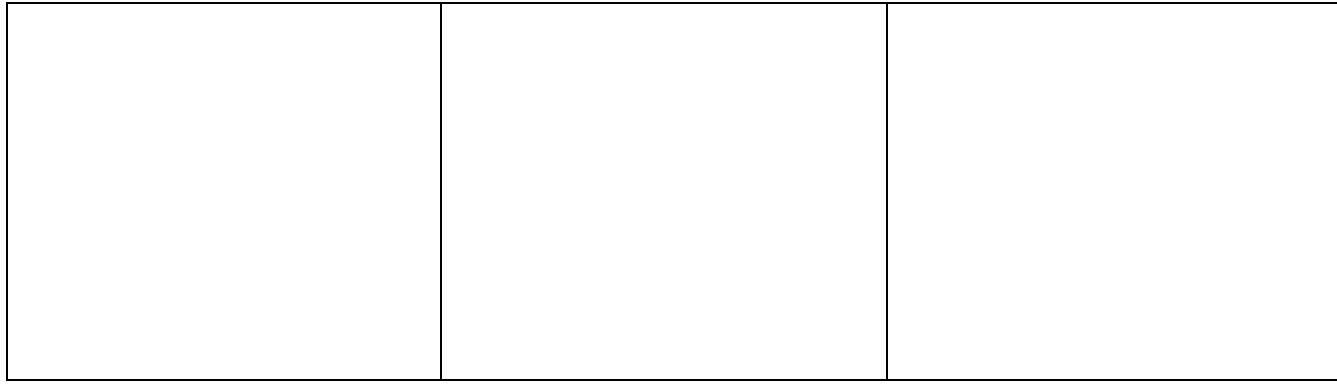
\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

# Identifying Animation Data Worksheet

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

(worksheet continues on the next page)

Define the Data Structure

```
# a _____ State is _____  
  
data _____ State:  
| _____ ( _____  
| _____  
| _____  
| _____ )  
end
```

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

## Unit 4

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

## Word Problem: location

Write a function called *location*, which consumes a *DeliveryState*, and produces a String representing the location of a box: either “road”, “delivery zone”, “house”, or “air”.

Contract+Purpose Statement

# \_\_\_\_\_ :: \_\_\_\_\_ → \_\_\_\_\_

# \_\_\_\_\_

Give Examples

examples :

\_\_\_\_\_ ( \_\_\_\_\_ ) is \_\_\_\_\_

end

(worksheet continues next page)

# Syntax and Style Bug Hunting: Piecewise Edition

|         | Buggy Code   | Correct Code / Explanation |
|---------|--|----------------------------|
| Round 1 | <pre>fun piecewisefun(n):     if (n &gt; 0): n     else: 0</pre>   |                            |
| Round 2 | <pre>fun cost(topping):     if string-equal(topping, "pepperoni"): 10.50     else string-equal(topping, "cheese"): 9.00     else string-equal(topping, "chicken"): 11.25     else string-equal(topping, "broccoli"): 10.25     else: "That's not on the menu!"     end end</pre> |                            |
| Round 3 | <pre>fun absolute-value(a b):     if a &gt; b: a - b     b - a     end end</pre>   |                            |
| Round 4 | <pre>fun best-function(f):     if string-equal(f, "blue"):         "you win!"     else if string-equal(f, "blue"):         "you lose!"     else if string-equal(f, "red"):         "Try again!"     else: "Invalid entry!"     end end</pre>                                     |                            |

# Animation Extension Worksheet

Describe the goal of your change: what new feature or behavior will it add to your animation?

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |

Make a To-Do List, and check off each as "Done" when you finish each one.

| Component       | When is there work to be done?  | To-Do                    | Done                     |
|-----------------|---|--------------------------|--------------------------|
| Data Structure  | If any new field(s) were added, changed or removed                    | <input type="checkbox"/> | <input type="checkbox"/> |
| draw-state      | If something is displayed in a new way or position                    | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-tick | If the Data Structure changed, or the animation happens automatically | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-key  | If the Data Structure changed, or a keypress triggers the animation   | <input type="checkbox"/> | <input type="checkbox"/> |
| reactor         | If either next-state function is new                                  | <input type="checkbox"/> | <input type="checkbox"/> |

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write at least one NEW example for one of the functions on your To-Do list

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

If you have another function on your To-Do list , write at least one NEW example

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

## Word Problem: draw-sun

Write a function called `draw-sun`, which consumes a `SunsetState`, and produces an image of a sun (a solid, 25 pixel circle), whose color is "yellow", when the sun's y-coordinate is greater than 225, "orange", when its y-coordinate is between 150 and 225, and "red" otherwise.

Contract+Purpose Statement

# \_\_\_\_\_ :: \_\_\_\_\_ → \_\_\_\_\_

# \_\_\_\_\_

Give Examples

examples :

\_\_\_\_\_ ( \_\_\_\_\_ ) is \_\_\_\_\_

\_\_\_\_\_ ( \_\_\_\_\_ ) is \_\_\_\_\_

\_\_\_\_\_ ( \_\_\_\_\_ ) is \_\_\_\_\_

end

(worksheet continues next page)

## Unit 5

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

# Animation Extension Worksheet

Describe the goal of your change: what new feature or behavior will it add to your animation?

Decrease the cat's hunger level by 2 and sleep level by 1 on each tick.

Draw a sketch for three distinct moments of the animation, focusing on the new behavior



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |

Make a To-Do List, and check off each as "Done" when you finish each one.

| Component       | When is there work to be done?  | To-Do                               | Done                     |
|-----------------|---|-------------------------------------|--------------------------|
| Data Structure  | If any new field(s) were added, changed or removed                    | <input type="checkbox"/>            | <input type="checkbox"/> |
| draw-state      | If something is displayed in a new way or position                    | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| next-state-tick | If the Data Structure changed, or the animation happens automatically | <input type="checkbox"/>            | <input type="checkbox"/> |
| next-state-key  | If the Data Structure changed, or a keypress triggers the animation   | <input type="checkbox"/>            | <input type="checkbox"/> |
| reactor         | If either next-state function is new                                  | <input type="checkbox"/>            | <input type="checkbox"/> |

Make a sample instance for each sketch from the previous page:

**FULLPET** = pet(100, 100)

**MIDPET** = pet(50, 75)

**LOSEPET** = pet(0, 0)

Write at least one NEW example for one of the functions on your To-Do list

---

---

next-state-tick(FULLPET) is pet(FULLPET.hunger - 2, FULLPET.sleep - 1)

next-state-tick(MIDPET) is pet(MIDPET.hunger - 2, MIDPET.sleep - 1)

next-state-tick(LOSEPET) is LOSEPET

---

---

If you have another function on your To-Do list , write at least one NEW example

---

---

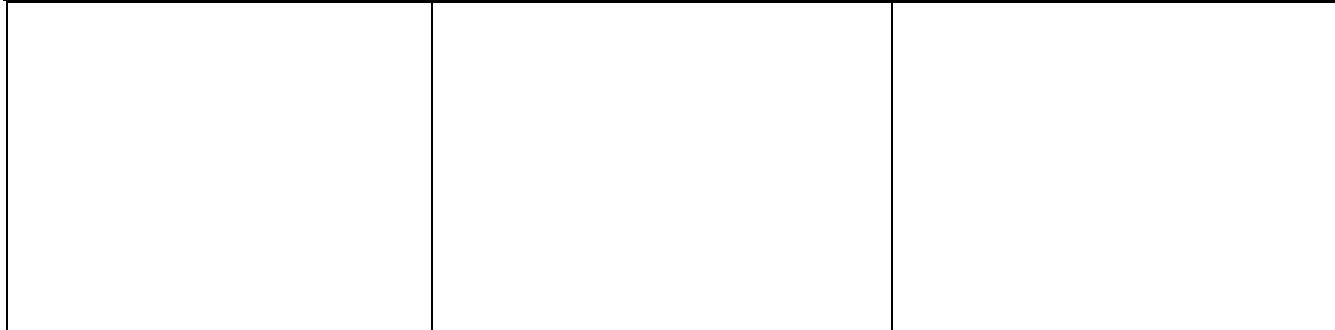
---

---

# Animation Extension Worksheet

Describe the goal of your change: what new feature or behavior will it add to your animation?

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |

Make a To-Do List, and check off each as "Done" when you finish each one.

| Component       | When is there work to be done?  | To-Do                    | Done                     |
|-----------------|---|--------------------------|--------------------------|
| Data Structure  | If any new field(s) were added, changed or removed                    | <input type="checkbox"/> | <input type="checkbox"/> |
| draw-state      | If something is displayed in a new way or position                    | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-tick | If the Data Structure changed, or the animation happens automatically | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-key  | If the Data Structure changed, or a keypress triggers the animation   | <input type="checkbox"/> | <input type="checkbox"/> |
| reactor         | If either next-state function is new                                  | <input type="checkbox"/> | <input type="checkbox"/> |

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write at least one NEW example for one of the functions on your To-Do list

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

If you have another function on your To-Do list , write at least one NEW example

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

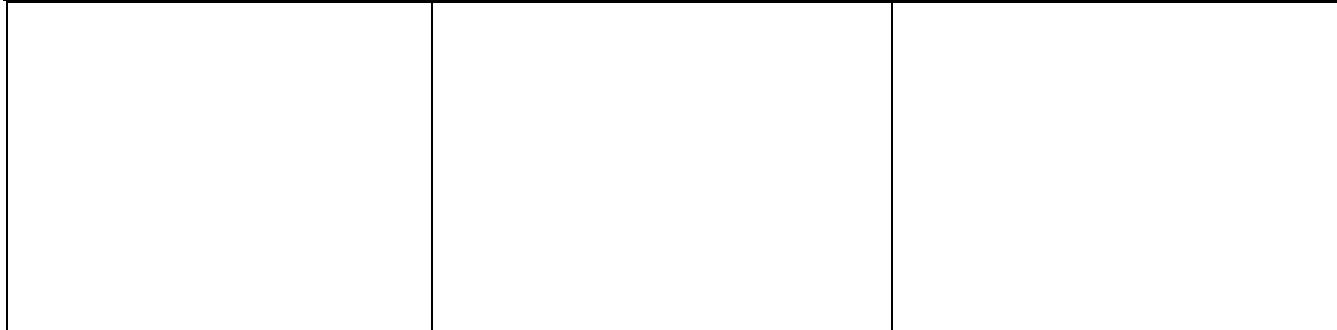
\_\_\_\_\_

\_\_\_\_\_

# Animation Extension Worksheet

Describe the goal of your change: what new feature or behavior will it add to your animation?

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |

Make a To-Do List, and check off each as "Done" when you finish each one.

| Component       | When is there work to be done?  | To-Do                    | Done                     |
|-----------------|---|--------------------------|--------------------------|
| Data Structure  | If any new field(s) were added, changed or removed                    | <input type="checkbox"/> | <input type="checkbox"/> |
| draw-state      | If something is displayed in a new way or position                    | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-tick | If the Data Structure changed, or the animation happens automatically | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-key  | If the Data Structure changed, or a keypress triggers the animation   | <input type="checkbox"/> | <input type="checkbox"/> |
| reactor         | If either next-state function is new                                  | <input type="checkbox"/> | <input type="checkbox"/> |

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write at least one NEW example for one of the functions on your To-Do list

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

If you have another function on your To-Do list , write at least one NEW example

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

## Build Your Own Animation

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

# Animation Design Worksheet

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

Make a To-Do List, and check off each as “Done” when you finish each one.

| Component       | When is there work to be done?  | To-Do                    | Done                     |
|-----------------|---|--------------------------|--------------------------|
| Data Structure  | If any new field(s) were added, changed or removed                    | <input type="checkbox"/> | <input type="checkbox"/> |
| draw-state      | If something is displayed in a new way or position                    | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-tick | If the Data Structure changed, or the animation happens automatically | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-key  | If the Data Structure changed, or a keypress triggers the animation   | <input type="checkbox"/> | <input type="checkbox"/> |
| reactor         | If either next-state function is new                                  | <input type="checkbox"/> | <input type="checkbox"/> |

Define the Data Structure

# a \_\_\_\_\_ **State** is \_\_\_\_\_

data \_\_\_\_\_ **State**:

| \_\_\_\_\_ ( \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ )

end

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write an example for one of the functions on the previous page:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

# Collision

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

## Distance:

The Player is at (4, 2) and the Target is at (0, 5).

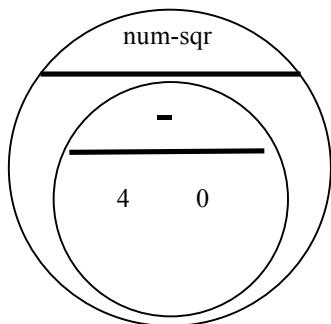
Distance takes in the player's x, player's y, character's x and character's y.

Use the formula below to fill in the EXAMPLE:

$$\sqrt{(4 - 0)^2 + (2 - 5)^2}$$

---

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
- : 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:

$$\text{Distance}^2 = (px - cx)^2 + (py - cy)^2$$

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

# 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:

\_\_\_\_\_ (\_\_\_\_\_)  
the user types...

is \_\_\_\_\_  
...which should become

\_\_\_\_\_ (\_\_\_\_\_)  
the user types...

is \_\_\_\_\_  
...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:

\_\_\_\_\_ (\_\_\_\_\_)  
the user types...

is \_\_\_\_\_  
...which should become

\_\_\_\_\_ (\_\_\_\_\_)  
the user types...

is \_\_\_\_\_  
...which should become

end

## Function

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

fun \_\_\_\_\_ (\_\_\_\_\_) :

end

# Animation Design Worksheet

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

Make a To-Do List, and check off each as “Done” when you finish each one.

| Component       | When is there work to be done?  | To-Do                    | Done                     |
|-----------------|---|--------------------------|--------------------------|
| Data Structure  | If any new field(s) were added, changed or removed                    | <input type="checkbox"/> | <input type="checkbox"/> |
| draw-state      | If something is displayed in a new way or position                    | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-tick | If the Data Structure changed, or the animation happens automatically | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-key  | If the Data Structure changed, or a keypress triggers the animation   | <input type="checkbox"/> | <input type="checkbox"/> |
| reactor         | If either next-state function is new                                  | <input type="checkbox"/> | <input type="checkbox"/> |

Define the Data Structure

# a \_\_\_\_\_ **State** is \_\_\_\_\_

data \_\_\_\_\_ **State**:

| \_\_\_\_\_ ( \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ )

end

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write an example for one of the functions on the previous page:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

# Animation Design Worksheet

Draw a sketch for three distinct moments of the animation



Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

Make a To-Do List, and check off each as “Done” when you finish each one.

| Component       | When is there work to be done?  | To-Do                    | Done                     |
|-----------------|---|--------------------------|--------------------------|
| Data Structure  | If any new field(s) were added, changed or removed                    | <input type="checkbox"/> | <input type="checkbox"/> |
| draw-state      | If something is displayed in a new way or position                    | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-tick | If the Data Structure changed, or the animation happens automatically | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-key  | If the Data Structure changed, or a keypress triggers the animation   | <input type="checkbox"/> | <input type="checkbox"/> |
| reactor         | If either next-state function is new                                  | <input type="checkbox"/> | <input type="checkbox"/> |

Define the Data Structure

# a \_\_\_\_\_ **State** is \_\_\_\_\_

data \_\_\_\_\_ **State**:

| \_\_\_\_\_ ( \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ )

end

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write an example for one of the functions on the previous page:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

# Animation Design Worksheet

Draw a sketch for three distinct moments of the animation

|  |  |  |
|--|--|--|
|  |  |  |
|--|--|--|

Sketch A

Sketch B

Sketch C

What things are changing?

| Thing | Describe how it changes |
|-------|-------------------------|
|       |                         |
|       |                         |
|       |                         |
|       |                         |

What fields do you need to represent the things that change?

| Field name (dangerX, score, playerIMG...) | Datatype (Number, String, Image, Boolean...) |
|---|--|
|   |  |
|   |  |
|   |  |
|   |  |

Make a To-Do List, and check off each as “Done” when you finish each one.

| Component       | When is there work to be done?  | To-Do                    | Done                     |
|-----------------|---|--------------------------|--------------------------|
| Data Structure  | If any new field(s) were added, changed or removed                    | <input type="checkbox"/> | <input type="checkbox"/> |
| draw-state      | If something is displayed in a new way or position                    | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-tick | If the Data Structure changed, or the animation happens automatically | <input type="checkbox"/> | <input type="checkbox"/> |
| next-state-key  | If the Data Structure changed, or a keypress triggers the animation   | <input type="checkbox"/> | <input type="checkbox"/> |
| reactor         | If either next-state function is new                                  | <input type="checkbox"/> | <input type="checkbox"/> |

Define the Data Structure

# a \_\_\_\_\_ **State** is \_\_\_\_\_

data \_\_\_\_\_ **State**:

| \_\_\_\_\_ ( \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_ )

end

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write an example for one of the functions on the previous page:

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write at least one NEW example for one of the functions on your To-Do list

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

If you have another function on your To-Do list , write at least one NEW example

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Make a sample instance for each sketch from the previous page:

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

\_\_\_\_\_ = \_\_\_\_\_

Write at least one NEW example for one of the functions on your To-Do list

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

If you have another function on your To-Do list , write at least one NEW example

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_