

Pen Up & Fill Image

Textbook

Pen Up & Fill Image



Pen Up & Pen Down

As you are drawing with your turtle, there might be times where you don't want the turtle to draw, but you still want to move the turtle. You can move your turtle without drawing by using the `.penup()` command.

You can then put the pen back down with the `.pendown()` command.

Imagine that the turtle is dragging a pen behind it. When the pen is down, it draws. When it lifts the pen up, it stops drawing.

```
1 turtle.forward(100)
2 turtle.right(90)
3 turtle.penup()
4 turtle.forward(100)
5 turtle.left(90)
6 turtle.pendown()
```

```
7 turtle.forward(100)
```

This will show the turtle moving without drawing.

Fill Image

You can also fill a shape once it's enclosed. This is done with the commands.

1. `.begin_fill()`
2. `.end_fill()`

Add the begin fill command before drawing the shape. Add the end fill command once the shape is complete.

```
1 turtle.begin_fill()
2 turtle.forward(100)
3 turtle.left(90)
4 turtle.forward(100)
5 turtle.left(90)
6 turtle.forward(100)
7 turtle.left(90)
8 turtle.forward(100)
9 turtle.end_fill()
```

This will draw a square and then fill it at the end.

Determine Fill Color

To change the fill color, you need to tell the shape what color you want before the begin fill command. Choose your color with the `turtle.fillcolor()` command. Put your color inside the parentheses.

```
1 turtle.fillcolor("red")
2 turtle.begin_fill()
3 turtle.forward(100)
4 turtle.left(90)
5 turtle.forward(100)
6 turtle.left(90)
7 turtle.forward(100)
8 turtle.left(90)
9 turtle.forward(100)
10 turtle.end_fill()
```

This will fill the square with red. You can also use hexadecimal values for your colors as well.

Expressing Solutions as Computational Steps

When you're working with computers, whether you're writing a program, designing a game, or even creating a complex spreadsheet, you don't just tell the computer the answer. Instead, you need to express solutions as computational steps. This means breaking down how to solve a problem into a clear, precise, step-by-step set of instructions that the computer can follow. You'll learn to solve problems one small piece at a time, rather than trying to do everything all at once.

To do this effectively, you might represent your solutions in multiple ways, perhaps by writing out the steps in plain English (like a recipe), drawing a flowchart, or eventually writing code. This helps you understand the problem from different angles. As you work, you'll also learn to use patterns and structures to see how different parts of a problem connect, making it easier to build your solution.

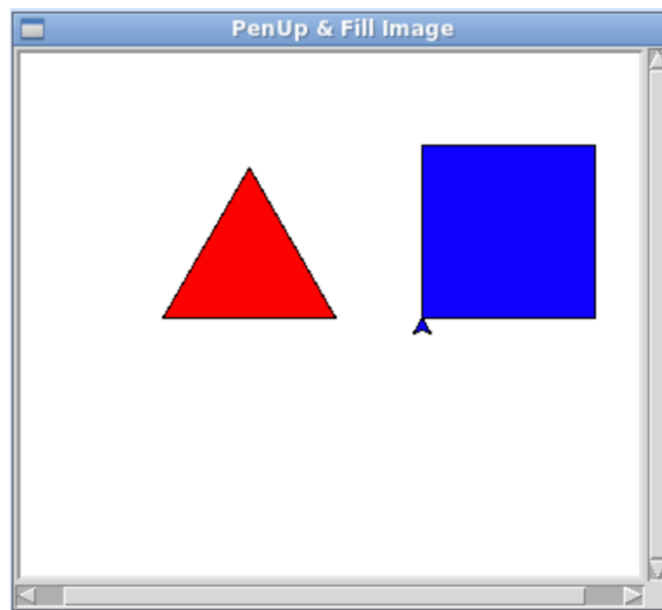
Most importantly, it's crucial to check your computations at every stage and continually ask yourself, "Does this solution make sense?" Regularly verifying your work helps you catch errors early and ensures your final solution is correct and logical.

Checkpoint

Pen Up & Fill Image

Draw a red triangle and a blue square.

1. Include the necessary code to start up a Python screen (import the library and generate a screen).
2. Add a custom title to your turtle screen.
3. Include a fillcolor of `red`.
4. Include a fillcolor of `blue`.
5. Include at least 2 instances of `begin_fill()`.
6. Include at least 2 instances of `end_fill()`.
7. Include at least 3 turns of 120 degrees.
8. Include at least 4 turns of 90 degrees.
9. Include at least 7 forward commands.



Requirements:

- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Add a custom title to your turtle screen.

- Include a fillcolor of `red` .
- Include a fillcolor of `blue` .
- Include at least 2 instances of `begin_fill()` .
- Include at least 2 instances of `end_fill()` .
- Include at least 3 turns of 120 degrees.
- Include at least 4 turns of 90 degrees.
- Include at least 7 forward commands.

Questions (10)

1. What command is used to make the turtle stop drawing while still allowing it to move?

MULTIPLE CHOICE

Choose the correct answer:

- A. `.penmove()`
- B. `.penup()`
- C. `.drawstop()`
- D. `.moveup()`

2. What is the purpose of the following command?

MULTIPLE CHOICE

`turtle.pendown()`

Choose the correct answer:

- A. To lift the pen and stop drawing.
- B. To change the pen color.
- C. To fill the shape.
- D. To start drawing.

3. How can you make the turtle draw a square and fill it with color?

MULTIPLE CHOICE

Choose the correct answer:

- A. Use `.drawsquare()`
- B. `.fill_square()`
- C. `.begin_fill()` and `.end_fill()`
- D. `.color_square()`

4. What command is used to start filling a shape with color?

Choose the correct answer:

- A. `.fillstart()`
- B. `.startcolor()`
- C. `.begin_fill()`
- D. `.colorstart()`

5. When drawing a square and filling it, where should the `.begin_fill()` command be placed?

MULTIPLE CHOICE

Choose the correct answer:

- A. Before drawing the shape.
- B. After completing the shape.
- C. Anywhere in the code.
- D. At the beginning of the script.

6. To change the fill color to red, what command should be used before the `.begin_fill()` command?

MULTIPLE CHOICE

Choose the correct answer:

- A. `.fillcolor("red")`
- B. `.color("red")`
- C. `.change_color("red")`
- D. `.setfillcolor("red")`

7. What does the `.end_fill()` command signify in the given code?

Choose the correct answer:

- A. It stops the turtle from moving.
- B. It completes the shape and fills it.
- C. It changes the color of the shape.
- D. It lifts the pen to stop drawing.

8. How do you draw a square without filling it?

Choose the correct answer:

- A. Omit the `.begin_fill()` and `.end_fill()` commands.
- B. Use `.drawsquare()` instead of `.begin_fill()`.
- C. Place `.fillstop()` before `.begin_fill()`.
- D. Add `.nofill()` after `.begin_fill()`.

9. Debug the following code:

DEBUG CODE

Code to Debug:

```
1 turtle.fillcolor("blue")
2 turtle.beginfill()
3 turtle.forward(100)
4 turtle.left(120)
5 turtle.forward(100)
6 turtle.left(120)
7 turtle.forward(100)
8 turtle.end_fill()
```

10. Debug the following code:

DEBUG CODE

Code to Debug:

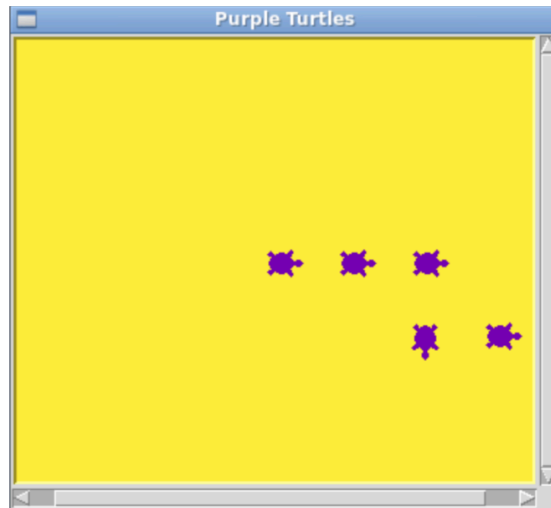
```
1 turtle.fillcolor("blue")
2 turtle.begin_fill()
3 turtle.forward(100)
4 turtle.left(120)
5 turtle.forward(100)
6 turtle.left(120)
7 turtle.forward(100)
8 turtle.end_fill)
```

Challenges (5)

1. Purple Turtles

Try using the stamp command with the pen up!

1. Include the necessary code to start up a Python screen (import the library and generate a screen).
2. Add a background color with a hexadecimal value.
3. Add a custom title to your turtle screen.
4. Add a `pencolor` to your turtle.
5. Add a `fillcolor` to your turtle.
6. Change the shape of your turtle to `turtle`.
7. Set the position of the pen to `penup`.
8. Move the turtle around the page and include at least 5 stamps.



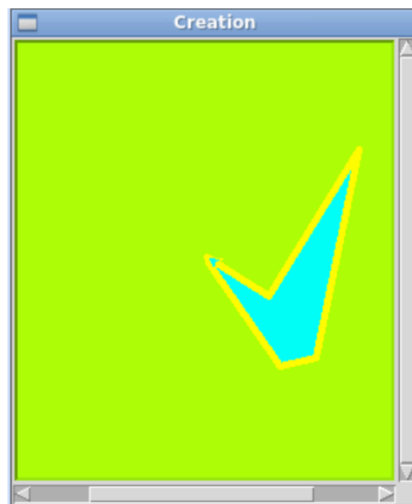
Requirements:

- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Add a background color with a hexadecimal value.
- Add a custom title to your turtle screen.
- Add a `pencolor` to your turtle.
- Add a `fillcolor` to your turtle.
- Change the shape of your turtle to `turtle`.
- Set the position of the pen to `penup`.
- Move the turtle around the page and include at least 5 stamps.

2. Create Your Own Shape!

Create your own shape with Python turtles!

1. Include the necessary code to start up a Python screen (import the library and generate a screen).
2. Add a background color with a hexadecimal value.
3. Add a custom title to your turtle screen.
4. Add at least 1 instance of `fillcolor` .
5. Add at least 1 instance of `begin_fill` .
6. Add at least 1 instance of `end_fill` .
7. Add at least 5 forward commands.
8. Adjust the `pensize` .
9. Adjust the `pencolor` .



Requirements:

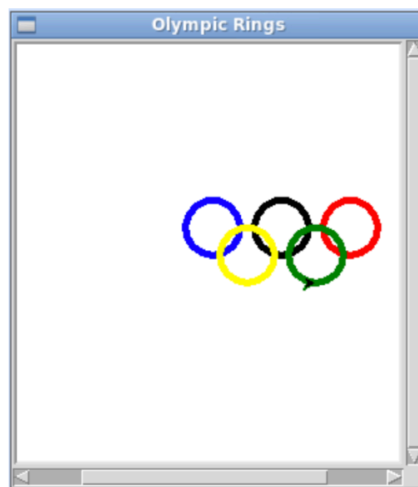
- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Add a background color with a hexadecimal value.
- Add at least 1 instance of `fillcolor` .
- Add at least 1 instance of `begin_fill` .
- Add at least 1 instance of `end_fill` .
- Add at least 5 forward commands.
- Adjust the `pensize` .
- Adjust the `pencolor` .

3. Olympic Rings

Draw the Olympic Rings!

1. Include the necessary code to start up a Python screen (import the library and generate a screen).
2. Add a custom title to your turtle screen.
3. Change the pensize to 5.
4. Use `penup` at least 4 times.
5. Use `pendown` at least 4 times.
6. Use `pencolor` at least 5 times with the colors `red` , `green` , `blue` , `black` , `yellow` .
7. Draw 5 circles.

Hint: Start with the top row. Then return to the starting point. Go down and then right to have your turtle face to the right to start your second row.



Requirements:

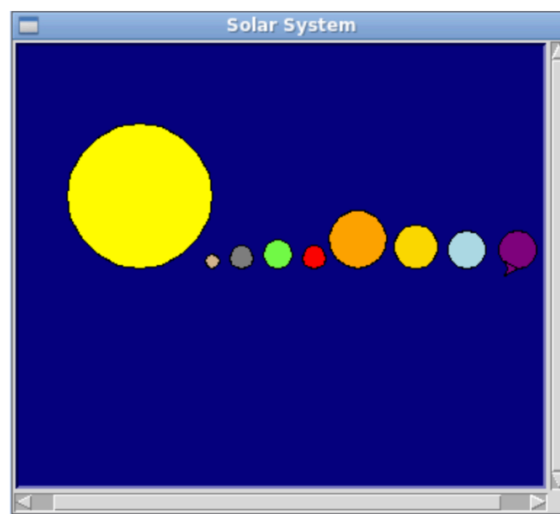
- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Add a custom title to your turtle screen.
- Change the pensize to 5.
- Use `penup` at least 4 times.
- Use `pendown` at least 4 times.
- Use `pencolor` at least 5 times with the colors `red` , `green` , `blue` , `black` , `yellow` .
- Draw 5 circles.

4. Solar System

Create a solar system! Draw all 8 planets as well as the sun. Make sure to pick up your pen in between the circles you draw!

(For this challenge, you will be filling circles. Do not use dots)

1. Include the necessary code to start up a Python screen (import the library and generate a screen).
2. Add a background color of `navy`.
3. Add a custom title to your turtle screen.
4. Add at least 9 instances of `penup`.
5. Add at least 9 instances of `pendown`.
6. Add at least 9 instances of `fillcolor`.
7. Add at least 9 instances of `begin_fill`.
8. Add at least 9 instances of `end_fill`.
9. Add at least 9 circle drawings.



Requirements:

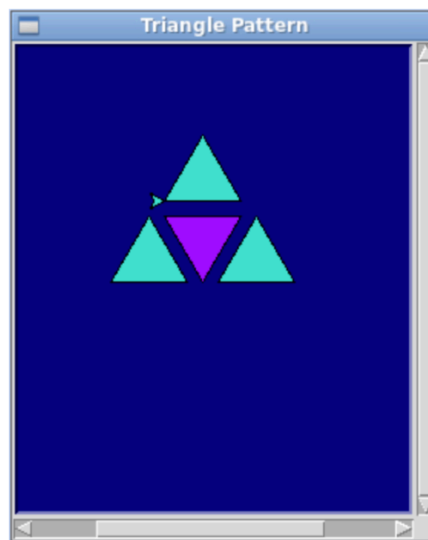
- Include the necessary code to start up a Python screen (import the library and generate a screen).
- Add a background color of `navy`.
- Add a custom title to your turtle screen.
- Add at least 9 instances of `penup`.
- Add at least 9 instances of `pendown`.
- Add at least 9 instances of `fillcolor`.
- Add at least 9 instances of `begin_fill`.
- Add at least 9 instances of `end_fill`.
- Add at least 9 circle drawings.

5. Triangle Pattern

Note: This challenge is tricky. Pay particular attention to the angles you are working with. Remember that 60 degrees is the angle for an equilateral triangle. (A triangle with equal sides.)

Create a triangle pattern! Create 4 equilateral triangles that seem to stack on each other like a card tower.

1. Include the necessary code to start up a python screen (import the library and generate a screen.)
2. Add a background color with a hexadecimal value.
3. Add a custom title to your turtle screen.
4. Add at least 3 instances of `penup` .
5. Add at least 3 instances of `pendown` .
6. Add at least 4 instances of `fillcolor` .
7. Add at least 4 instances of `begin_fill` .
8. Add at least 4 instances of `end_fill` .
9. Add at least 10 forward commands.



Requirements:

- Include the necessary code to start up a python screen (import the library and generate a screen.)
- Add at least 3 instances of `penup` .
- Add at least 3 instances of `pendown` .
- Add at least 4 instances of `fillcolor` .
- Add at least 4 instances of `begin_fill` .
- Add at least 4 instances of `end_fill` .
- Add at least 10 forward commands.

Answer Keys & Solutions

Checkpoint Solutions

Pen Up & Fill Image

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.title("PenUp & Fill Image")
5
6
7 turtle.fillcolor("red")
8 turtle.begin_fill()
9 turtle.left(120)
10 turtle.forward(100)
11 turtle.left(120)
12 turtle.forward(100)
13 turtle.left(120)
14 turtle.forward(100)
15 turtle.end_fill()
16
17 turtle.penup()
18 turtle.forward(50)
19 turtle.pendown()
20
21 turtle.fillcolor("blue")
22 turtle.begin_fill()
23 turtle.left(90)
24 turtle.forward(100)
25 turtle.right(90)
26 turtle.forward(100)
27 turtle.right(90)
28 turtle.forward(100)
29 turtle.right(90)
30 turtle.forward(100)
31 turtle.right(90)
32 turtle.end_fill()
```

Questions

1. What command is used to make the turtle stop drawing while still allowing it to move?

MULTIPLE CHOICE

Correct Answer:

A. .penmove()

✗ Incorrect

B. `.penup()`

✓ Correct

C. `.drawstop()`

✗ Incorrect

D. `.moveup()`

✗ Incorrect

Explanation:

It's like the turtle is dragging the pen behind it and when it lifts the pen up, it stops drawing.

2. What is the purpose of the following command?

MULTIPLE CHOICE

Correct Answer:

A. To lift the pen and stop drawing.

✗ Incorrect

B. To change the pen color.

✗ Incorrect

C. To fill the shape.

✗ Incorrect

D. To start drawing.

✓ Correct

Explanation:

When the pen is down, it starts drawing.

3. How can you make the turtle draw a square and fill it with color?

MULTIPLE CHOICE

Correct Answer:

A. Use `.drawsquare()`

✗ Incorrect

B. `.fill_square()`

✗ Incorrect

C. `.begin_fill()` and `.end_fill()`

✓ Correct

D. `.color_square()`

✗ Incorrect

Explanation:

It requires two commands.

MULTIPLE CHOICE

4. What command is used to start filling a shape with color?

Correct Answer:

- A. `.fillstart()` ✗ Incorrect
- B. `.startcolor()` ✗ Incorrect
- C. `.begin_fill()` ✓ Correct
- D. `.colorstart()` ✗ Incorrect

Explanation:

The command is to begin filling

5. When drawing a square and filling it, where should the `.begin_fill()` command be placed?

MULTIPLE CHOICE

Correct Answer:

- A. Before drawing the shape. ✓ Correct
- B. After completing the shape. ✗ Incorrect
- C. Anywhere in the code. ✗ Incorrect
- D. At the beginning of the script. ✗ Incorrect

Explanation:

The begin fill command needs to be planned ahead of time

6. To change the fill color to red, what command should be used before the `.begin_fill()` command?

MULTIPLE CHOICE

Correct Answer:

- A. `.fillcolor("red")` ✓ Correct
- B. `.color("red")` ✗ Incorrect
- C. `.change_color("red")` ✗ Incorrect
- D. `.setfillcolor("red")` ✗ Incorrect

Explanation:

We would be updating the fillcolor

7. What does the .end_fill() command signify in the given code?

MULTIPLE CHOICE

Correct Answer:

- A. It stops the turtle from moving. ✗ Incorrect
- B. It completes the shape and fills it. ✓ Correct
- C. It changes the color of the shape. ✗ Incorrect
- D. It lifts the pen to stop drawing. ✗ Incorrect

Explanation:

This is the command to finish filling

8. How do you draw a square without filling it?

MULTIPLE CHOICE

Correct Answer:

- A. Omit the .begin_fill() and .end_fill() commands. ✓ Correct
- B. Use .drawsquare() instead of .begin_fill(). ✗ Incorrect
- C. Place .fillstop() before .begin_fill(). ✗ Incorrect
- D. Add .nofill() after .begin_fill(). ✗ Incorrect

Explanation:

The default behavior is to not fill the shapes

9. Debug the following code:

DEBUG CODE

Incorrect Code:

```
1 turtle.fillcolor("blue")
2 turtle.beginfill()
3 turtle.forward(100)
4 turtle.left(120)
5 turtle.forward(100)
```

```
6 turtle.left(120)
7 turtle.forward(100)
8 turtle.end_fill()
```

Correct Solution:

```
1 turtle.fillcolor("blue")
2 turtle.begin_fill()
3 turtle.forward(100)
4 turtle.left(120)
5 turtle.forward(100)
6 turtle.left(120)
7 turtle.forward(100)
8 turtle.end_fill()
```

Explanation:

This code is missing an underscore

10. Debug the following code:

DEBUG CODE

Incorrect Code:

```
1 turtle.fillcolor("blue")
2 turtle.begin_fill()
3 turtle.forward(100)
4 turtle.left(120)
5 turtle.forward(100)
6 turtle.left(120)
7 turtle.forward(100)
8 turtle.end_fill)
```

Correct Solution:

```
1 turtle.fillcolor("blue")
2 turtle.begin_fill()
3 turtle.forward(100)
4 turtle.left(120)
5 turtle.forward(100)
6 turtle.left(120)
7 turtle.forward(100)
8 turtle.end_fill()
```

Explanation:

This code is missing a parenthesis

Challenges

1. Purple Turtles

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.title("Purple Turtles")
5
6 turtle.bgcolor("#ffee38")
7
8 turtle.pencolor("#7400b3")
9 turtle.fillcolor("#7400b3")
10 turtle.shape("turtle")
11
12 turtle.penup()
13 turtle.stamp()
14 turtle.forward(50)
15 turtle.stamp()
16 turtle.forward(50)
17 turtle.stamp()
18 turtle.right(90)
19 turtle.forward(50)
20 turtle.stamp()
21 turtle.left(90)
22 turtle.forward(50)
23 turtle.stamp()
```

2. Create Your Own Shape!

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.title("Creation")
5
6 turtle.bgcolor("#abff0f")
7 turtle.pensize(5)
8 turtle.pencolor("yellow")
9
10 turtle.fillcolor("#0ffffb")
11 turtle.begin_fill()
12 turtle.right(55)
13 turtle.forward(88)
14 turtle.left(70)
15 turtle.forward(25)
16 turtle.left(63)
17 turtle.forward(148)
18 turtle.left(160)
19 turtle.forward(120)
20 turtle.right(90)
21 turtle.forward(50)
22 turtle.end_fill()
```

3. Olympic Rings

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.title("Olympic Rings")
5 turtle.pensize(5)
6
7 turtle.pencolor("blue")
8 turtle.circle(20)
9 turtle.penup()
10
11 turtle.forward(50)
12 turtle.pendown()
13 turtle.pencolor("black")
14 turtle.circle(20)
15 turtle.penup()
16
17 turtle.forward(50)
18 turtle.pendown()
19 turtle.pencolor("red")
20 turtle.circle(20)
21 turtle.penup()
22
23 turtle.backward(100)
24 turtle.right(90)
25 turtle.forward(20)
26 turtle.left(90)
27 turtle.forward(25)
28
29 turtle.pendown()
30 turtle.pencolor("yellow")
31 turtle.circle(20)
32 turtle.penup()
33
34 turtle.forward(50)
35 turtle.pendown()
36 turtle.pencolor("green")
37 turtle.circle(20)
```

4. Solar System

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.title("Solar System")
5
6 turtle.bgcolor("navy")
7
8 turtle.penup()
9 turtle.backward(100)
10
```

```
11 turtle.pendown()
12 turtle.fillcolor("yellow")
13 turtle.begin_fill()
14 turtle.circle(50)
15 turtle.end_fill()
16
17 turtle.penup()
18 turtle.forward(50)
19 turtle.pendown()
20 turtle.fillcolor("tan")
21 turtle.begin_fill()
22 turtle.circle(5)
23 turtle.end_fill()
24
25 turtle.penup()
26 turtle.forward(20)
27 turtle.pendown()
28 turtle.fillcolor("gray")
29 turtle.begin_fill()
30 turtle.circle(8)
31 turtle.end_fill()
32
33 turtle.penup()
34 turtle.forward(25)
35 turtle.pendown()
36 turtle.fillcolor("#76fb46")
37 turtle.begin_fill()
38 turtle.circle(10)
39 turtle.end_fill()
40
41 turtle.penup()
42 turtle.forward(25)
43 turtle.pendown()
44 turtle.fillcolor("red")
45 turtle.begin_fill()
46 turtle.circle(8)
47 turtle.end_fill()
48
49 turtle.penup()
50 turtle.forward(30)
51 turtle.pendown()
52 turtle.fillcolor("orange")
53 turtle.begin_fill()
54 turtle.circle(20)
55 turtle.end_fill()
56
57 turtle.penup()
58 turtle.forward(40)
59 turtle.pendown()
60 turtle.fillcolor("gold")
61 turtle.begin_fill()
62 turtle.circle(15)
63 turtle.end_fill()
64
65 turtle.penup()
66 turtle.forward(35)
67 turtle.pendown()
68 turtle.fillcolor("lightblue")
69 turtle.begin_fill()
70 turtle.circle(13)
71 turtle.end_fill()
72
73 turtle.penup()
74 turtle.forward(35)
75 turtle.pendown()
76 turtle.fillcolor("purple")
```

```
77 turtle.begin_fill()
78 turtle.circle(13)
79 turtle.end_fill()
```

5. Triangle Pattern

Solution:

```
1 import turtle
2 turtle.getscreen()
3
4 turtle.title("Triangle Pattern")
5
6 turtle.bgcolor("navy")
7
8 turtle.fillcolor("turquoise")
9 turtle.begin_fill()
10 turtle.forward(50)
11 turtle.left(120)
12 turtle.forward(50)
13 turtle.left(120)
14 turtle.forward(50)
15 turtle.left(120)
16 turtle.end_fill()
17
18 turtle.penup()
19 turtle.backward(10)
20 turtle.pendown()
21
22 turtle.left(60)
23
24 turtle.fillcolor("#9f0fff")
25 turtle.begin_fill()
26 turtle.forward(50)
27 turtle.left(120)
28 turtle.forward(50)
29 turtle.left(120)
30 turtle.forward(50)
31 turtle.end_fill()
32
33 turtle.left(60)
34 turtle.penup()
35 turtle.backward(10)
36 turtle.pendown()
37
38 turtle.fillcolor("turquoise")
39 turtle.begin_fill()
40 turtle.left(120)
41 turtle.forward(50)
42 turtle.left(120)
43 turtle.forward(50)
44 turtle.left(120)
45 turtle.forward(50)
```

```
46 turtle.end_fill()
47
48 turtle.left(90)
49 turtle.penup()
50 turtle.forward(53)
51 turtle.right(90)
52 turtle.backward(15)
53 turtle.pendown()
54
55 turtle.fillcolor("turquoise")
56 turtle.begin_fill()
57 turtle.forward(50)
58 turtle.left(120)
59 turtle.forward(50)
60 turtle.left(120)
61 turtle.forward(50)
62 turtle.left(120)
63 turtle.end_fill()
```