

# Computational Thinking with Adinkra

## **What is Computer Programming?**

Whenever you use a computer for email, games, calculating, etc., you are using a computer program that someone wrote. A computer program or “software application” is just a list of instructions telling the computer what to do. It is first written in a language such as Java or C++, so that humans can read it, and then “compiled” or “interpreted” into the code that only computers can read. When that code is “executed” you see it working as a game or other application.

CSnap is a visual programming environment. CSnap Users are able to write programs to create animations, graphic designs, games, music, and other media. The advantage of using CSnap is that you simply drag and drop code blocks which snap together like Legos. Programming in a language like Java or C++ requires that you follow complicated syntax rules, so that merely misplacing a single comma can cause the program to fail. The blocks in CSnap only require that you stack them in the order you want them to execute.

## **What is Adinkra Computing?**

Adinkra Computing brings computer programming concepts together with Adinkra symbol carving and design. The discipline of ethnocomputing is used across different cultural contexts to reveal that present day computing concepts are embedded in local designs and artifacts. These designs and artifacts often have cultural and/or religious significance to groups of people in specific geographical locations. Adinkra symbols, being indigenous to Ghana, are examples of where computing and culture intersect.

Since Adinkra symbols have concepts with computational significance embedded in them, learning to program different Adinkra symbols using CSnap introduces students to deep connections between computing and their heritage/culture. Adinkra Computing helps students to learn computational thinking by drawing on local Ghanaian design knowledge.

## **Computational Thinking and Adinkra**

The real value in using CSnap to simulate Adinkra symbols is that students learn computational thinking. By computational thinking, we mean that students learn to recognize problems that they want to solve (problem recognition), define how to solve the problem (problem definition), and then break their solution down into manageable pieces (decomposition). We call this computational thinking or algorithmic thinking.

Like the algorithms that make up a computer program, the carving of Adinkra designs also results from specific step-by-step solutions. After determining what symbol will be carved, the Adinkra craftsman must evaluate their chosen algorithms of the symbol and decide the order of operation to create the design. In a very real way this process involves computational thinking.

### **Computer Science Topics Taught Through CSnap Adinkra Computing Tool:**

Agile Software Design - In agile software design, programmers concentrate on getting part of a working solution built as soon as possible, and then work on refining their solution to solve all aspects of the problem.

Flow of Control - The flow of control a CSnap program starts with the “Flag When Clicked” block, with each code block attached being executed in turn from top to bottom. Flow of control can be altered by using repeat blocks that repeat code more than once, or conditionally executed when a condition is met.

Looping- Looping alters the flow of control to repeat small sections of program code.

Variables- A variable is a storage location in memory, used during a code execution to reference an established value. In CSnap students can define both local variables for individual Sprites as well as global variables accessible by all Sprites. The scope of a variable refers to when a value stored in a variable is available for use.

Conditionals - Conditionals alters the flow of control to execute a group of code blocks only when an expression evaluates to true.

## Exemplary Symbols

### Akoma



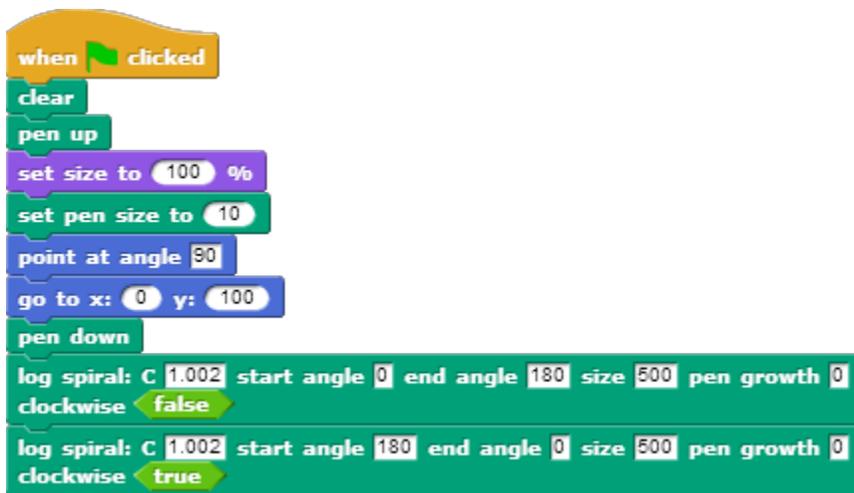
#### Meaning:

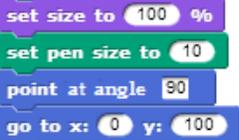
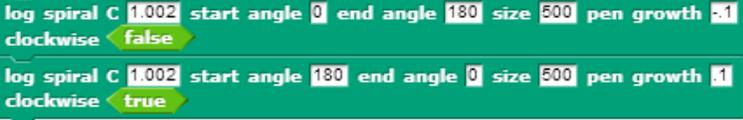
Akoma, translated in Akan to “heart”, is suggestive of the spiritual heart, not the physical heart. This symbol represents love, goodwill, patience, faithfulness, and fondness.

#### Applicable Computer Science Topics:

Flow of Control- The flow of control in a CSnap program starts with the “Flag When Clicked” block, with each code block attached being executed in turn from top to bottom.

#### CSnap Script to create Akoma:



	<p>Flow of control in CSnap always begins with a "Control" block that will start the script. As projects undergo remixing, the clear block erases anything on the stage. The program will run from the top down, executing each block in turn.</p>
	<p>The next block to execute in the script is pen up. Pen up allows movement across the stage without drawing.</p>
	<p>The set size to % block controls the size of the costume, in Adinkra computing this is the carving hand. Point at angle then sets the direction in which drawing will take place. Go to x:0, y:100 directs the pen to where the drawing will begin on the stage.</p>
	<p>Pen down places the pen back on the stage.</p>
	<p>The log spiral block is used to draw logarithmic spirals. It takes as parameters: the coilness constant C, the starting and ending angles of the design, size as a multiplier of the spiral on the screen, the changes in width as pen growth, and clockwise which can be set to true or false.</p>

## Adinkrahene



### Meaning:

Adinkrahene, "Chief of the Adinkra symbols", represents greatness and leadership. The symbol of concentric circles was originally on gold medallions worn by "soul washers", officials who performed religious rituals for the chief, such as bathing ceremonial swords. The circles are said to resemble the expanding ripples from a pebble tossed into a pool of water, just as a great leader's influence can ripple across the nation.

### Applicable Computer Science Topics:

Looping- Looping alters the flow of control to repeat small sections of program code.

## CSnap Script to create Adinkrahene:

```

when clicked
clear
set pen size to 18
pen up
go to x: 10 y: 0
pen down
circle diameter x position x 2.0 sweep 360
pen up
set pen size to 8
repeat 3
change x by 25
go to x: x position y: 0
pen down
circle diameter x position x 2.0 sweep 360
pen up
  
```

	Establishes an action to execute the code
	Clears the screen
	Using a large pen size here helps to fill in the inner circle
	Pen must be up to execute the next block without drawing a line
	Moves the pen to a now defined x and y position
	The pen is put down to begin "drawing" the symbol
	This block draws the first circle. When setting the circle diameter, it must be twice the circle radius. Therefore, we use the operator  where the x position has already been defined as the circle radius in the 5th block.
	The pen is lifted up here to prepare for the first translation which will occur at the beginning of the loop
	A smaller pen size is set to prepare for the three outer circles, which are thinner than the inner circle
	A loop is introduced here, which is a way to make a computer do the same thing, or a very similar thing, over and over again. We repeat this loop three times because Adinkrahene has three outer circles.
	The first action in the loop is to change the x position, which is set at 10, by a positive 25. This will put the first outer circle 25 units away from the original x position. After this first loop is completed the new x position will change by 25 units again. When the loop repeats three times there will be three x positions: 35, 60, and 85.
	After a new x position has been established, this block will move the pen to the desired x position, which is the radius of the new circle
	Because the pen is up last in the sequencing, it must be put down before a circle can be drawn
	This block draws each of the three outer circles. As done earlier, the circle diameter is set as twice the radius using an operator block to multiply the x position by two.
	At the end of the loop, the pen must be lifted up before going to the new x and y position

## **Akokonan**



### **Meaning:**

Akokonan translates as "hen's foot," and it resembles one. The saying that comes with the symbol is "The hen treads on her chicks, but she does not kill them". Thus, it represents the idea of "tough love", the combination of care and responsibility that we see in good leadership from our own parents or government institutions.

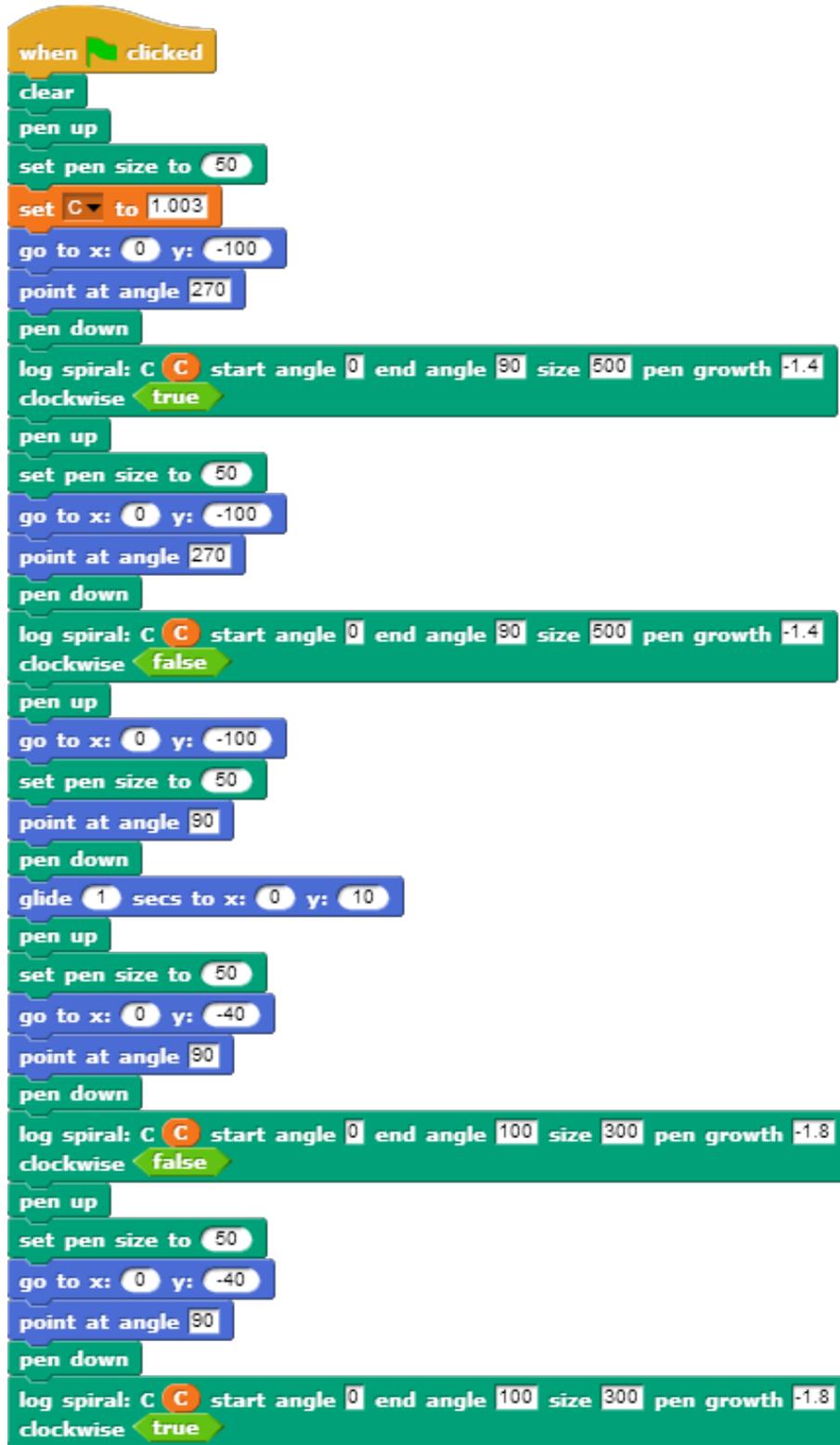
### **Applicable Computer Science Topics:**

Variables- A variable is a storage location in memory, used during a code execution to reference an established value. In CSnap students can define both local variables for individual Sprites as well as global variables accessible by all Sprites. The scope of a variable refers to when a value stored in a variable is available for use.

Conditionals - Conditionals alter the flow of control to execute a group of code blocks only when an expression evaluates to true.

In addition to variables in Akokonan, conditional control structures can be examined as they are used in the log spiral block to draw Akokonan. The code for the log spiral block can be manipulated to see how conditionals are implemented for clockwise and anticlockwise spiral drawings. To do this, right click on the log spiral block and choose 'Edit'. A block editor window will open, showing the block definition below. Scroll to the bottom of the block definition window and notice how the 'if' control blocks use Boolean logic operators 'true' and 'false' to determine whether or not the code in the 'if' control block should execute.

## CSnap Script to create Akokonan:



```
when green flag clicked
clear
pen up
set pen size to 50
set C to 1.003
go to x: 0 y: -100
point at angle 270
pen down
log spiral: C C start angle 0 end angle 90 size 500 pen growth -1.4
clockwise true
pen up
set pen size to 50
go to x: 0 y: -100
point at angle 270
pen down
log spiral: C C start angle 0 end angle 90 size 500 pen growth -1.4
clockwise false
pen up
go to x: 0 y: -100
set pen size to 50
point at angle 90
pen down
glide 1 secs to x: 0 y: 10
pen up
set pen size to 50
go to x: 0 y: -40
point at angle 90
pen down
log spiral: C C start angle 0 end angle 100 size 300 pen growth -1.8
clockwise false
pen up
set pen size to 50
go to x: 0 y: -40
point at angle 90
pen down
log spiral: C C start angle 0 end angle 100 size 300 pen growth -1.8
clockwise true
```

The script is a Scratch script designed to create the Akokonan logo. It begins with a 'when green flag clicked' event. The initial setup includes clearing the stage, setting the pen size to 50, and defining a constant 'C' as 1.003. The drawing process starts at the point (0, -100) and is oriented at 270 degrees. The first spiral is drawn clockwise from 0 to 90 degrees with a size of 500 and a pen growth of -1.4. This is followed by a second spiral drawn counter-clockwise with the same parameters. The script then moves to the point (0, -100) again, sets the pen size to 50, and orients the pen at 90 degrees. A glide block moves the pen to (0, 10) over 1 second. From there, it moves to (0, -40) and orients the pen at 90 degrees. The final part of the script consists of two more spirals: one drawn counter-clockwise from 0 to 100 degrees with a size of 300 and a pen growth of -1.8, followed by a second spiral drawn clockwise with the same parameters.

	<p>Here the script begins. Clear erases everything previously on the stage. Pen up, like the name, allows movement around the screen without leaving traces. Alas the pen size is set.</p>
	<p>The variable C is assigned the value 1.003</p>
	<p>The pen is moved to <math>x=0</math>, <math>y=-100</math> and the drawing angle is set to 270 degrees. The pen is now ready to draw and thus put down.</p>
	<p>The log spiral block uses the variable assignment for C as the input for the C parameter</p>
	<p>This code lifts the pen, restores the pensize to 50, moves the pen to <math>x=0</math>, <math>y=-100</math>, points the drawing angle to 270 degrees and puts the pen down ready to draw.</p>
	<p>Again the log spiral block uses the variable assignment for C as the input for the C parameter.</p>
	<p>These blocks draw the center line in Akokonan</p>
	<p>This process should now be familiar and is very similar to the first half of this script.</p>
	<p>The point at angle 90 is used to direct these log spirals opposite the first two</p>
	<p>Come the last log spiral one should hopefully see the benefit of using C as a variable. Changing this one variable easily manipulates the overall size of the Akokonan.</p>

## Editing the Log Spiral Block:

log spiral C C start angle startangle end angle endangle  
size Size pen growth pengrowth clockwise clockwise?

script variables  
x\_origin y\_origin starting direction beta t tinc roffset  
r

Variables that are set by the user. Note that C, which is how "coiled" the spiral is, will be converted to the standard exponent constant "beta."

Variables that are internal to this script only. "beta" is the exponent constant in standard form, "t" is "theta" (the angle at which you calculate the radius). Each time you change the angle by "tinc" (theta increment) you evaluate the radius. A radius off-set ("roffset") ensures the spiral will start where the carving hand is located.

if C = 1 or C < 1  
say C cannot be less than or equal to 1 for 2 secs  
stop all

if C > 1  
say C cannot be greater than 1.1 for 2 secs  
stop all

Check to see if the exponent constant has a bad value, and if so, stop the script so it doesn't crash the browser.

set x\_origin to x position Remember the place you were at when this script started, so that you know where to start drawing from.  
set y\_origin to y position  
set starting direction to angle Remember the angle you were at when this script started, so that you know the orientation to draw from.

if startangle > endangle  
set starting direction to angle + 90 Normally you would have a starting angle less than the ending angle, so if you are drawing it in reverse, from end to start, you need to add 90 (I am not sure why).

set beta to ln of C Convert C to the standard exponent constant beta.  
set t to startangle Start drawing from the start angle. Normally there would be a gap between the origin and the starting radius. This "radius off set" eliminates that gap, thus maintaining the analogy to carving a continuous path.  
set roffset to Size x e^beta of startangle - Size

if endangle > startangle  
set tinc to 4 The normal direction for drawing a spiral. Thus the angle increment "tinc" is positive 4. The increment size seemed large enough to avoid delays and small enough to keep the curve smooth.  
if clockwise  
turn atan of 1 / beta degrees The carving hand needs to align with the tangent to the spiral.  
else  
turn atan of 1 / beta degrees

else  
set tinc to -4 The reversed direction for drawing a spiral. Thus the angle increment "tinc" is negative 4.  
if clockwise  
turn atan of 1 / beta + 180 degrees The carving hand needs to align with the tangent to the spiral.  
else  
turn atan of 1 / beta + 180 degrees

repeat abs of endangle - startangle / tinc  
set t to t + tinc  
set r to Size x e^beta of beta x t - Size Now we start the iterative loop for drawing the spiral. The number of iterations is the angle sweep divided by the increment. The radius at each angle is determined by the standard equation for a log spiral.

if startangle > endangle  
if clockwise  
turn tinc degrees The carving hand has already been aligned with the tangent to the spiral, so it just needs to keep adjusting with each increment.  
else  
turn -tinc degrees  
else  
if clockwise  
turn tinc degrees  
else  
turn -tinc degrees

change pen size by pengrowth Grow the pen with each increment

if clockwise = false  
go to x:  
x\_origin + r x cos of t + starting direction  
roffset x cos of startangle + starting direction  
y:  
y\_origin + r x sin of t + starting direction  
roffset x sin of startangle + starting direction  
else  
go to x:  
x\_origin + r x cos of t x -1 + starting direction  
roffset x cos of startangle x -1 + starting direction  
y:  
y\_origin + r x sin of t x -1 + starting direction  
roffset x sin of startangle x -1 + starting direction

The radius is used to calculate an updated x,y coordinate. The pen simply moves to each successive coordinate point.