# Using Functions in Graphics

## Question: 1

Why are functions helpful in our programs?

1. Functions break down the problem into smaller chunks.
2. Functions allow parts of our code to be reusable.
3. Functions make our code more readable.
4. All of these

## Question: 2

How would we use a parameter color to control the color of a circle being drawn in the make\_circle function?

def make\_circle(color):

screen.create\_circle(50, 50, 100, 100, fill= 'color')

def make\_circle(color):

screen.create\_circle(50, 50, 100, 100, 'fill'= 'color')

def make\_circle(color):

screen.create\_circle(50, 50, 100, 100, fill= color)

def make\_circle(color):

screen.create\_circle(50, 50, 100, 100, color= fill)

# Using Loops in Graphics

## Question: 1

Which of the following code snippets would draw 5 circles on the canvas starting with a radius of 10 and ending with a radius of 50?

radius = 10

for i in range(5):

screen.create\_oval(0, radius, 0, radius)

radius = radius + 10

radius = 10

for i in range(5):

screen.create\_oval(0, 0, radius, radius)

radius = radius + 10

radius = 10

for i in range(5):

screen.create\_oval(radius, radius, radius, radius)

radius = radius + 10

radius = 10

for i in range(5):

screen.create\_oval(radius, 0, radius, 0)

radius = radius + 10

## Question: 2

Which of the following code snippets would draw rectangles that increase in height by 10 pixels on each iteration until the height of the rectangle is as tall as the canvas\_height?

rect\_height= 10

while rect\_height < canvas\_height:

screen.create\_rectangle(100, 0, 50, rect\_height)

rect\_height = canvas\_height

rect\_height= 10

while rect\_height < canvas\_height:

screen.create\_rectangle(100, 0, 50, rect\_height)

rect\_height = canvas\_height + 10

rect\_height= 10

while rect\_height < canvas\_height:

screen.create\_rectangle(100, 0, 50, rect\_height)

rect\_height = rect\_height + 10

rect\_height= 10

while rect\_height <= canvas\_height:

screen.create\_rectangle(100, 0, 50, rect\_height)

rect\_height = rect\_height + 10