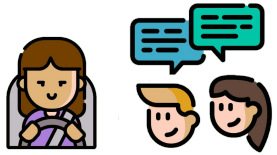
Sam the Butterfly

@BootstrapWorld

Introducing Sam

Save a copy.



Click “Run”, and use the arrow keys to investigate the program with your partner.

Complete **Introducing Sam**.

Students browse: code.pyret.org/editor#share=1YSPPa4BLhOG7_qhWdqQ-cG1u3yUUi-zZ

Introducing Sam

What did you notice about the program?

What did you see when Sam was at (0,0)? Why is that?

Students, write your response!

Introducing Sam

There are three functions defined in this file. What are they?

What *should* `is-safe-left` do?

What *should* `is-safe-right` do?

What *should* `is-onscreen` do?

Students, write your response!

Introducing Sam

Sam is safe on the left when...

`x > -50`

is-safe-left

Sam is safe on the right when...

`x < 690`

is-safe-right

Introducing Sam

1. Use the Design Recipe to complete **Left and Right**, so that these functions do what they should.

2. Open your saved Butterfly file and fix the code!

Note: when you type in `is-safe-right`, it may not do what you expect...

Introducing Sam

Directions: Use the Design Recipe to write a function 'safe-left?', which takes in an x-coordinate and checks to see if it is greater than -50

Contract and Purpose Statement

Every contract has three parts...

```
# is-safe-left :: Number -> Boolean
```
function name — *domain* — *range*

```
# Consumes x-coord, and produces true if it's greater than -50
```
what does the function do?

Examples

Write some examples, then circle and label what changes...

examples:

x-coord

```
is-safe-left ( 22 ) is 22 > -50
```
function name — *input(s)* — *what the function produces*

```
is-safe-left ( -91 ) is -91 > -50
```
function name — *input(s)* — *what the function produces*

end

Definition

Write the definition, giving variable names to all your input values...

```
fun is-safe-left ( x-coord ):
```
function name — *variable(s)*

```
  x-coord > -50
```
what the function does with those variable(s)

end

Introducing Sam

Does `is-safe-left` work correctly? **How do you know?**

Does `is-safe-right?` work correctly? **How do you know?**

Don't trust the behavior of a complex system! `example`s are where you look first. If they all pass, that's a strong hint that bug is elsewhere...

Students, write your response!

Protecting Sam on Both Sides

```
is-safe-left :: Number -> Boolean
# is Sam protected on the left side of the screen?
fun is-safe-left (x):  x > -50 end

is-safe-right :: Number -> Boolean
# is Sam protected on the right side of the screen?
fun is safe-right (x):  x <  690 end

is-onscreen :: Number -> Boolean
# is the butterfly protected on the left and the right?
fun is-onscreen(x): is-safe-left(x) end
```

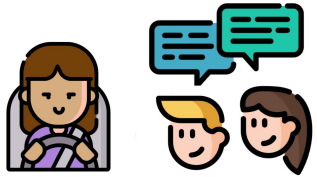
What does is-onscreen check for? What should it check for?

Students, write your response!

onscreen?

1. Complete **is-onscreen** in your workbook.

2. Select a driver to share to their screen, and protect Sam on both sides!



Synthesize

- How did it feel when you saw Sam hit both walls?
- Are there multiple solutions for `onscreen?`
- Is this *Top-Down* or *Bottom-Up* Design?

Students, write your response!

Boundary Detection in the Game

- Open your saved Game file and click Run.
- What happens when your game characters go offscreen?
- When should they come back?
- How does this relate to protecting Sam?

Boundary Detection

- Select a driver, and work together to add boundary detection to one game file.
- Add boundary detection to your own files!
- What are the “interesting” examples?