Parabolic Hoops

How's your jump shot? Can you make a basket from anywhere within reasonable range of the hoop? You can with mathematics!! It's as easy as $y = a(x - h)^2 + k$.

In this project, you will create a virtual basketball court. With each new shot, your program will randomly place you, the shooter, a random distance from a 10 ft hoop. The game will generate a reasonable (x,y) location for the maximum point of your parabolic shot. Using the location of the hoop and the vertex, you the shooter will calculate the amplitude, a, needed to "Swish" nothing but net!

Objectives:

Programming Objectives:

- Use the randint() function to generate random integers.
- Use the plot library to draw line segments and plot points
- Use loops to repeat code
- Use if statements to make selections

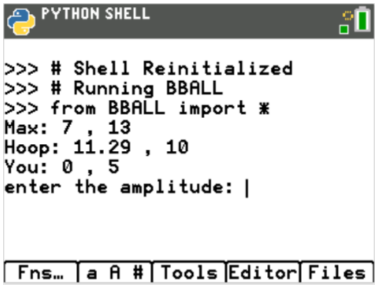
Math Objectives:

- Use the vertex form of a quadratics to solve problems
- Use the absolute value function to validate solutions.

Math Course Connections: Algebra 1 or Algebra 2- Vertex form of a parabola and absolute value

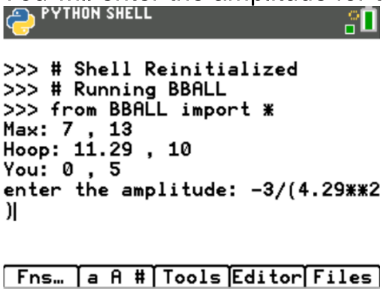
Your project will:

Generate a generate the scene.



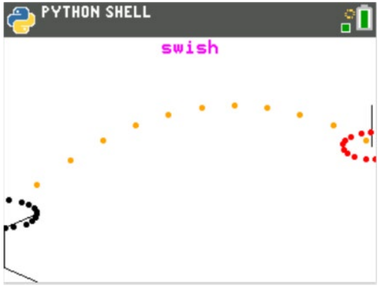
```
>>> # Shell Reinitialized
>>> # Running BBALL
>>> from BBALL import *
Max: 7 , 13
Hoop: 11.29 , 10
You: 0 , 5
enter the amplitude: |
```

You will enter the amplitude for the shot.

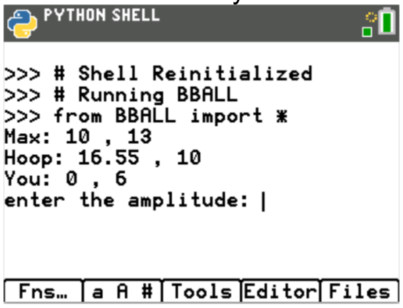


```
>>> # Shell Reinitialized
>>> # Running BBALL
>>> from BBALL import *
Max: 7 , 13
Hoop: 11.29 , 10
You: 0 , 5
enter the amplitude: -3/(4.29**2
)|
```

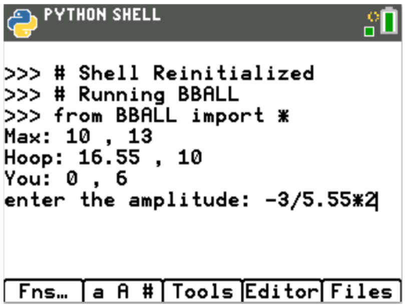
If you calculate correctly: Swish!!



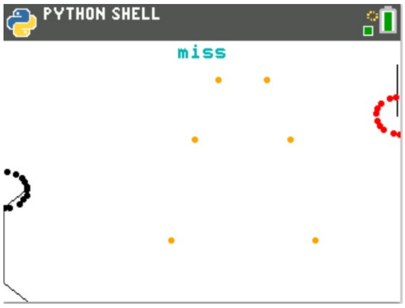
Calculate Incorrectly….Miss… "air ball"

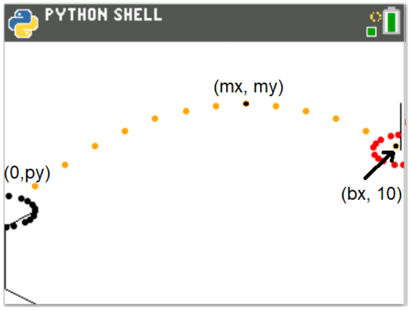


```
>>> # Shell Reinitialized
>>> # Running BBALL
>>> from BBALL import *
Max: 10 , 13
Hoop: 16.55 , 10
You: 0 , 6
enter the amplitude: |
```



```
>>> # Shell Reinitialized
>>> # Running BBALL
>>> from BBALL import *
Max: 10 , 13
Hoop: 16.55 , 10
You: 0 , 6
enter the amplitude: -3/5.55*2|
```





1. The path of your basketball will be parabolic.

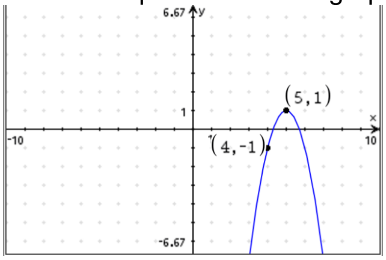
That means, it can be written in the form $y = a(x - h)^2 + k$.

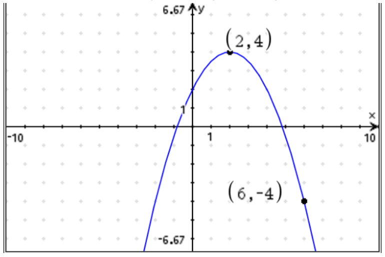
a.) Let's review the properties for the vertex form of a parabola, $y = a(x - h)^2 + k$.

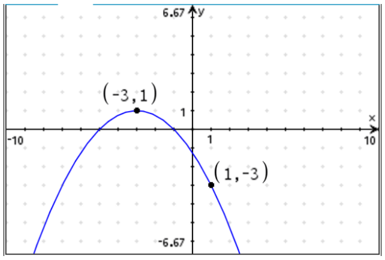
How does ***a*** effect the graph of a parabola?

How do ***h*** and ***k*** effect the graph of a parabola?

Write the equation for each graph below in the form $y = a(x - h)^2 + k$.







y =

y =

y =

b.) Graph each of your equations above on your calculator. Verify your equation contains the points in the diagrams.

Teacher Tip:

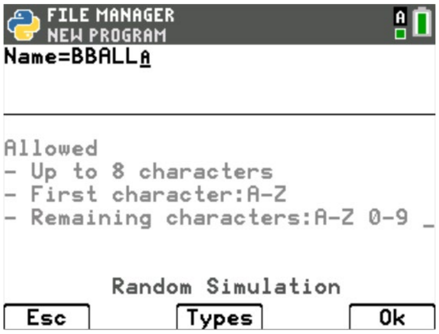
a is the amplitude for the parabola. It stretches the graph.
(h,k) is the vertex of the graph.

$y = -2(x - 5)^2 + 1$ $y = -0.5(x - 2)^2 + 4$ $y = -0.25(x + 3)^2 + 1$

2. The first step in the coding project will be to create a Python Random Simulation document.

Create a new python project named “BBALL”.

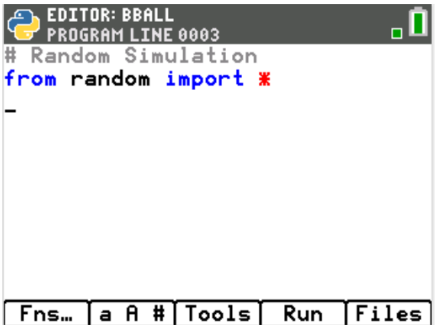
Select “Random Simulations” from the Type Menu.



```
Name=BBALL

Allowed
- Up to 8 characters
- First character:A-Z
- Remaining characters:A-Z 0-9 _

Random Simulation
```

This will automatically import the random library.

You need the randint function from this library to generate random integers.

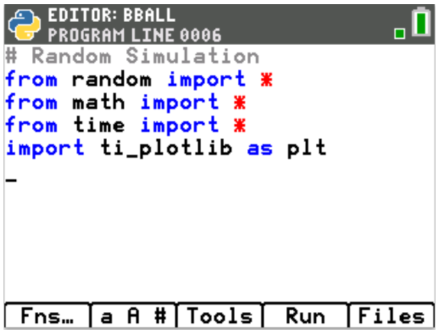


```
# Random Simulation
from random import *
```

Fns > Modul > math > from math import *

Fns > Modul > time > from time import *

Fns > Modul > ti_plotlib > import ti_plotlib as plt



```
# Random Simulation
from random import *
from math import *
from time import *
import ti_plotlib as plt
```

math library

You will use the fabs function from the math library to find the absolute value.

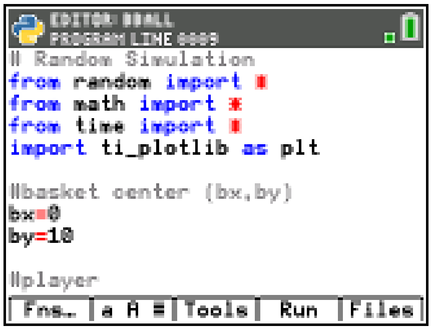
time library

ti_plotlib

You will use many of the plotting features to draw your objects.

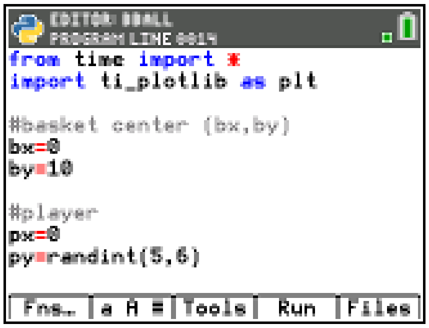
4. Use two variables to keep track of the location of the basket. Create the variables bx and by.

```
#basket center (bx, by)
bx = 0
by = 10
```



```
# Random Simulation
from random import *
from math import *
from time import *
import ti_plotlib as plt

#basket center (bx,by)
bx=0
by=10

#player
```

*The line ***#basket center(bx, by)*** is known as a comment. Programmers use comments to document parts of their code. You may omit this line if you like. Comments start with a # symbol. Notice the comment appears in a light gray color.



EDITOR: BBALL
PROGRAM LINE 0014

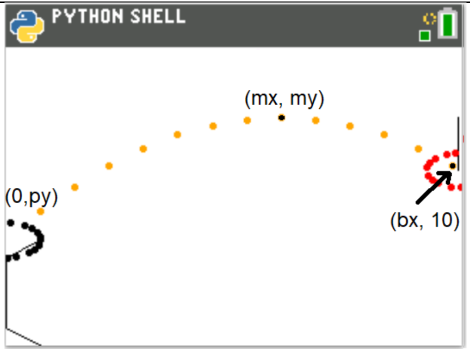
```
from time import *
import ti_plotlib as plt

#basket center (bx,by)
bx=0
by=10

#player
px=0
py=randint(5,6)
```

For now, let's say the ball will leave the player's hand from either 5 or 6 feet off the ground. The player's x value will be 0.

px = 0
py = randint(5, 6)

Fns > Modul > random > randint

**If you would like the user's shot height to be from the set {5, 5.1, 5.2, ….6}
Type py = randint(50,60)/10



The picture to the left shows the set up for the player, the hoop, and the path for the ball.

The player, will have the point (0, py) where py is the height the ball is released.

The hoop will be regulation in height. Therefore, it will have a height of 10 ft. You will calculate the x value, bx, the distance to the basketball hoop.

The vertex for the parabola (mx, my) will have a height, my. It will be higher than 10 ft, the height of the hoop. The x value, mx, will be between 0 and bx.

6. The maximum height, **my,** should be over 10 feet to keep it above the rim.
To keep the shot with realistic values, let my be a random integer between 11 and 16 feet.
my = randint(11,16)

The typical free throw line is 15 feet from the basket.
Assume the max height of the toss occurs anywhere from 3 to 10 feet away from the player.

mx = randint(6,12)

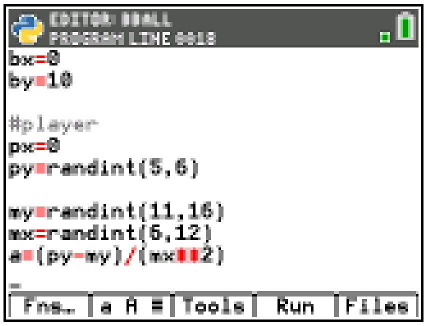
Now that you have py, mx, and my; where is bx? To answer this question, you must find a, the amplitude first.
Use $y = a(x – h)^2 + k$ to find and equation for a.

7. Did you say: $y = a(x – h)^2 + k$

$py = a(0 – mx)^2 + my$

$py – my = a(mx^2)$

$\frac{py-my}{mx^2} = a$



```
bx=0
by=10

#player
px=0
py=randint(5,6)

my=randint(11,16)
mx=randint(6,12)
a=(py-my)/(mx**2)
```

Add the three lines:

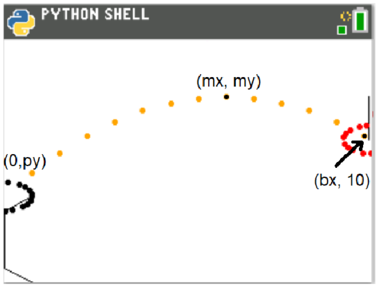
```
my = randint(11,16)
mx = randint(6,12)
a = (py – my) / (mx**2)
```

Fns > Modul > random > randint

*In python, **2 is used to square a number.
Press the key [x^2] to get **2.

Teacher Tip:
Incorrect parenthesis use is a common mistake. Ensure student calculate a correctly with parenthesis.

8.Now that you have py, mx, my and a how can you find the value of bx?



9. Did you say use the formula $y = a(x – h)^2 + k$?

The vertex will be (mx,my). Substitute these values into the appropriate place in the formula.
The point (bx,10) is one (x,y) point on the equation. Substitute these values into the appropriate place in the formula.

10. Solve the equation for bx: $10 = a(bx – mx)^2 +my$

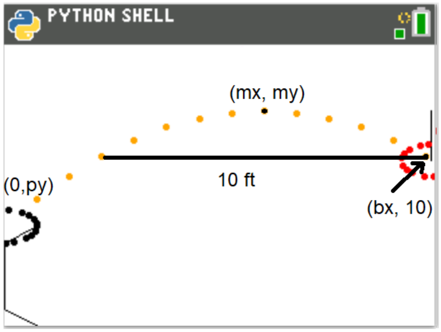
11. Did you get: $10 - my = a(bx - mx)^2$

$$\frac{10-my}{a} = (bx - mx)^2$$

$$\pm\sqrt{\frac{10-my}{a}} = bx - mx$$

$$mx \pm\sqrt{\frac{10-my}{a}} = bx$$

12. The parabola has to x-values that reach a height of 10 feet.

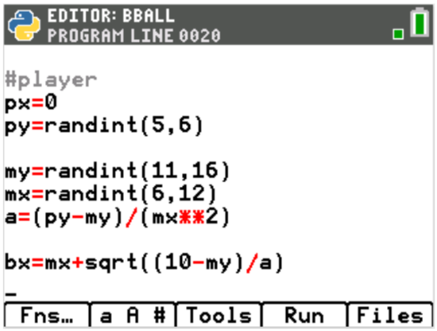


You want the second time the ball reaches 10.

Use the + not the – sign in your equation.

bx = mx + sqrt((10 – my) / a)

Fns > Modul > math > sqrt

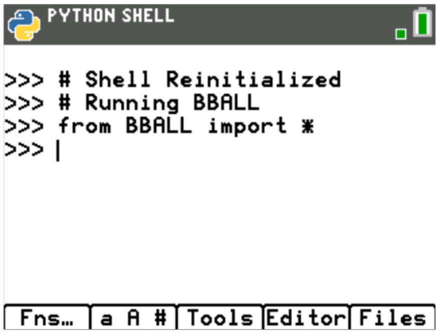


```
#player
px=0
py=randint(5,6)

my=randint(11,16)
mx=randint(6,12)
a=(py-my)/(mx**2)

bx=mx+sqrt((10-my)/a)
_
```

13. Run your code. [Trace]

You should get the screen to the right. If not, fix your errors.

If you reach the page on the right, press [Trace] to get back to the editor.



```
>>> # Shell Reinitialized
>>> # Running BBALL
>>> from BBALL import *
>>> |
```

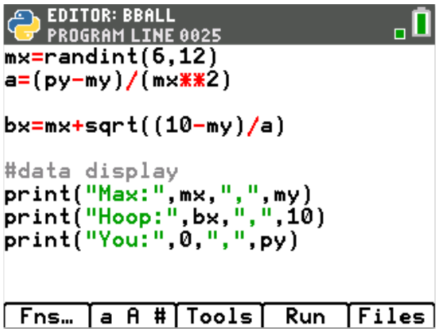
14. Now to display the data points for the shot. Add the lines:

print(“Max”, mx, “,”, my)
print(“Hoop”, bx, “,”, 10)
print(“You”, 0, “,”, py)

Fns > I/O > print

The words “Max:”, “Hoop:”, “You:”, and “,” should all be green because they are string. Everything else should have black font.

The [a A #] key might make it easier to type.

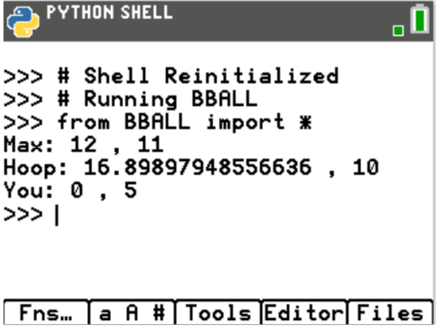


```
mx=randint(6,12)
a=(py-my)/(mx**2)

bx=mx+sqrt((10-my)/a)

#data display
print("Max:",mx,",",my)
print("Hoop:",bx,",",10)
print("You:",0,",",py)
```

Run your code. [Trace]

Does it display your data?
The sample on the right is only one possible situation.

Sample Random Output:



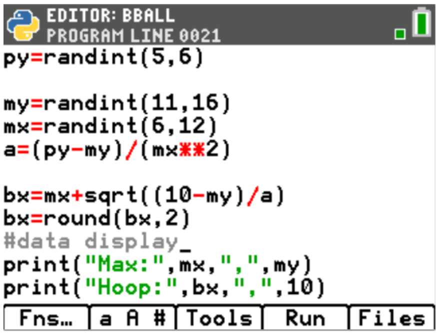
```
>>> # Shell Reinitialized
>>> # Running BBALL
>>> from BBALL import *
Max: 12 , 11
Hoop: 16.89897948556636 , 10
You: 0 , 5
>>> |
```

15. To simplify the display, round the hoop's x location to 2 decimal places.

    Add the line: bx = round(bx,2)
    before the print statements.

    Fns > Type > round

    Execute your code. [Trace].
    Verify your display now rounds to two places.

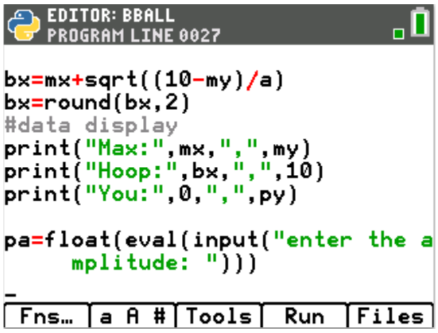


```
py=randint(5,6)

my=randint(11,16)
mx=randint(6,12)
a=(py-my)/(mx**2)

bx=mx+sqrt((10-my)/a)
bx=round(bx,2)
#data display_
print("Max:",mx,",",my)
print("Hoop:",bx,",",10)
```

16. Now add a line that lets the user enter a guess for the amplitude.
    By itself, input lets the user enter a string value. To evaluate the input and save it as a float number, you must float(eval(input.

    Type:
    pa = float(eval(input(“enter the amplitutde: “)))

    Fns > Type > float
    Fns > I/O > eval
    Fns > I/O > input

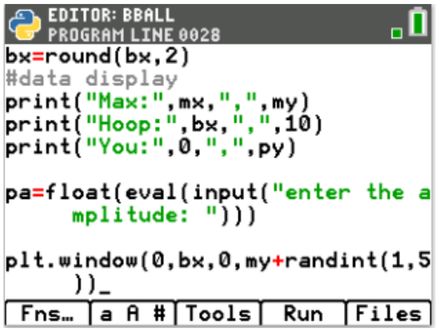


```
bx=mx+sqrt((10-my)/a)
bx=round(bx,2)
#data display
print("Max:",mx,",",my)
print("Hoop:",bx,",",10)
print("You:",0,",",py)

pa=float(eval(input("enter the a
     mplitude: ")))
_
```

17. Set up the window.
    Fns > Module > ti_plotlib > Setup > window

    Let the domain be [0, bx] and the range [0, my+ randint(1,5)]
    The randint will give some extra cushion in the window.

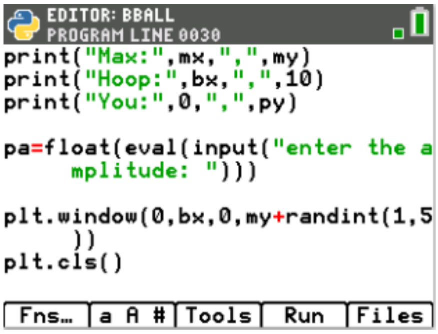
    plt.window(0, bx, 0, my+randint(1,5))



```
bx=round(bx,2)
#data display
print("Max:",mx,",",my)
print("Hoop:",bx,",",10)
print("You:",0,",",py)

pa=float(eval(input("enter the a
     mplitude: ")))

plt.window(0,bx,0,my+randint(1,5
     ))_
```

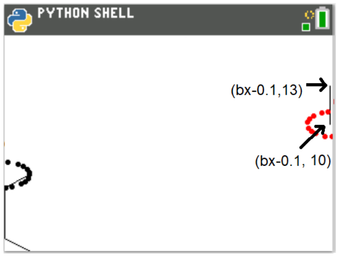
18. Clear the window.
    Fns > Module > ti_plotlib > setup > clear

    plt.cls()



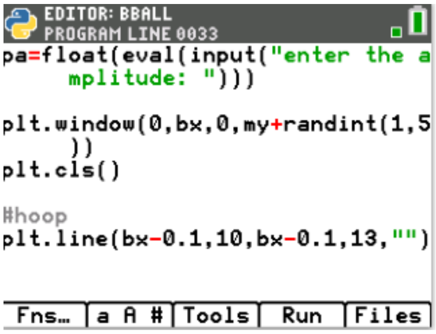
```
print("Max:",mx,",",my)
print("Hoop:",bx,",",10)
print("You:",0,",",py)

pa=float(eval(input("enter the a
    mplitude: ")))

plt.window(0,bx,0,my+randint(1,5
    ))
plt.cls()
```

19. Now to draw the basketball hoop and backboard.



Backboard:
A line segement will represent the backboard. The height of the board will be 3 ft. The picture to the left has the backboard placed at x=bx-0.1. The base of the board is at y = 10, while the top is at y = 13.

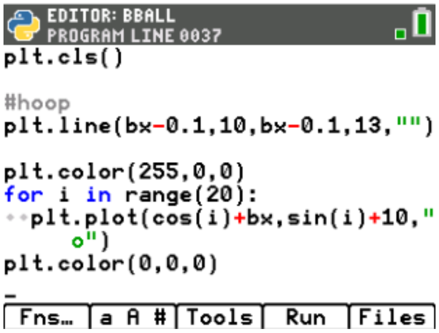
To create the board type:
plt.line(bx-0.1, 10, bx-0.1, 13)



```
pa=float(eval(input("enter the a
    mplitude: ")))

plt.window(0,bx,0,my+randint(1,5
    ))
plt.cls()

#hoop
plt.line(bx-0.1,10,bx-0.1,13,"")
```

Fns > Modul > ti_plotlib > draw> line

Red Basket:
To make the hoop red, set the rgb color to: 255, 0, 0.



```
plt.cls()

#hoop
plt.line(bx-0.1,10,bx-0.1,13,"")

plt.color(255,0,0)
for i in range(20):
  plt.plot(cos(i)+bx,sin(i)+10,"
    o")
plt.color(0,0,0)
```

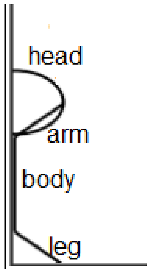
The plot library doesn't have a circle command, however, we can make one using a for loop and plotted points.

Lastly, you will set the plot color back to black.

```
plt.color(255, 0, 0)
for I in range(20):
   plt.plot( cos(i)+bx, sin(i)+10, "o")
plt.color(0, 0, 0)
```

Fns > Modul > ti_plotlib > draw > color
Fns > Modul > ti_plotlib > draw > plot
Fns > Modul > math > trig > cos
Fns > Modul > math > trig > sin

20. Now, draw the player. For simplicity, your player needs a head, arm, body and leg. The head will be a circle, the arm, leg, and body will be line segments.



The code template will be:
    plt.line(x1, y1, x2, y2,””)

Remember, the player’s height is at (px, py)

What do you think the lines of code will look like? Fill in the templates below with values you think will work for the player.

```
#head
for i in range(20):
    plt.plot( cos(i), sin(i) + py, “o”)

#body
plt.line( _______, _______, _______, _______, “ “ )

#arm
plt.line( _______, _______, _______, _______, “ “ )

#leg
plt.line( _______, _______, _______, _______, “ “ )
```

Add these lines to the bottom of your code.

Teacher Tip:
Step 19 gives students one possible solution for the drawing.

21. Execute your code: [Trace]

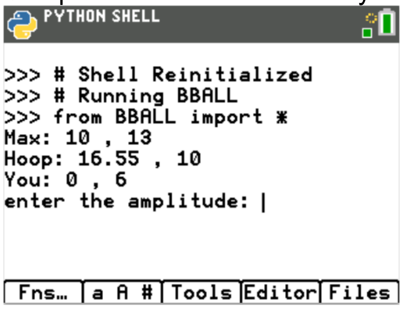
The code to the right shows a sample run.

Enter any value for the amplitude and press [enter]

Does your person and basketball hoop look similar to the one on the right?

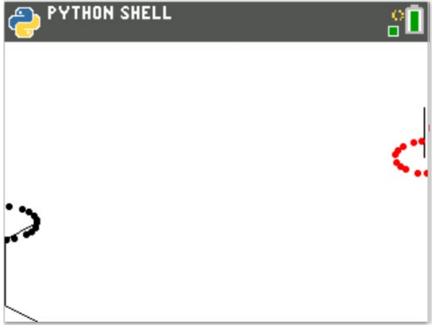
Run your code several more times. Each time the hoop and player should look a bit different due to the variabliity in our random variables.

Sample Run- Answers will vary



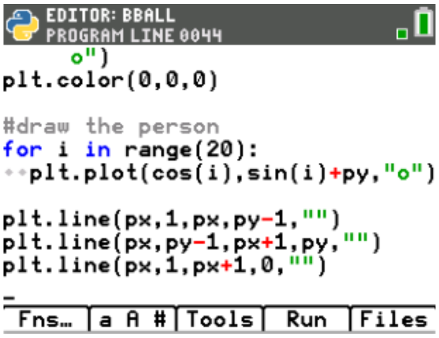
```
>>> # Shell Reinitialized
>>> # Running BBALL
>>> from BBALL import *
Max: 10 , 13
Hoop: 16.55 , 10
You: 0 , 6
enter the amplitude: |
```

Sample Run- pictures will vary



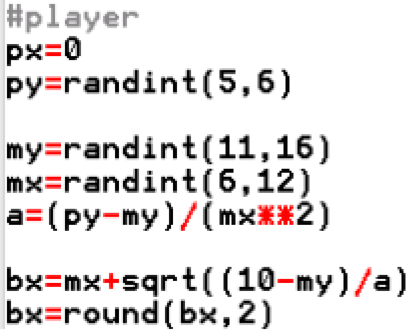
22. One possible solution for the person is shown below.

```
#draw person
for I in rang(20):
   plt.plot( cos(i), sin(i)+py, “o”)

plt.line(px, 1, px, py-1, “ “)
plt.line(px, py-1, px+1, py, “ “)
plt.line(px, 1, px+1, 0, “ “)
```



```
    o")
plt.color(0,0,0)

#draw the person
for i in range(20):
··plt.plot(cos(i),sin(i)+py,"o")

plt.line(px,1,px,py-1,"")
plt.line(px,py-1,px+1,py,"")
plt.line(px,1,px+1,0,"")
```

23. Now to plot the player’s shot.



```
#player
px=0
py=randint(5,6)

my=randint(11,16)
mx=randint(6,12)
a=(py-my)/(mx**2)

bx=mx+sqrt((10-my)/a)
bx=round(bx,2)
```

2. What does the variable ‘a’ represent?

3. What does the variable ‘bx’ represent?

Teacher Tip:

(my, mx) is the vertex for the parabola

a is the amplitude

bx is the distance from the player to the hoop

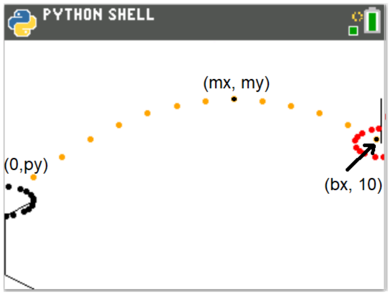
24. Recall the vertex form for a parabola is: $y = a(x – h)^2 + k$.

What variable did you use to represent h, the horizontal component for the vertex?

What variable did you use to represent k, the vertical component for the vertex?

Teacher Tip:
h is mx, the x portion of the vertex.
k is my, the y portion of the vertex.

25. To plot the path of the user's shot:



You will use the vertex form of a parabola: $y = a(x - h)^2 + k$

Substituting the the point (mx,my) as the vertex:

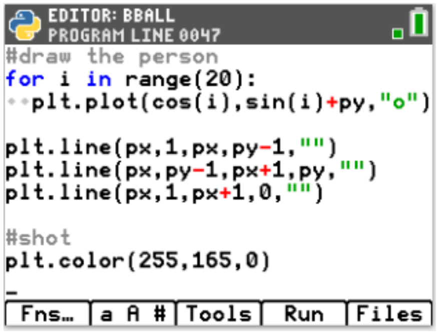
$$y = a(x - mx)^2 + my$$

About line 26 you request the variable pa, as the user's amplitude.

$$y = pa(x - mx)^2 + my$$

26. First, change the plot color to orange:

```
#shot
plt.color(255,165,0)
```


EDITOR: BBALL
PROGRAM LINE 0047

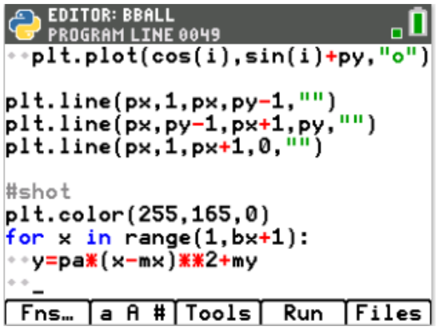
```
#draw the person
for i in range(20):
◆◆plt.plot(cos(i),sin(i)+py,"o")

plt.line(px,1,px,py-1,"")
plt.line(px,py-1,px+1,py,"")
plt.line(px,1,px+1,0,"")

#shot
plt.color(255,165,0)
_
```

27. Now to graph the parabolic shot.
Use a loop to cycle through x values starting at 1, ending at bx.

```
for x in range(1, bx+1):
    y=pa*(x - mx)**2 + my
```

Fns > ctl > for index in range(start,stop)

Make sure the line y = pa*(x-mx)**2 + my has two diamonds to indent the for loop. Python does not use the ^ for exponents. The symbols ** are used for exponents in python.

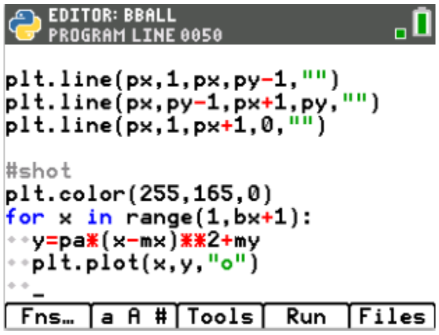

EDITOR: BBALL
PROGRAM LINE 0049

```
◆◆plt.plot(cos(i),sin(i)+py,"o")

plt.line(px,1,px,py-1,"")
plt.line(px,py-1,px+1,py,"")
plt.line(px,1,px+1,0,"")

#shot
plt.color(255,165,0)
for x in range(1,bx+1):
◆◆y=pa*(x-mx)**2+my
◆◆_
```

28. Plot points to represent the path of the ball. The points will use the x and y values from the for loop.

plt.plot(x, y, "o")

Make sure the plt.plot has two diamonds in front of the line.
This keeps the code part of the for loop.

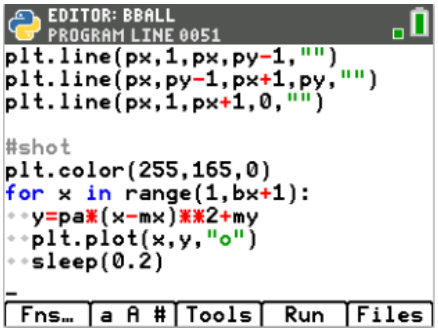

EDITOR: BBALL
PROGRAM LINE 0050

```

plt.line(px,1,px,py-1,"")
plt.line(px,py-1,px+1,py,"")
plt.line(px,1,px+1,0,"")

#shot
plt.color(255,165,0)
for x in range(1,bx+1):
◆◆y=pa*(x-mx)**2+my
◆◆plt.plot(x,y,"o")
◆◆_
```

29. To animate your shot add the line

    sleep(0.2)

    Fns > Modul > Time > sleep

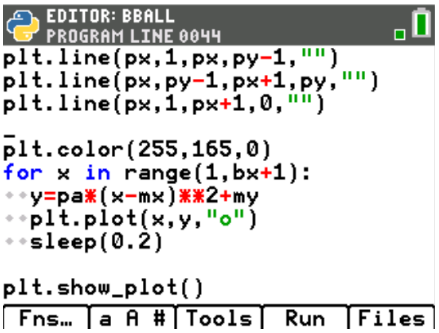


```
plt.line(px,1,px,py-1,"")
plt.line(px,py-1,px+1,py,"")
plt.line(px,1,px+1,0,"")

#shot
plt.color(255,165,0)
for x in range(1,bx+1):
  y=pa*(x-mx)**2+my
  plt.plot(x,y,"o")
  sleep(0.2)
```

30. Add the line plt.show().

    This line will keep the plot on the screen until the user presses the [clear] button.

    Fns > Modul > ti_plotlib > show



```
plt.line(px,1,px,py-1,"")
plt.line(px,py-1,px+1,py,"")
plt.line(px,1,px+1,0,"")

plt.color(255,165,0)
for x in range(1,bx+1):
  y=pa*(x-mx)**2+my
  plt.plot(x,y,"o")
  sleep(0.2)

plt.show_plot()
```

31. Execute your program. [Trace]

    Here is the sample to the right worked out.

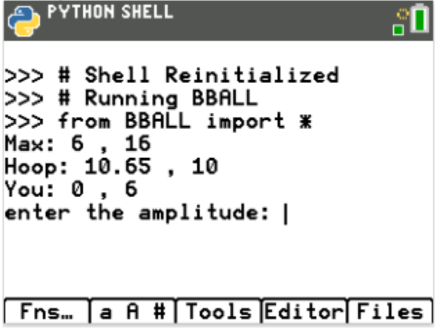
    $y = a(x – h)^2 + k$

    $y = a(x – mx)^2 + my$ #max height (mx, my)

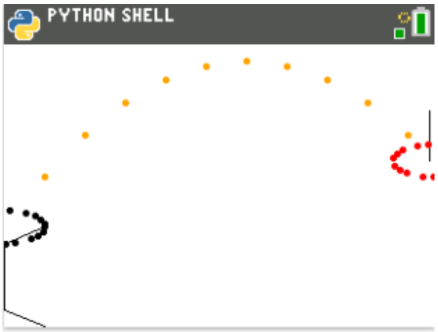
    $y = a(x – 6)^2 + 16$

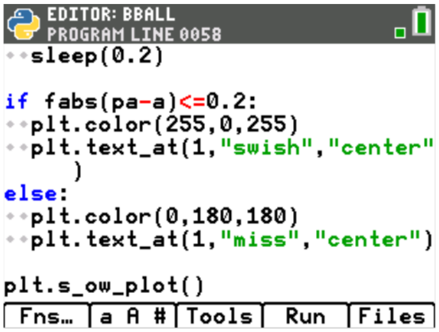
    a = -6/(4.65)**2

Sample Random Problem:



```
>>> # Shell Reinitialized
>>> # Running BBALL
>>> from BBALL import *
Max: 6 , 16
Hoop: 10.65 , 10
You: 0 , 6
enter the amplitude: |
```





```
◆◆sleep(0.2)

if fabs(pa-a)<=0.2:
◆◆plt.color(255,0,255)
◆◆plt.text_at(1,"swish","center"
    )
else:
◆◆plt.color(0,180,180)
◆◆plt.text_at(1,"miss","center")

plt.s_ow_plot()
```

32. Lastly, add code to check the accuracy.
    This code will go right before the last line plt.show_plot()

    If the difference between the user’s a value at x = bx and the real a value is within a 0.2 print “swish” otherwise print “miss”.

    Mathematically, that would be | pa – a | ≤ 1.
    Using python, that is fabs(pa – a) <= 1.

```
if fabs(pa – a) <= 0.2:
        plt.color(255,0,255)
        plt.text_at(1, “swish”,”center”)
else:
        plt.color(0,180,180)
        plt.text(1, “miss”,”center”)
```

    Make sure the plt.color and plt.text_at have two diamonds for indentation.

    Fns > Ctl > if..else
    Fns > Modul > TI plot_lib > Draw > color
    Fns > Modul > TI plot_lib > Drawe > draw_text

33. How many shots can you make in a row before you miss?

Teacher Notes:

```
# Random Simulation
from random import *
from math import *
from time import *
import ti_plotlib as plt

#basket center (bx,by)
bx=0
by=10

#player
px=0
py=randint(5,6)

my=randint(11,16)
mx=randint(6,12)
a=(py-my)/(mx**2)

bx=mx+sqrt((10-my)/a)
bx=round(bx,2)
```

```
#data display
print("Max:",mx,",",my)
print("Hoop:",bx,",",10)
print("You:",0,",",py)

pa=float(eval(input("enter the amplitude: ")))

plt.window(0,bx,0,my+randint(1,5))
plt.cls()

#hoop
plt.line(bx-0.1,10,bx-0.1,13,"")
plt.color(255,0,0)
for i in range(20):
    plt.plot(cos(i)+bx,sin(i)+10,"o")
plt.color(0,0,0)

#draw the person
for i in range(20):
    plt.plot(cos(i),sin(i)+py,"o")
plt.line(px,1,px,py-1,"")
plt.line(px,py-1,px+1,py,"")
plt.line(px,1,px+1,0,"")

#shot
plt.color(255,165,0)
for x in range(1,bx+1):
    y=pa*(x-mx)**2+my
    plt.plot(x,y,"o")
    sleep(0.2)

if fabs(pa - a) <= 0.2:
    plt.color(255,0,255)
    plt.text_at(1,"swish","center")
else:
    plt.color(0,180,180)
    plt.text_at(1,"miss","center")

plt.show_plot()
```