

# Chaos

## Teacher Notes and Answers

7 8 9 10 11 12



TI-Nspire



Coding



Student



60 min

## Introduction

This activity involves repeatedly finding midpoints, starting with a point randomly placed inside a triangle. Once the first point has been placed, subsequent points are placed based on the following algorithm:

A vertex is chosen at random. The new point is placed half way between the most recent point and the randomly selected vertex.

To help automate this process the triangle is placed on the Cartesian plane.

### Question: 1.

Let the vertices A, B and C be located at: (0, 0); (6, 8) and (10, 2) respectively. The first point,  $P_1$  is located at (4, 4)

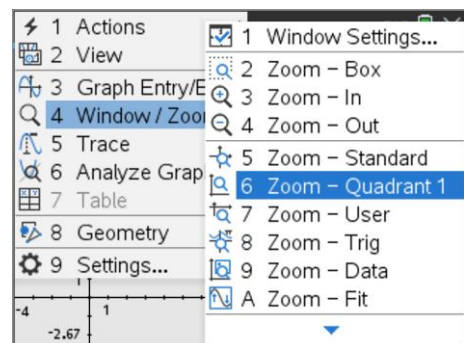
- Vertex A is selected at random. Determine the coordinates of the second point:  $P_2$
- Vertex B is now selected (random). Determine the coordinates of the third point:  $P_3$
- Vertex A is now selected (random). Determine the coordinates of the fourth point:  $P_4$
- Select a vertex at (random) and determine the coordinates of the fifth point:  $P_5$
- Imagine this process is continued so that 200 points are plotted. What do you think the plot might look like?

## Setting up the Triangle


### Instructions:

Start a new document; insert a Graphs application.

Use the zoom menu to focus on Quadrant 1.



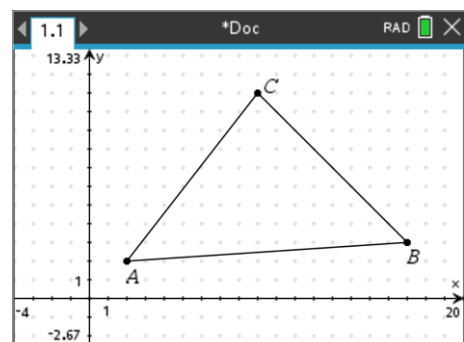
Use the Geometry menu to draw a triangle.

 > **Geometry** > **Shapes** > **Triangle**



If you want vertices A, B and C to have integer coordinates, switch on the grid and place each vertex on a grid point.

Vertex labelling can be switched on/off through the settings menu.



The coordinates of each vertex are required for the program.

Move the mouse (cursor) over the top of a vertex then press:

**ctrl** + **menu**

This will display the contextual menu shown opposite, select:

**Coordinates and Equations.**

Do this for each of the triangle's vertices.

The next task is to link each of the coordinates to variables that can be used in the program.

Move the mouse over the top of the x coordinate for point A, then press:

**ctrl** + **menu**

This will display the contextual menu shown opposite, select:

**Store**

For the x coordinate (abscissa) of point A store the variable as XA.

Repeat this process for each of the coordinates as follows:

A (XA, YA)

B (XB, YB)

C (XC, YC)

Notice that each of the coordinates becomes **bold** once it is linked.

Hovering the mouse over the coordinate confirms that it is linked to the corresponding variable.

An initial point needs to be plotted. It is not easy to confine the point to inside region of the triangle, however the individual user can ensure it is inside the triangle to begin and then explore what happens if the point is moved outside the triangle.

Place a point inside the triangle:

**menu** > **Geometry** > **Points and Lines** > **Point**

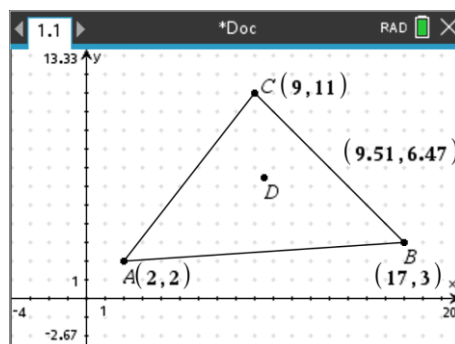
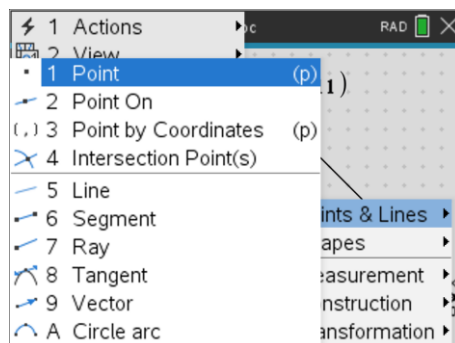
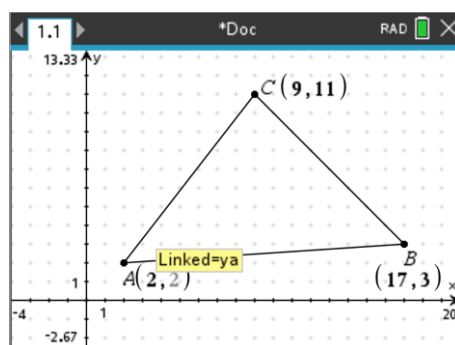
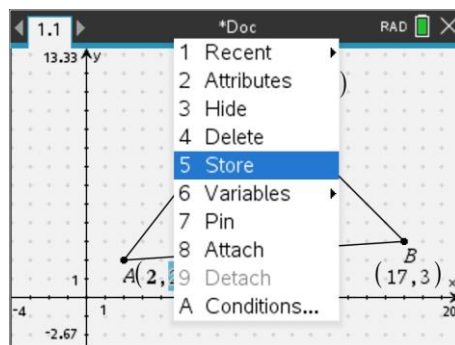
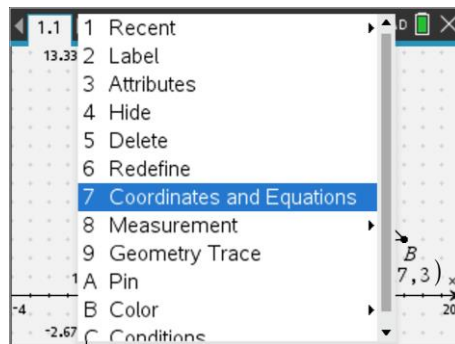
**Note:** A short-cut on TI-Nspire CX II is to simply press "P".

Display the coordinates of the point and store as follows:

xv for x coordinate (abscissa)

yv for y coordinate (ordinate)

By storing all the coordinates as variables, the program will be able to interact with the diagram.



## Writing the Program

Insert a new page with the Program Editor. Call the program: Chaos.

The first task for the program is to identify how many midpoints will be plotted. A request statement could be used to input the number of points; however, it will be useful for automation purposes to have this value in the original definition of the program.

Navigate to the chaos() definition and insert an 'n'.

The number of points to be plotted is stored in 'n', two empty lists can be created to record these points. A list filled with zeros can be created with the sequence command: seq(expression, variable, start, low, high)

```
xp:=seq(0,x,1,n)
```

Create another list for the y – points.

**Note:** The variable 'x' used to create the list is not used in this example but necessary for the syntax of the sequence command.

It is useful to keep track of the initial point (xv, yv). To avoid changing this in the program, the program will use (xm, ym) to represent all subsequent midpoints. While the first point is not a midpoint, it is useful to store xv in xm and yv in ym for the purpose of the loop (next step).

The number of times the loop needs to be executed is known (n), so a FOR loop is appropriate. Each time the loop is executed it will need to:

- Randomly select a vertex
- Calculate the new midpoint
- Store and update the current midpoint.

Insert a For loop:

menu > **Control** > **For ... EndFor**

The randint(lower,upper) command generates a random integer between the lower and upper limits. There are only three vertices to select from, so the random integer needs to vary between 1 and 3.

The randint() command can be typed directly or retrieved from the catalogue.

The randomly generated value needs to be stored so that it may be used to identify which vertex has been selected.

```
* chaos 1/1
Define chaos(n)=
Prgm
[ ]
EndPrgm
```

```
* chaos 3/3
Define chaos(n)=
Prgm
xp:=seq(0,x,1,n)
yp:=seq(0,x,1,n)
[ ]
EndPrgm
```

```
* chaos 5/5
Define chaos(n)=
Prgm
xp:=seq(0,x,1,n)
yp:=seq(0,x,1,n)
xm:=xv
ym:=yv
[ ]
EndPrgm
```

```
* chaos 5/7
Define chaos(n)=
Prgm
xp:=seq(0,x,1,n)
yp:=seq(0,x,1,n)
xm:=xv
ym:=yv
For i,1,n
[ ]
EndFor
EndPrgm
```

```
* chaos 7/8
Define chaos(n)=
Prgm
xp:=seq(0,x,1,n)
yp:=seq(0,x,1,n)
xm:=xv
ym:=yv
For i,1,n
v:=randint(1,3)
[ ]
EndFor
```

If the random value is 1, then the new midpoint is halfway between the current value ( $x_m, y_m$ ) and vertex A.

Examine the code included opposite to see how this is being done.

Add two more **If ... Then** commands and include the instructions for vertex B or C.

The program is almost finished. The coordinates need to be stored each time the loop is executed. The value of 'i' (loop counter) increments automatically each time the loop is executed. The lists xp and yp contain 'n' locations to store xv and yv.

Once these values have been stored the program is ready to run.

Press **Ctrl + R** to run the program.

Run the program and generate 50 points from a Calculator application.

```
chaos(50)
```

Navigate back to the Graphs application and create a scatterplot for the points (xp, yp).

Sample shown opposite.

```

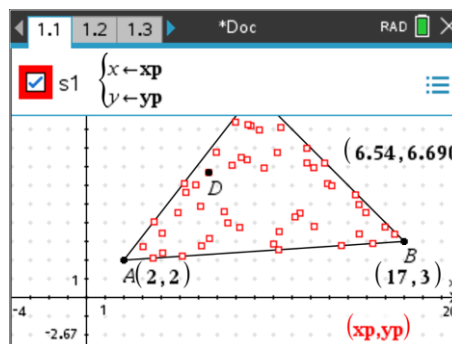
1.1 1.2 1.3 *Triangle ...aos RAD 9/12
* chaos
For i,1,n
v:=randint(1,3)
If v=1 Then
xm:= $\frac{xm+xa}{2}$ 
ym:= $\frac{ym+ya}{2}$ 
EndIf

```

```

1.1 1.2 1.3 *Triangle ...aos RAD 20/21
* chaos
If v=3 Then
xm:= $\frac{xm+xc}{2}$ 
ym:= $\frac{ym+yc}{2}$ 
EndIf
xp[i]:=xm
yp[i]:=ym
EndFor

```



### Question: 2.

An important stage in any programming task is to check the results.

- Place the initial point close to the centre of the triangle and record the coordinates of this point.  
**Note:** You can also select the coordinates and enter their values directly.
- Run your program and generate 10 points. Navigate back to the Graphs application to see the scatterplot. Access the **Trace > Graph Trace** command from the menu, the coordinates of the first midpoint will be displayed. Write down the coordinates of the first midpoint.
- Write down the coordinates of your first point, the midpoint and identify which vertex was selected.
- Using the trace tool, identify the second midpoint, write down the coordinates and confirm that they have been calculated correctly.

### Question: 3.

Run your program 3 to 4 times generating 50 points each time. (The scatterplot will automatically update each time). Do you notice any pattern with the points or do they appear random / chaotic?

**Question: 4.**

Run your program 3 to 4 times generating 150 points each time. (The scatterplot will automatically update each time). Do you notice any pattern with the points this time?

**Question: 5.**

Run your program 3 to 4 times generating 500 points each time. Do you notice any pattern with the points this time? Are they chaotic?

**Question: 6.**

The vertices on the triangle are dynamic. Move points A, B and C to new locations then run the program again. Does the form of the triangle alter your answer to the previous question?

**Extension**

The program can be launched from the Notes application within a Maths-Box. (Avoid using a large number of points!) Now what happens when you move the vertices or starting point from within the Graphs application?

What happens if instead of using a midpoint, the new points are weighted so that they are  $\frac{1}{3}$  of the distance to the vertex each time instead of the midpoint?

Hint: Consider using a weighted average.