

Factor Darts: Solving Quadratic Equations

In this project, you will create a dart game that uses factoring to earn darts. The game will generate 4 random factor problems. If you answer correctly, you will earn a dart. Answer two questions in a row, earn an additional dart. After factoring 4 questions, toss your darts to earn a score. How high can you score? Answer all 4 questions correctly, earn 6 darts. Can you score a perfect 30?

Objectives:

Programming Objectives:

- Use the randint() function to generate random integers.
- Use for loops to repeat code.
- Use selection statements to make decisions.
- Use the draw functions from the TI-Draw library to create circles and text.
- Use the sleep function from the Time library to create animation.

Math Objectives:

- Practice solving quadratic equations
- Use the distance formula to find the distance between two points
- Use the modulus operator, %, to find a remainder

You will create a dart game that generates four random quadratic expressions of the form: $x^2 + bx + c$ or $ax^2 + bx + c$. For each question the player answers correctly, the player will get another shot at the dart board. At the end of the game, the dartboard shows the darts on the board as well as the score the round.

Example 1:

Display Question 1

```
Python Shell 5/5
>>>#Running factoring_darts.py
>>>from factoring_darts import *
Factor: x^2-2x-24
(x+a)(x+b)
a:
```

Answer 1 correct, display Q2

```
Python Shell 10/10
>>>#Running factoring_darts.py
>>>from factoring_darts import *
Factor: x^2-2x-24
(x+a)(x+b)
a: -6
b: 4
>>>
Factor: 4x^2+2x-2
(ax+b)(x+c)
a:|
```

Answer 2 incorrect, display Q3

```
Python Shell 17/17
>>>
Factor: 4x^2+2x-2
(ax+b)(x+c)
a: 2
b: 2
c: 2
sorry a = 4 b= -2 c= 1
>>>
Factor: -4x^2+29x-45
(ax+b)(x+c)
a:|
```

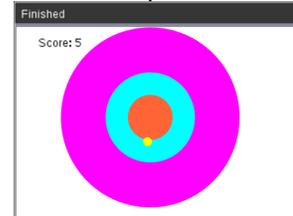
Answer 3 is incorrect,

```
Python Shell 24/24
>>>
Factor: -4x^2+29x-45
(ax+b)(x+c)
a: -4
b: 5
c: -9
sorry a = -4 b= 9 c= -5
>>>
Factor: 5x^2-23x+24
(ax+b)(x+c)
a:
```

Last answer is incorrect.

```
Python Shell 30/30
sorry a = -4 b= 9 c= -5
>>>
Factor: 5x^2-23x+24
(ax+b)(x+c)
a: 5
b: -3
c: -8
sorry a = 5 b= -8 c= -3
>>>
Correct: 1
>>>
```

One correct questions = 1 darts.



Example2:

```
Python Shell 6/6
>>>#Running factoring_darts.py
>>>from factoring_darts import *
Factor: x^2+2x-8
(x+a)(x+b)
a: 4
b: -2|
```

Correct: 1 dart

```
Python Shell 10/10
>>>#Running factoring_darts.py
>>>from factoring_darts import *
Factor: x^2+2x-8
(x+a)(x+b)
a: 4
b: -2
>>>
Factor: 3x^2+10x-25
(ax+b)(x+c)
a:|
```

Correct: 2 total darts

```
Python Shell 16/16
b: -2
>>>
Factor: 3x^2+10x-25
(ax+b)(x+c)
a: 3
b: -5
c: 5
>>>
Factor: 3x^2+16x+21
(ax+b)(x+c)
a:|
```

Correct: 3 total darts

```
Python Shell 22/22
c: 5
>>>
Factor: 3x^2+16x+21
(ax+b)(x+c)
a: 3
b: 7
c: 3
>>>
Factor: -3x^2-14x-8
(ax+b)(x+c)
a: |
```

Incorrect

```
Python Shell 28/28
c: 3
>>>
Factor: -3x^2-14x-8
(ax+b)(x+c)
a: -3
b: -4
c: 2
sorry a = -3 b= -2 c= 4
>>>
Correct: 3
>>>|
```

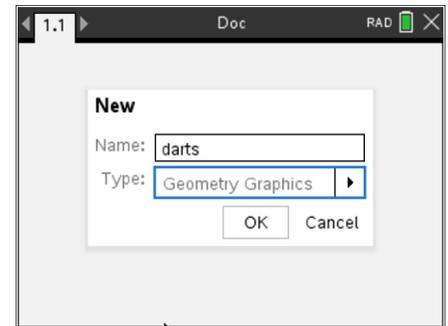
Three correct questions = 3 darts



1. The first step will be to create a Python geometry document.

Create a new python project named “darts”.

Select “Geometry Graphics” from the type menu.



2. You will need four more libraries: random, math, ti_system and time

The random library contains the randint() function.

The math library contains the square root function sqrt().

The time library includes the sleep() function .

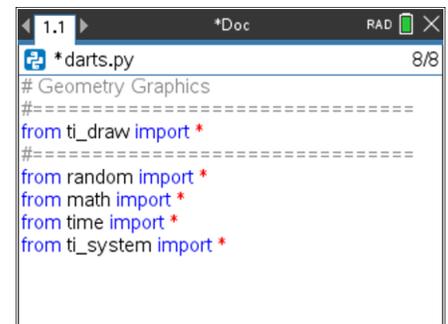
Place your cursor on the line below the **import ti_draw**

Menu> Random> from random import *

Menu> Math> from math import *

Menu > More Modules > Time > from time import *

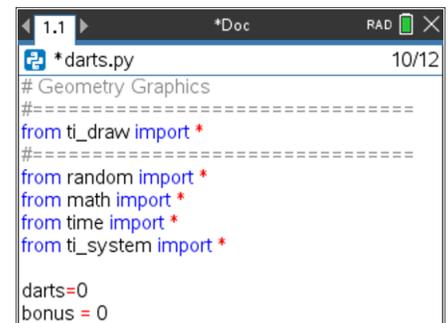
Menu > More Modules > TI System > from ti_system imoport *



3. The user will earn one dart for each correct answer. The user will earn a bonus dart for every two correct questions answered in a row. Create two variables, one to hold the number of darts, the other to keep track of the questions answered in a row.

darts = 0

bonus = 0

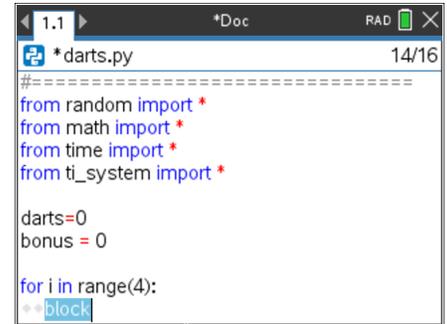


4. Initially, you will set the game to play for 4 rounds. Once the entire game is coded, you can easily change this to as many rounds as you like.

Add a for loop that will repeat 4 times.

Menu > Built-ins Control > for index in range(size)

for i in range(4):



```

1.1 *darts.py 14/16
=====
from random import *
from math import *
from time import *
from ti_system import *

darts=0
bonus = 0

for i in range(4):
    block
    
```

5. Each question will either be an “easy” or “difficult” question. If you want to set the probability to 50% each question will be “easy”, You need two choices. For that, use randint(0,1). If the number is a 0, generate an easy question, otherwise generate a “difficult” question. If you want more easy questions to appear, change the randint(0,1) to something like randint(0,3). That will generate numbers 0-3. Now say if randint(0,3) <3 generate an “easy” question. That will make each question have a 75% probability of being an “easy” question.

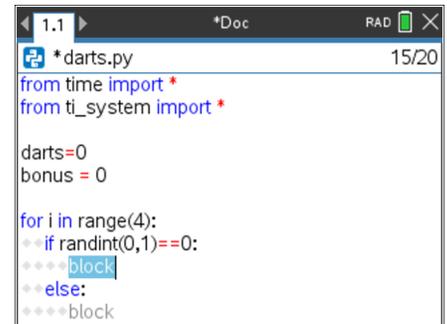
Add an if..else statement with randint values of your choice.

if randint(0,1) == 0:

else:

Menu > Built-ins > Control > if..else

Menu > Random > randint



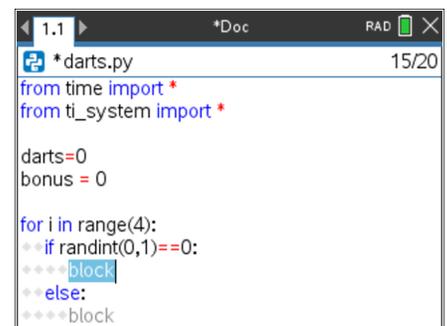
```

1.1 *darts.py 15/20
from time import *
from ti_system import *

darts=0
bonus = 0

for i in range(4):
    if randint(0,1)==0:
        block
    else:
        block
    
```

6. Compare your code to the one on the right. Make sure your indentation is correct. Python uses indentation to keep loops and selection statements together. The lines under the for statement have one level of indentation (two diamond spaces). Each line that is part of the if statement is also indented one level (two diamond spaces).



```

1.1 *darts.py 15/20
from time import *
from ti_system import *

darts=0
bonus = 0

for i in range(4):
    if randint(0,1)==0:
        block
    else:
        block
    
```

7. In the first if statement, you will generate the “easy” questions in the form $x^2 + bx + c$. These questions will be factored in the form $(x + n1)(x + n2)$.

Why? You might ask.

$$(x + n1)(x + n2)$$

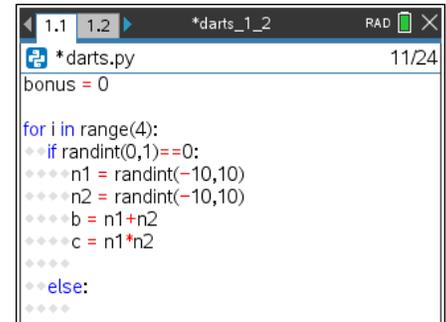

$$x^2 + n2*x + n1*x + n1*n2$$

$$x^2 + (n1+n2)x + n1*n$$

The sum of n1 and n2 create the b value. The product of n1 and n2 create the c value.

8. Inside the if statement. Randomly generate n1 and n2
Use these numbers to create b and c.

```
n1 = randint(-10,10)
n2 = randint(-10,10)
b = n1+n2
c = n1*n2
```



```
*darts_1_2
RAD 11/24
*darts.py
bonus = 0

for i in range(4):
    if randint(0,1)==0:
        n1 = randint(-10,10)
        n2 = randint(-10,10)
        b = n1+n2
        c = n1*n2
    else:
```

9. To display the function correctly is a bit tricky. For example, if the sum is -3 and the product is 10, you want the display to be $x^2-3x+10$. If we coded the project with `disp = "x^2" + b + "x" + c`, it would display correctly for this problem.

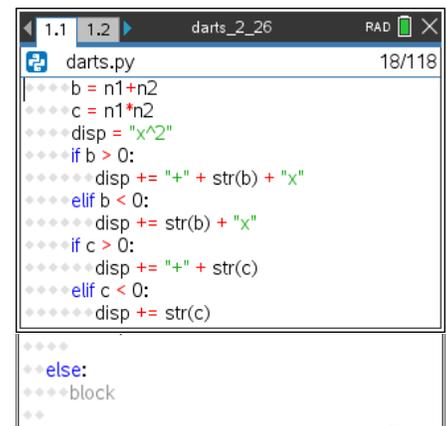
However, if the values were $b = 3$ and $c = -10$, the display code above would display: $x^23x+-10$.

Therefore, you will need an if statement.

- all displays start with x^2
- if the sum is positive,
 - add "+" and b and "x"
- otherwise if the sum is negative,
 - add only b and "x" (b by default will add a negative)
- if product is positive,
 - add "+" and c
- otherwise if the sum is negative,
 - add only c.

Add the lines:

```
disp = "x^2"
if b > 0:
    disp += "+" + str(b) + "x"
elif b < 0:
    disp += str(b) + "x"
if c > 0:
```



```
darts_2_26
RAD 18/118
darts.py
b = n1+n2
c = n1*n2
disp = "x^2"
if b > 0:
    disp += "+" + str(b) + "x"
elif b < 0:
    disp += str(b) + "x"
if c > 0:
    disp += "+" + str(c)
elif c < 0:
    disp += str(c)
else:
    block
```

```

disp += "+" + str(c)
elif c < 0:
    disp += str(c)
    
```

10. Now to display the question and the form of the input.
 print(disp) will print the equation on the first line.
 print("(x+n1)(x+n2)") will tell the user the form of question input.

Add the lines:

```

print(disp)
print("(x+n1)(x+n2)")
    
```

Menu > Built-ins > I/O > print

11. Ask the user for the values, n1 and n2.
 Store these values in variables named u1 and u2.

```

u1 = int(input("enter n1 "))
u2 = int(input("enter n2 "))
    
```

Menu > Built-ins > Type > int

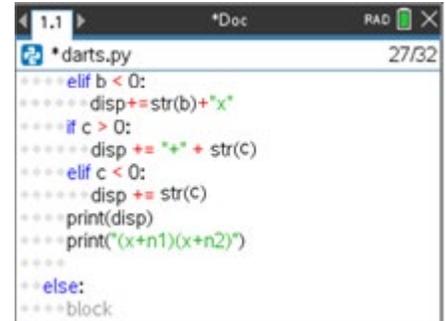
Menu > Built-ins > I/O > input

**By default, input returns a string value. String values are treated like characters not numbers. Place int() around the input casts the input as an integer.

12. If the input is correct, add one to the number of darts and the bonus variable.
 Otherwise, print the correct result and set bonus back to 0.

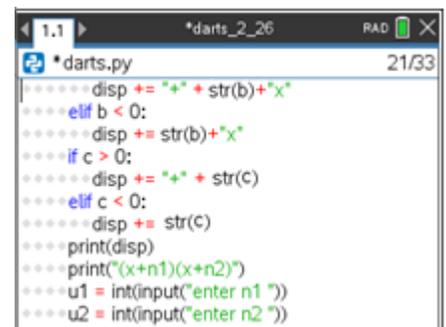
```

if u1+u2==b and u1*u2==c:
    darts+=1
    bonus+=1
else:
    print("n1 =", n1, " n2 =", n2)
    bonus=0
    
```



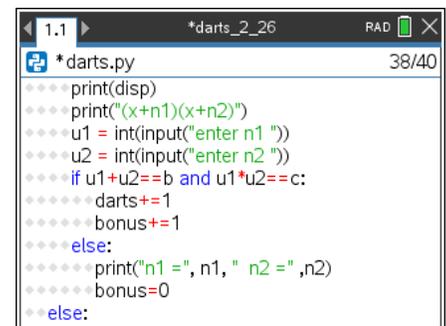
```

1.1 *darts.py 27/32
elif b < 0:
    disp += str(b) + "x"
if c > 0:
    disp += "+" + str(c)
elif c < 0:
    disp += str(c)
print(disp)
print("(x+n1)(x+n2)")
else:
    block
    
```



```

1.1 *darts_2_26 21/33
disp += "+" + str(b) + "x"
elif b < 0:
    disp += str(b) + "x"
if c > 0:
    disp += "+" + str(c)
elif c < 0:
    disp += str(c)
print(disp)
print("(x+n1)(x+n2)")
u1 = int(input("enter n1 "))
u2 = int(input("enter n2 "))
    
```

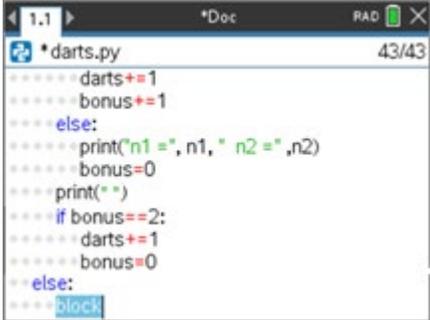


```

1.1 *darts_2_26 38/40
print(disp)
print("(x+n1)(x+n2)")
u1 = int(input("enter n1 "))
u2 = int(input("enter n2 "))
if u1+u2==b and u1*u2==c:
    darts+=1
    bonus+=1
else:
    print("n1 =", n1, " n2 =", n2)
    bonus=0
else:
    
```

13. Unindent from the if statement. (Make sure you have 4 diamonds).
Print a blank line. If the user has answered two questions in a row correctly (bonus=2), then add a free dart and set the bonus back to 0.

```
print(" ")
if bonus==2:
    darts+=1
    bonus=0
```



```
*darts.py 43/43
..... darts+=1
..... bonus+=1
..... else:
..... print("n1 =", n1, " n2 =", n2)
..... bonus=0
..... print(" ")
..... if bonus==2:
.....     darts+=1
.....     bonus=0
..... else:
.....     block
```

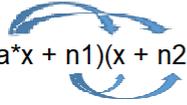
14. Now to repeat the same process for the “difficult” questions.
You will:
- generate the values
 - display the question
 - get the user’s input
 - check the user’s input
 - add darts if need be, otherwise display the correct values.

15. The second quadratic form will factor into $(ax + n1)(x + n2)$.

How would you calculate b and c in this form?

Use distribution to factor $(ax + n1)(x + n2)$

16. Did you get the following?

$$(a*x + n1)(x + n2)$$


$$ax^2 + a*n2*x + n1*x + n1*n2$$

$$ax^2 + (a*n2+n1)x + n1*n2$$

$$b = a*n2 + n1$$

$$c = n1*n2$$

17. Add 6 lines of code.

a.) Generate a number from 0 to 1.

If the number is a 0, variable a should be between 2 and 5
otherwise a is a number between -5 and -2.

b.) Generate n1 to be a random number between -10 and 10

c.) Generate n2 to be a random number between -10 and 10.

18. Add the two lines to calculate b and c

$$b = a*n2 + n1$$

$$c = n1*n2$$

19. Check the last 8 lines of code. Did you do the following?

```
if randint(0,1)==0:
```

```
    a=randint(2,5)
```

```
else:
```

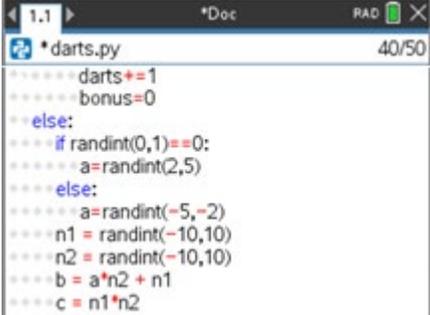
```
    a=randint(-5,-2)
```

```
n1 = randint(-10,10)
```

```
n2 = randint(-10,10)
```

```
b = a*n2 + n1
```

```
c = n1*n2
```



```
1.1 *darts.py 40/50
darts += 1
bonus = 0
else:
if randint(0,1)==0:
a = randint(2,5)
else:
a = randint(-5,-2)
n1 = randint(-10,10)
n2 = randint(-10,10)
b = a*n2 + n1
c = n1*n2
```

20. The display code for the “difficult” form is the exact same as the display code for the “easy” form. The only difference is the first line of code. Add the line:

```
disp = str(a) + "x^2"
```

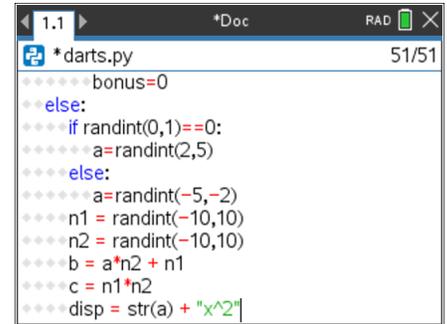
21. You could retype the 8 lines it takes to finish the display or you can copy and paste those line. Scroll back up to the lines that start with
if b > 0

Put the cursor in front of the “i” of the if. Press the [shift] key and arrow down to the line “disp += c”. This should highlight the 8 lines of code in blue.

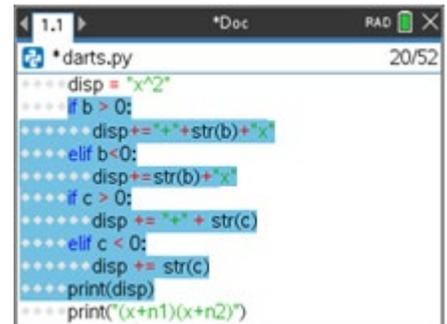
Press [ctrl] [c] to copy the lines. Scroll to the bottom of your code and press [ctrl] [v] to paste the lines below disp = str(a)+ “x^2”

Make sure your code matches the code to the right.
Make sure the indentation matches.

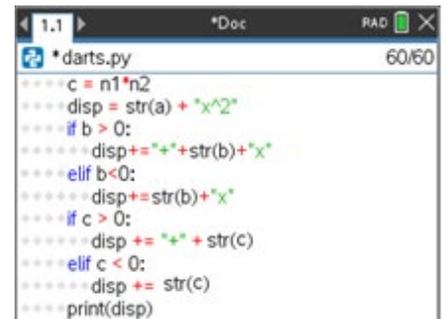
22. Add the display line:
print(“(ax+n1)(x+n2)”)



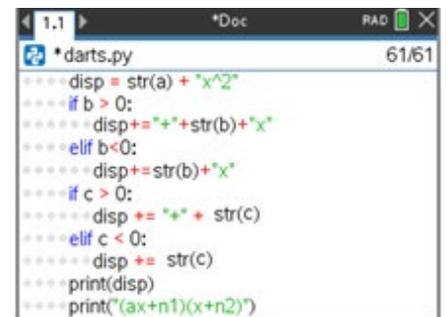
```
1.1 *darts.py 51/51
bonus=0
else:
if randint(0,1)==0:
a=randint(2,5)
else:
a=randint(-5,-2)
n1 = randint(-10,10)
n2 = randint(-10,10)
b = a*n2 + n1
c = n1*n2
disp = str(a) + "x^2"
```



```
1.1 *darts.py 20/52
disp = "x^2"
if b > 0:
disp += "+" + str(b) + "x"
elif b < 0:
disp += str(b) + "x"
if c > 0:
disp += "+" + str(c)
elif c < 0:
disp += str(c)
print(disp)
print("(x+n1)(x+n2)")
```



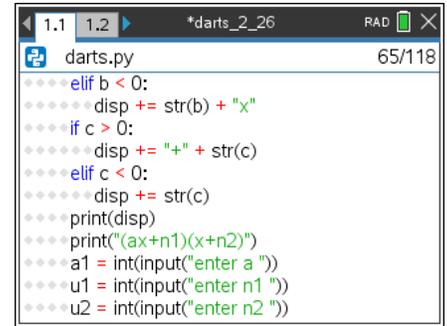
```
1.1 *darts.py 60/60
c = n1*n2
disp = str(a) + "x^2"
if b > 0:
disp += "+" + str(b) + "x"
elif b < 0:
disp += str(b) + "x"
if c > 0:
disp += "+" + str(c)
elif c < 0:
disp += str(c)
print(disp)
```



```
1.1 *darts.py 61/61
disp = str(a) + "x^2"
if b > 0:
disp += "+" + str(b) + "x"
elif b < 0:
disp += str(b) + "x"
if c > 0:
disp += "+" + str(c)
elif c < 0:
disp += str(c)
print(disp)
print("(ax+n1)(x+n2)")
```

23. Ask the user for a, n1, and n2. Store the values as a1, u1 and u2.

```
a1 = int(input("enter a "))
u1 = int(input("enter n1 "))
u2 = int(input("enter n2 "))
```

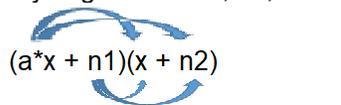


```
darts.py 65/118
elif b < 0:
    disp += str(b) + "x"
if c > 0:
    disp += "+" + str(c)
elif c < 0:
    disp += str(c)
print(disp)
print("(ax+n1)(x+n2)")
a1 = int(input("enter a "))
u1 = int(input("enter n1 "))
u2 = int(input("enter n2 "))
```

24. Check the user's input against the correct values.

This code will be very similar to the code you wrote for the "easy" form.

Remember you generated a, n1, n2 then stored them as b and c.

$$(a*x + n1)(x + n2)$$


$$ax^2 + a*n2*x + n1*x + n1*n2$$

$$ax^2 + (a*n2+n1)x + n1*n2$$

You asked the user for a1, u1, u2.

Fill in the missing code statements below:

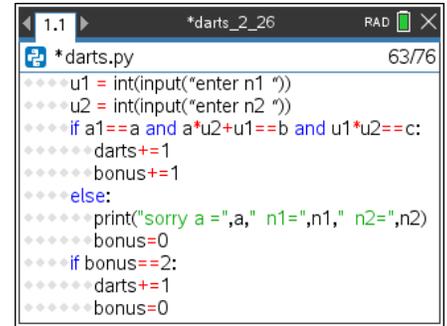
```
if _____ == a and _____ == b and _____ == c:
    dart _____
    bonus _____
else:
    print(_____ )
    bonus _____

if bonus == ____:
    dart _____
    bonus _____
```

25. Does your code match the code below?

```

if a1==a and a*u2+u1==b and u1*u2==c:
    darts+=1
    bonus+=1
else:
    print("sorry a =", a, " n1=", n1," n2=", n2)
    bonus=0
if bonus==2:
    darts+=1
    bonus=0
    
```

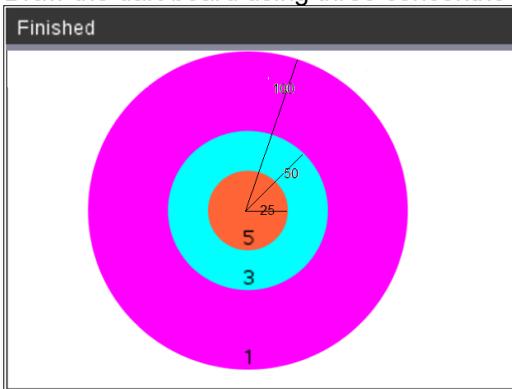


```

1.1 *darts_2_26 RAD 63/76
*darts.py
u1 = int(input("enter n1 "))
u2 = int(input("enter n2 "))
if a1==a and a*u2+u1==b and u1*u2==c:
    darts+=1
    bonus+=1
else:
    print("sorry a =",a," n1=",n1," n2=",n2)
    bonus=0
if bonus==2:
    darts+=1
    bonus=0
    
```

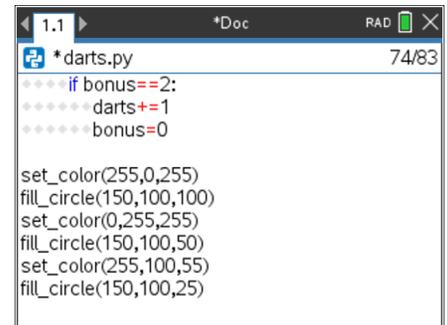
26. The dartboard will be drawn after all the questions have been answered. Therefore, unindent.

Draw the dart board using three concentric circles.



To draw a circle:
`fill_circle(x, y, radius)`
where x and y are the center

Change the color:
`set_color(red, green, blue)`
numbers for red, green, blue must be integers 0-255



```

1.1 *Doc RAD 74/83
*darts.py
if bonus==2:
    darts+=1
    bonus=0

set_color(255,0,255)
fill_circle(150,100,100)
set_color(0,255,255)
fill_circle(150,100,50)
set_color(255,100,55)
fill_circle(150,100,25)
    
```

```

set_color(255,0,255)
fill_circle(150,100,100)
set_color(0,255,255)
fill_circle(150,100,50)
set_color(255,100,55)
fill_circle(150,100,25)
    
```

Menu > More Modules > TI Draw > Control > `set_color`
Menu > More Modules > TI Draw > Shape > `fill_circle`

27. Add the point labels. A bullseye will result in 5 points. The middle circle will count for 3 points. The outer circle will count for 1 point.

```
set_color(0,0,0)
draw_text(148,200,"1")
draw_text(148,150,"3")
draw_text(148,125,"5")
```

28. Create a score variable that starts at 0.

29. You will code the project to let the user press “t” to toss a dart. To let the user know this, draw “Toss Dart (t)” to the screen.

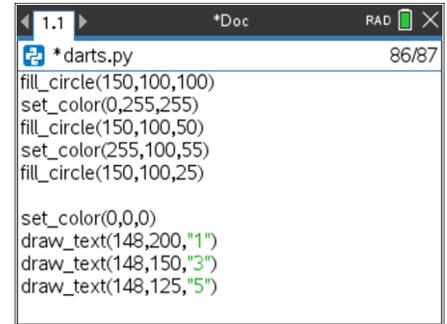
```
draw_text(225, 25, "Toss Dart (t)")
```

Menu > More Modules > TI Draw > Shape > draw_text

30. Now to toss the darts.
Create a loop that will repeat itself for each dart.

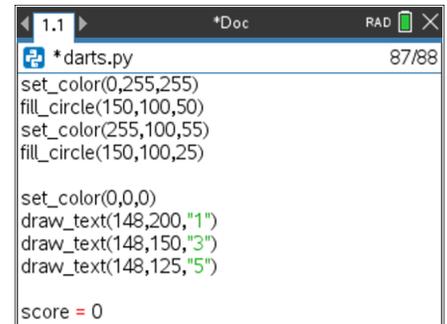
```
for i in range(darts):
```

Factor Darts STUDENT DOCUMENT



```
1.1 *darts.py 86/87
fill_circle(150,100,100)
set_color(0,255,255)
fill_circle(150,100,50)
set_color(255,100,55)
fill_circle(150,100,25)

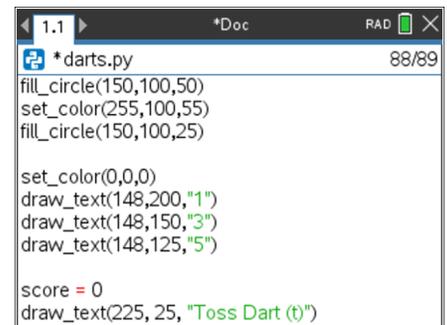
set_color(0,0,0)
draw_text(148,200,"1")
draw_text(148,150,"3")
draw_text(148,125,"5")
```



```
1.1 *darts.py 87/88
set_color(0,255,255)
fill_circle(150,100,50)
set_color(255,100,55)
fill_circle(150,100,25)

set_color(0,0,0)
draw_text(148,200,"1")
draw_text(148,150,"3")
draw_text(148,125,"5")

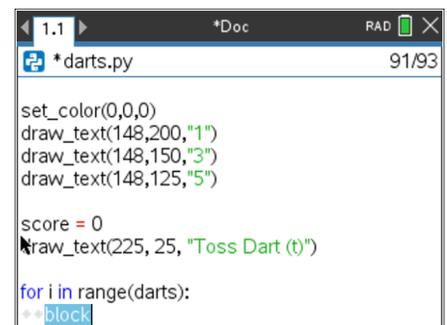
score = 0
```



```
1.1 *darts.py 88/89
fill_circle(150,100,50)
set_color(255,100,55)
fill_circle(150,100,25)

set_color(0,0,0)
draw_text(148,200,"1")
draw_text(148,150,"3")
draw_text(148,125,"5")

score = 0
draw_text(225, 25, "Toss Dart (t)")
```



```
1.1 *darts.py 91/93
set_color(0,0,0)
draw_text(148,200,"1")
draw_text(148,150,"3")
draw_text(148,125,"5")

score = 0
draw_text(225, 25, "Toss Dart (t)")

for i in range(darts):
    block
```

31. For each toss of a dart, you will erase the old dart count and draw a new one. To “erase” the old count, you’ll draw a white rectangle over the old text, then draw new text. Add the lines:

```
set_color(255,255,255)
fill_rect(0,0,88,23)
set_color(0,0,0)
draw_text(0,25,"darts left " + str(darts-i))
```

32. We don’t want to throw the dart until the user presses [t].

The command `get_key()` gets the value of the next key press.

If you put this in the conditional of a while loop, you can create a pause until the user presses [t]. Add the lines:

```
while get_key() != "t":
    continue
```

Menu > Built-ins > Control > while

Menu > More Modules > TI System > `get_key()`

Menu > Built-ins > Control > continue

33. For some variation, we will say if the shot was on an “even” dart throw, generate x: [50,250] and y: [0,200]. If the shot is “odd”, then generate x: [100,200] and y: [50, 150].

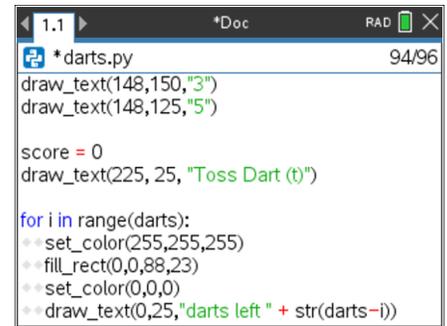
The modulus symbol `%` is used to find the remainder in division.

Even dart numbers such as 8, 6 and 4 all return 0 if you type `darts%2`.

Odd dart numbers such as 9, 7 and 5 all return 1 if you type `darts%`.

Add the following lines:

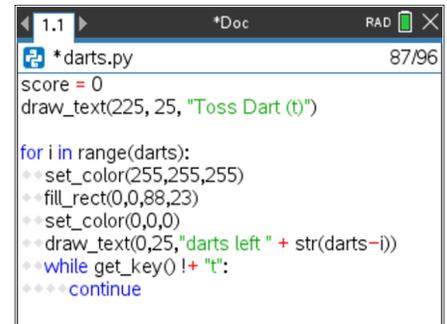
```
if darts%2==0:
    x=randint(50,250)
    y=randint(0,200)
else:
    x=randint(100,200)
    y=randint(50,150)
```



```
1.1 *Doc RAD 94/96
*darts.py
draw_text(148,150,"3")
draw_text(148,125,"5")

score = 0
draw_text(225, 25, "Toss Dart (t)")

for i in range(darts):
    *set_color(255,255,255)
    *fill_rect(0,0,88,23)
    *set_color(0,0,0)
    *draw_text(0,25,"darts left " + str(darts-i))
```

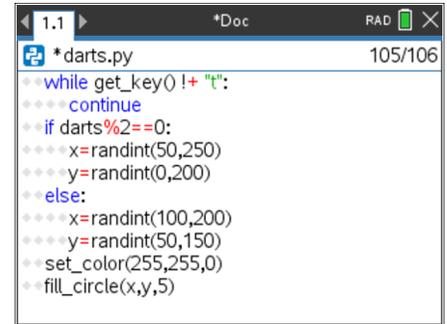


```
1.1 *Doc RAD 87/96
*darts.py
score = 0
draw_text(225, 25, "Toss Dart (t)")

for i in range(darts):
    *set_color(255,255,255)
    *fill_rect(0,0,88,23)
    *set_color(0,0,0)
    *draw_text(0,25,"darts left " + str(darts-i))
    *while get_key() != "t":
        *continue
```

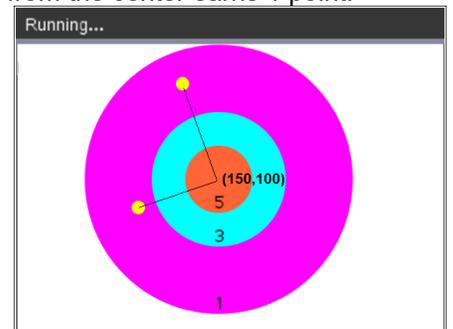
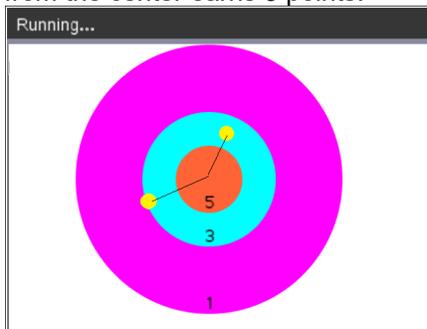
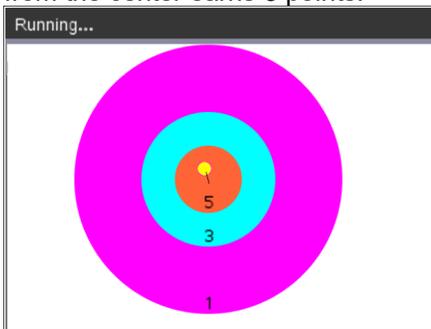
34. Set the dart color to yellow, or any other color of your choice.
Draw the circular dart.

```
set_color(255,255,0)
fill_circle(x,y,5)
```



```
*darts.py 105/106
while get_key() != "t":
    continue
if darts%2==0:
    x=randint(50,250)
    y=randint(0,200)
else:
    x=randint(100,200)
    y=randint(50,150)
set_color(255,255,0)
fill_circle(x,y,5)
```

35. Points are scored based on the distance the dart lands from the bullseye.
Any dart that lands within 27.5 units from the center earns 5 points.
Any dart that lands within 52.5 units from the center earns 3 points.
Any dart that lands within 102.5 units from the center earns 1 point.

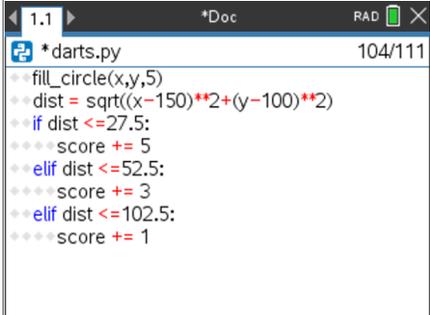


If the dart is located at (x,y) and the center is at $(150,100)$, what is the equation that represents the distance between the two points?

36. Use the equation from above to fill in the blanks below. Then code the lines on your project. Remember to use $**2$ instead of 2 when coding the distance.

```
dist=_____
if dist <= 27.5:
    score += 5
elif dist _____:
    score+=3
elif dist<=102.5:
    score+=1
```

37. Does your code match the code to the right?



```

1.1 *darts.py 104/111
fill_circle(x,y,5)
dist = sqrt((x-150)**2+(y-100)**2)
if dist <= 27.5:
    score += 5
elif dist <= 52.5:
    score += 3
elif dist <= 102.5:
    score += 1
    
```

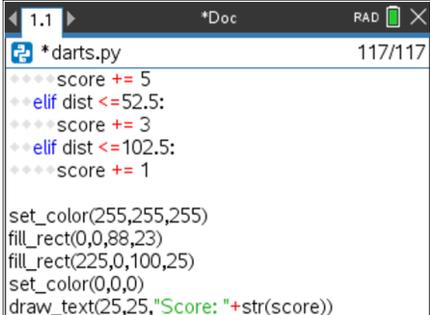
38. Only 5 more lines of code.

Draw white rectangles over the dart count down and the “t” hint.
Draw the score to the board.

Add the lines:

```

set_color(255,255,255)
fill_rect(0,0,88,23)
fill_rect(225,100,25)
set_color(0,0,0)
draw_text(25,25,"score:"+str(score))
    
```



```

1.1 *darts.py 117/117
score += 5
elif dist <= 52.5:
    score += 3
elif dist <= 102.5:
    score += 1

set_color(255,255,255)
fill_rect(0,0,88,23)
fill_rect(225,100,25)
set_color(0,0,0)
draw_text(25,25,"Score: "+str(score))
    
```

39. Enjoy playing Factor Darts. How many can you answer correctly? Can you answer all 4 correctly, earning 6 darts? Can you earn a perfect 30?