PYTHON PROGRAMMING

Key Stage 3

COMPUTING

TEACHER NOTES

PART II

Using Minecraft Pi and Codecadmy

Minecraft Pi Book

Craig Richardson

June 13, 2013

Contents

Chapter 1

Teacher's Introduction to the Book

This is the teacher's edition of the Python Programming with Minecraft Pi and Codecademy book. It includes an explanation of the main book and suggested answers to each exercise. You will find the answers from Chapter 2 onwards.

The Minecraft Pi Python book is intended to support learners as they learn to program with Python. The material is split into two categories: exercises and reference. Reference materials concentrate on the low-level skills of remembering and understanding, whereas exercises focus on the higher-level skills of application, problem solving, analysis, evaluation and creation.

All exercises use Minecraft Pi edition on the Raspberry Pi to provide a fun and challenging environment for students to practice and develop their programming skills. The main focus of exercises is problem solving, which is an essential programming skill. Students should be encouraged to deviate from the brief and develop their own ideas as creativity is another essential skill for programmers. Share your students' ideas for programs and I might include them in future versions of the book.

1.1 Exercises

At the start of every chapter there are a set of exercises that help learners develop their programming skills. The emphasis of the exercises is to apply the knowledge and understanding in order to create a programs.

All programming exercises are challenging and differentiated. Instead

of merely telling the student to copy code, each exercise sets the player a challenge with instructions of varying difficulty. They apply concepts learned in the current chapter and corresponding Codecademy lesson to solve these challenges and gain problem solving skills. Extension exercises are included with every exercise, enabling students who are more comfortable with the exercises to develop further.

This version of the book uses Minecraft Pi on the Raspberry Pi to practice programming in exercises. The Raspberry Pi is an excellent, low-cost computer designed to teach programming. It has a strong, enthusiastic community and is an excellent way to get started programming.

Minecraft Pi edition is an official version of Minecraft that has been developed for the Raspberry Pi. Thanks to its amazing developers at Mojang, it is free software and has a special feature, an API, that allows users to control the game with programming. It is very simple to use and gives learners a very engaging and creative way to practice programming.

1.2 Reference

The reference parts of the book support a learner's ability to understand and remember concepts. However, they are not intended to be used in isolation. They were designed to support the Python track at Codecademy. Codecademy is a free online learning system that gives learners hands-on experience and guidance with programming.

The reference parts of the book are intended as a supplement to the Codecademy lessons in a number of ways:

- Extra support for students who need help understanding concepts introduced in Codecademy
- Reference for learners when they are developing their own programs
- Revision materials for students undertaking assessment

Of course it is possible to read the book from start to finish, however there is nothing better than gaining hands-on experience of programming.

If you are a teacher I cannot recommend Codecademy enough. Due to the structure of the site it provides exactly the right support for a beginner meaning they require less direct intervention from you. As you no longer

need to provide debugging advice and marking for all your students you can use your time more effectively and support the students that really need your help. Monitoring progress and providing clear targets to students is essential for effective progression through the site.

1.3 KS3 Curriculum

Not all of the topics covered in the Codecademy Python track and the book are essential for the UK Key Stage 3 draft Computing curriculum. I have summarised what I believe are the most important units/chapters:

1. Data types and maths operators
2. String and command line input and output
3. Boolean Logic, Conditionals and If statements
4. Functions
5. Lists
6. Loops

Furthermore, within each of these topics there are concepts which I believe are a minimum requirement for students to understand and a number of more advanced topics. For example in the lists unit students should at a minimum understand the purpose of lists, index positions and accessing/changing list values. Intermediate knowledge include loops with lists, appending items and removing items. Advanced topics include list slicing and so on. A breakdown of these levels of understanding by topic may be included in a future update of this book.

Of course you should encourage your students to go further if they have the time and enthusiasm.

In a future draft I will state how each topic of the curriculum is covered in the book. There will of course be some gaps, which will eventually be filled by other resources.

The material can also be used for a different curriculum as programming concepts stay the same.

1.4 Open Source

The book is open source, meaning anyone can change and modify it as they please. A creative commons license is used, which carries some restrictions. Details of the license of the book can be found at the start of the book.

The book is written in LaTeX, a free, open-source typesetting language. If you're not familiar with LaTeX, it is similar to HTML in that it formats text. A guide to getting started with LaTeX can be found via Google search and an explanation of the conventions used in the book can be found in the first chapter.

The book was designed so that different exercises can be swapped into the book. Say you don't want to teach students with Minecraft Pi, you want to use SimpleCV, Blender or PyGame. Do this. Create your own exercises and swap them for the ones in the book and we now have a different edition of the same book that other people will find useful. All of the reference materials can stay the same as they make no mention of Minecraft Pi.

Chapter 2

Python Syntax

2.1 Teleport the Player

```
#connect to Minecraft
import mcpi.minecraft as minecraft
mc = minecraft.Minecraft.create()

"""set x, y and z variables
to represent co-ordinates"""
x = 10
y = 11
z = 12

#change the player's position
mc.player.setTilePos(x, y, z)
```

2.2 Teleport the Player Precisely

```
#Connect to Minecraft
import mcpi.minecraft as minecraft
mc = minecraft.Minecraft.create()

"""Set x, y and z variables
to floats that represent
co-ordinates"""
x = 10.5
y = 10.5
z = 10.5
```

```
11
12 #Set the player's position
13 mc.player.setPos(x, y, z)
```

2.3 Teleportation Tour

```
1  # connect to Minecraft
2  import mcpi.minecraft as minecraft
3  mc = minecraft.Minecraft.create()
4  # import the time module
5  import time
6  # set x, y and z variables
7  x = 10
8  y = 11
9  z = 12
10 # change the player's position
11 mc.player.setTilePos(x,y,z)
12 # wait 5 seconds
13 time.sleep(5)
14 # change x, y and z variables
15 x = 13
16 y = 14
17 z = 15
18 # change the player's position
19 mc.player.setTilePos(x,y,z)
```

2.4 Stop Smashing Things

True version:

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 mc.setting('world_immutable', True)
```

False version:

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 mc.setting('world_immutable', False)
```

Chapter 3

Maths Operations

3.1 Stacking Blocks

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4  x = 6
5  y = 5
6  z = 18
7  blockType = 103
8
9  mc.setBlock(x, y, z, blockType)
10
11 y = y + 1
12 mc.setBlock(x, y, z, blockType)
13
14 y = y + 1
15 mc.setBlock(x, y, z, blockType)
16
17 y = y + 1
18 mc.setBlock(x, y, z, blockType)
```

3.2 Super Jump

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4  position = mc.player.getPos()
```

```
5 x = position.x
6 y = position.y
7 z = position.z
8
9 y = y + 10
10 mc.player.setPos(x,y,z)
```

3.3 Set Block Below Player

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 playerPosition = mc.player.getPos()
5 x = playerPosition.x
6 y = playerPosition.y
7 z = playerPosition.z
8 blockType = 20
9
10 y -= 1
11
12 mc.setBlock(x,y,z,blockType)
```

3.4 Speed Building

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 position = mc.player.getTilePos()
5 x = position.x
6 y = position.y
7 z = position.z
8
9 width = 10
10 length = 12
11 height = 5
12
13 blockType = 4
14
15 mc.setBlocks(x, y, z, x + width, y + height, z +
     ↪ length, blockType)
```

```
16 mc.setBlocks(x + 1, y + 1, z + 1, x + width - 1, y
     ↪ + height - 1, z + length - 1, 0)
```

3.5 Proportions

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4  position = mc.player.getTilePos()
5  x = position.x
6  y = position.y
7  z = position.z
8
9  width = 10
10 length = width * 2
11 height = width / 2
12
13 blockType = 4
14
15 mc.setBlocks(x, y, z, x + width, y + height, z +
     ↪ length, blockType)
16 mc.setBlocks(x + 1, y + 1, z + 1, x + width - 1, y
     ↪ + height - 1, z + length - 1, 0)
```

Chapter 4

Strings and Console Output

4.1 Hello Minecraft World

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 mc.postToChat(”Hello”)
```

4.2 Inputting Your Message

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 message = raw_input(”Please enter a message: ”)
5
6 mc.postToChat(message)
```

4.3 User Name

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 name = raw_input(”Enter a user name: ”)
5 message = raw_input(”Enter a message: ”)
6 mc.postToChat(name + ”: ” + message)
```

4.4 Mad Libs

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 word1 = raw_input("Noun: ")
5 word2 = raw_input("Noun: ")
6 word3 = raw_input("Noun: ")
7
8 song = "Jingle %s, Jingle %s, Jingle all the %s" %
    ↪ (word1, word2, word3)
9 mc.postToChat(song)
```

4.5 Create a Block with Input

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 blockType = input("Please enter an integer: ")
5
6 position = mc.player.getTilePos()
7 mc.setBlock(position.x,position.y, position.z,
    ↪ blockType)
```

4.6 Sprint Record

```
1 import mcpi.minecraft as minecraft
2 import time
3 mc = minecraft.Minecraft.create()
4
5 mc.postToChat("3...")
6 time.sleep(1)
7 mc.postToChat("2...")
8 time.sleep(1)
9 mc.postToChat("1...")
10 time.sleep(1)
11 mc.postToChat("Go!")
12
13 position = mc.player.getTilePos()
14 xStart = position.x
```

```
15 yStart = position.y
16 zStart = position.z
17
18 time.sleep(10)
19
20 position = mc.player.getTilePos()
21 xEnd = position.x
22 yEnd = position.y
23 zEnd = position.z
24
25 message = "You have moved %s blocks along the
     ↪ x-axis, %s blocks along the y-axis and %s
     ↪ blocks along the z-axis." % (xEnd - xStart,
     ↪ yEnd - yStart, zEnd - zStart)
26
27 mc.postToChat(message)
```

Chapter 5

Comparators and Control Flow

5.1 Swimming

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 position = mc.player.getTilePos()
5
6 blockType = mc.getBlock(position.x,position.y,
    ↪ position.z)
7
8 mc.postToChat(str(blockType == 9))
```

5.2 Do you want to stop smashing things?

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 choice = raw_input("Do you want to make the world
    ↪ immutable? Yes/No: ")
5 if choice == "Yes":
6     mc.setting('world_immutable', True)
7     mc.postToChat("World is immutable")
8 else:
9     mc.setting('world_immutable', False)
10    mc.postToChat("World is mutable")
```

5.3 Bring us a Shrubbery

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4  x = -109
5  y = 9
6  z = -72
7  blockType = mc.getBlock(x,y,z)
8  print blockType
9
10 if blockType == 6:
11     mc.postToChat("Ni!")
12 else:
13     mc.postToChat("Bring a shrubbery to " + str(x)
       ↪ + ", " + str(y) + ", " + str(z))
```

5.4 Take a Shower

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4  x1 = -16
5  x2 = -19
6  z1 = -79
7  z2 = -81
8  y = 12
9  pos = mc.player.getTilePos()
10
11 onx = pos.x >= x2 and pos.x <= x1
12 onz = pos.z >= z2 and pos.z <= z1
13
14 if onx and onz and pos.y == y:
15     mc.setBlocks(x1, y + 2, z1, x2, y + 2, z2, 8)
16 else:
17
18     mc.setBlocks(x1, y + 2, z1, x2, y + 2, z2, 0)
```

5.5 Secret Passage

```
import mcpi.minecraft as minecraft
mc = minecraft.Minecraft.create()

x1 = 5
x2 = 8
z1 = -66
z2 = -61
y = -9
pos = mc.player.getTilePos()

print pos

onx = pos.x >= x1 and pos.x <= x2
onz = pos.z >= z1 and pos.z <= z2
block = mc.getBlock(8, -8, -65)

if onx and onz and pos.y == y and block == 57:
    mc.postToChat("access granted")
    mc.setBlocks(9, -9, -64, 9, -8, -63, 0)
elif onx and onz and pos.y == y and block != 57:
    mc.setBlocks(9, -9, -64, 9, -9, -63, 24, 2)
    mc.setBlocks(9, -8, -64, 9, -8, -63, 24, 1)
    mc.postToChat("lava")
    mc.setBlocks(x1, y - 1, z1, x2, y - 1, z2, 10)
else:
    mc.setBlocks(9, -9, -64, 9, -9, -63, 24, 2)
    mc.setBlocks(9, -8, -64, 9, -8, -63, 24, 1)
    mc.postToChat("Put a diamond on the pedestal")
    mc.setBlocks(x1, y - 1, z1, x2, y - 1, z2, 48)
```

Chapter 6

Functions

6.1 A Forest

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4
5  def makeTree(x,y,z):
6      wood = 17
7      leaf = 18
8      nothing = 0
9      mc.setBlocks(x,y,z,x,y+3,z,wood)
10     mc.setBlocks(x-2,y+4,z-2,x+2,y+5,z+2,leaf)
11     mc.setBlocks(x-1,y+6,z-1,x+1,y+7,z+1,leaf)
12     mc.setBlock(x-1,y+7,z-1,nothing)
13     mc.setBlock(x+2,y+5,z-2,nothing)
14     mc.setBlock(x+2,y+4,z+2,nothing)
15     mc.setBlocks(x+1,y+6,z-1,x+1,y+7,z-1,nothing)
16     mc.setBlocks(x+1,y+6,z+1,x+1,y+7,z+1,nothing)
17     mc.setBlocks(x-1,y+6,z+1,x-1,y+7,z+1,nothing)
18
19 pos = mc.player.getPos()
20 x = pos.x
21 y = pos.y
22 z = pos.z
23 makeTree(x+1,y,z)
24 makeTree(x+8,y,z)
25 makeTree(x+15,y,z)
26 makeTree(x+1,y,z+7)
```

```
27 makeTree(x+8,y,z+7)
28 makeTree(x+15,y,z+7)
29 makeTree(x+1,y,z-7)
30 makeTree(x+8,y,z-7)
31 makeTree(x+15,y,z-7)
```

6.2 Arming TNT

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4
5  def armTnt(x, y, z):
6      tnt = 46
7      block = mc.getBlock(x, y, z)
8      if block == tnt:
9          mc.setBlock(x, y, z, tnt, 1)
10         mc.postToChat("TNT armed")
11     else:
12         mc.postToChat("Block is not TNT")
13
14 pos = mc.player.getTilePos()
15 armTnt(pos.x, pos.y - 1, pos.z)
```

6.3 Wool Colour

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4
5  def getWool(colour):
6      if colour == "white":
7          return 0
8      elif colour == "orange":
9          return 1
10     elif colour == "purple":
11         return 2
12     elif colour == "blue":
13         return 3
14     elif colour == "yellow":
15         return 4
```

```
16      elif colour == "green":
17          return 5
18      elif colour == "pink":
19          return 6
20      elif colour == "charcoal":
21          return 7
22      elif colour == "grey":
23          return 8
24      elif colour == "turquoise":
25          return 9
26      elif colour == "lillac":
27          return 10
28      elif colour == "dark blue":
29          return 11
30      elif colour == "brown":
31          return 12
32      elif colour == "dark green":
33          return 13
34      elif colour == "red":
35          return 14
36      elif colour == "black":
37          return 15
38      else:
39          return -1
40
41  colourString = raw_input("Enter a colour: ")
42  colourId = getWool(colourString)
43  if colourId != -1:
44      pos = mc.player.getTilePos()
45      mc.setBlock(pos.x, pos.y - 1, pos.z, 35,
          ↪ colourId)
46  else:
47      mc.postToChat(colourString + " is not a valid
          ↪ colour.")
```

6.4 Turtle

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4  pos = mc.player.getPos()
5  x = pos.x
```

```
y = pos.y
z = pos.z

pen = True
blockType = 35
state = 1


def forward(blocks):
    global x
    global y
    global z
    if pen:
        mc.setBlocks(x + 1, y, z, x + blocks, y,
           ↪ z, blockType, state)
    x += blocks


def backward(blocks):
    global x
    global y
    global z
    if pen:
        mc.setBlocks(x - 1, y, z, x - blocks, y,
           ↪ z, blockType, state)
    x -= blocks


def right(blocks):
    global x
    global y
    global z
    if pen:
        mc.setBlocks(x, y, z + 1, x, y, z +
           ↪ blocks, blockType, state)
    z += blocks


def left(blocks):
    global x
    global y
    global z
    if pen:
```

```
46          mc.setBlocks(x, y, z - 1, x, y, z -
               ↪ blocks, blockType, state)
47      z -= blocks
48
49
50  def up(blocks):
51      global x
52      global y
53      global z
54      if pen:
55          mc.setBlocks(x, y + 1, z, x, y + blocks,
               ↪ z, blockType, state)
56      y += blocks
57
58
59  def down(blocks):
60      global x
61      global y
62      global z
63      if pen:
64          mc.setBlocks(x, y - 1, z, x, y - blocks,
               ↪ z, blockType, state)
65      y -= blocks
66
67
68  def penDown():
69      global pen
70      pen = True
71
72
73  def penUp():
74      global pen
75      pen = False
76
77  # square
78  penDown()
79  forward(3)
80  left(3)
81  backward(3)
82  right(3)
83
84  # move to a new location
85  penUp()
```

```
86 right(10)
87
88 #parallel lines
89 penDown()
90 up(10)
91 penUp()
92 up(1)
93 left(2)
94 penDown()
95 down(10)
96 penUp()
97 down(1)
98 left(2)
99 penDown()
100 up(10)
101
102 # move to a new location
103 penUp()
104 down(10)
105 right(10)
106
107 # face
108 penUp()
109 up(1)
110 penDown()
111 up(4)
112 penUp()
113 up(1)
114 penDown()
115 right(1)
116 penUp()
117 up(1)
118 penDown()
119 right(4)
120 penUp()
121 right(1)
122 penDown()
123 down(1)
124 penUp()
125 right(1)
126 penDown()
127 down(4)
128 penUp()
```

```
129 down(1)
130 penDown()
131 left(1)
132 penUp()
133 down(1)
134 penDown()
135 left(4)
136 penUp()
137 left(1)
138 penDown()
139 up(1)
140 penUp()
141 up(2)
142 penDown()
143 right(1)
144 penUp()
145 down(1)
146 penDown()
147 right(2)
148 penUp()
149 right(1)
150 penDown()
151 up(1)
152 penUp()
153 up(1)
154 penDown()
155 up(1)
156 penUp()
157 left(2)
158 penDown()
159 left(1)
```

6.5 Import Block Module

Students don't really write code for this. They just find a file.

Chapter 7

Lists and Dictionaries

7.1 Glitching Signs

```
1  import mcpi.minecraft as minecraft
2  import time
3  mc = minecraft.Minecraft.create()
4
5  states = [0, 5, 6, 11, 3, 8, 9, 2, 12, 7, 4, 16]
6
7  pos = mc.player.getPos()
8  x = pos.x + 2
9  y = pos.y
10 z = pos.z
11 blockId = 63
12
13 for state in states:
14     mc.setBlock(x, y, z, blockId, state)
15     time.sleep(0.3)
```

7.2 Blocks by Numbers

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4  pos = mc.player.getTilePos()
5  x = pos.x
6  y = pos.y
7  z = pos.z
```

```
 8
 9 blocks = [47, 20, 103, 81, 57]
10
11 #bookshelf, glass, melon, cactus, diamond
12 y += 1
13 minecraft.setBlock(x, y, z, blocks[0])
14 minecraft.setBlock(x, y, z + 1, blocks[1])
15 minecraft.setBlock(x, y, z + 2, blocks[2])
16 minecraft.setBlock(x, y, z + 3, blocks[3])
17 minecraft.setBlock(x, y, z + 4, blocks[4])
18
19 #diamond, diamond, cactus, melon, bookshelf
20 y += 1
21 minecraft.setBlock(x, y, z, blocks[4])
22 minecraft.setBlock(x, y, z + 1, blocks[4])
23 minecraft.setBlock(x, y, z + 2, blocks[3])
24 minecraft.setBlock(x, y, z + 3, blocks[2])
25 minecraft.setBlock(x, y, z + 4, blocks[0])
26
27 #melon, glass, melon, glass, melon
28 y += 1
29 minecraft.setBlock(x, y, z, blocks[2])
30 minecraft.setBlock(x, y, z + 1, blocks[1])
31 minecraft.setBlock(x, y, z + 2, blocks[2])
32 minecraft.setBlock(x, y, z + 3, blocks[1])
33 minecraft.setBlock(x, y, z + 4, blocks[2])
34
35 #bookshelf, bookshelf, glass, bookshelf, bookshelf
36 y += 1
37 minecraft.setBlock(x, y, z, blocks[0])
38 minecraft.setBlock(x, y, z + 1, blocks[0])
39 minecraft.setBlock(x, y, z + 2, blocks[1])
40 minecraft.setBlock(x, y, z + 3, blocks[0])
41 minecraft.setBlock(x, y, z + 4, blocks[0])
42
43 #diamond, cactus, melon, glass, bookshelf
44 y += 1
45 minecraft.setBlock(x, y, z, blocks[4])
46 minecraft.setBlock(x, y, z + 1, blocks[3])
47 minecraft.setBlock(x, y, z + 2, blocks[2])
48 minecraft.setBlock(x, y, z + 3, blocks[1])
49 minecraft.setBlock(x, y, z + 4, blocks[0])
```

7.3 Team Camera

```
1  import mcpi.minecraft as minecraft
2  import time
3  mc = minecraft.Minecraft.create()
4
5  players = mc.getPlayerEntityIds()
6  mc.camera.setFollow(players[1])
7  # single player alternative
8  # mc.camera.setFollow(players[0])
9  time.sleep(10)
10 mc.camera.setNormal()
```

7.4 Dictionary of Wool

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4
5  def getWool(colour):
6      wool = {"white": 0, "orange": 1, "purple": 2,
          ↪ "blue": 3,
7           "yellow": 4, "green": 5, "pink": 6,
              ↪ "charcoal": 7,
8           "grey": 8, "turquoise": 9, "lillac": 10,
              ↪ "dark blue": 11,
9           "brown": 12, "dark green": 13, "red": 14,
              ↪ "black": 15}
10     return wool[colour]
11
12
13 colourString = raw_input("Enter a colour: ")
14 colourId = getWool(colourString)
15 if colourId != -1:
16     pos = mc.player.getTilePos()
17     mc.setBlock(pos.x, pos.y - 1, pos.z, 35,
          ↪ colourId)
18 else:
19     mc.postToChat(colourString + " is not a valid
          ↪ colour.")
```

7.5 Hack a Friend's Game

Note: This code has not been tested.

```
1  import mcpi.minecraft as minecraft
2  # local connection
3  me = minecraft.Minecraft.create()
4  """ The IP following addresses will need to be
     ↪ changed
5  to match the IP address of connected players every
     ↪ time
6  this program is run.
7
8  Variable names used here are just for
     ↪ illustration, students
9  can change these to whatever they want"""
10 # Connection to Andrew's Raspberry Pi
11 charlotte =
     ↪ minecraft.Minecraft.create("192.168.1.62")
12 david = minecraft.Minecraft.create("192.168.1.29")
13 amy = minecraft.Minecraft.create("192.168.1.96")
14
15 # list of players
16 players = [me, charlotte, david, amy]
17
18 """ code beyond this point is just an example.
19 Students will change it depending on their own
     ↪ ideas"""
20 for player in players:
21     player.player.setTilePos(16,16,16)
22
23 players[charlotte].setting("world_immutable", True)
```

Chapter 8

Functions and Lists

8.1 Pixel Art

```
import mcpi.minecraft as minecraft
mc = minecraft.Minecraft.create()

onBlock = 35
onBlockState = 3
offBlock = 35
offBlockState = 15

pixelArt = [[0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0],
    [0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0],
    [0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0],
    [0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0],
    [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1],
    [1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1],
    [1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1],
    [0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0]]

pos = mc.player.getTilePos()
x = pos.x
y = pos.y + 8
z = pos.z

for row in range(len(pixelArt)):
    for pixel in range(len(pixelArt[row])):
        if pixelArt[row][pixel] == 1:
```

```
26              mc.setBlock(x, y - row, z + pixel,
                  ↪ onBlock, onBlockState)
27          elif pixelArt[row][pixel] == 0:
28              mc.setBlock(x, y - row, z + pixel,
                  ↪ offBlock, offBlockState)
```

8.2 Shadow Castle

Shadow castle without correction for block states bug:

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4
5  def getBlocks(x1, y1, z1, x2, y2, z2):
6      xhigh = max(x1, x2)
7      xlow = min(x1, x2)
8      yhigh = max(y1, y2)
9      ylow = min(y1, y2)
10     zhigh = max(z1, z2)
11     zlow = min(z1, z2)
12
13     blocks = []
14     for x in range(xhigh - xlow + 1):
15         blocks.append([])
16         for y in range(yhigh - ylow + 1):
17             blocks[x].append([])
18             for z in range(zhigh - zlow + 1):
19                 blocks[x][y].append([])
20                 block = mc.getBlock(xlow + x, ylow
                       ↪ + y, zlow + z)
21                 blocks[x][y][z] = block
22     return blocks
23
24 prompt = raw_input("Move to the first corner and
     ↪ press enter in this window")
25 pos1 = mc.player.getTilePos()
26
27 prompt = raw_input("Move to the opposite corner
     ↪ and press enter in this window")
28 pos2 = mc.player.getTilePos()
29 print "Please wait"
```

```
30
31 structure = getBlocks(pos1.x, pos1.y, pos1.z,
     ↪ pos2.x, pos2.y, pos2.z)
32
33 print "Move to the location you want to place the
     ↪ structure"
34 prompt = raw_input("and press enter in this
     ↪ window.")
35 pos3 = mc.player.getTilePos()
36
37 for x in range(len(structure)):
38     for y in range(len(structure[x])):
39         for z in range(len(structure[x][y])):
40             blockType = structure[x][y][z]
41             if blockType == 4:
42                 blockType = 49
43             elif blockType == 9 or blockType == 8:
44                 blockType = 10
45             mc.setBlock(pos3.x + x, pos3.y + y,
                 ↪ pos3.z + z, blockType)
```

To fix the block states bug, students should change the getBlocks() functions to use the getBlockWithData() method instead of getBlock() method. The rest of the code should be adjusted to accomodate this. Shadow castle with correction for block states bug:

```
1  import mcpi.minecraft as minecraft
2  mc = minecraft.Minecraft.create()
3
4
5  def getBlocksWithData(x1, y1, z1, x2, y2, z2):
6      xhigh = max(x1, x2)
7      xlow = min(x1, x2)
8      yhigh = max(y1, y2)
9      ylow = min(y1, y2)
10     zhigh = max(z1, z2)
11     zlow = min(z1, z2)
12
13     blocks = []
14     for x in range(xhigh - xlow + 1):
15         blocks.append([])
16         for y in range(yhigh - ylow + 1):
17             blocks[x].append([])
18             for z in range(zhigh - zlow + 1):
```

```
19                 blocks[x][y].append([])
20                 block = mc.getBlockWithData(xlow +
                      ↪ x, ylow + y, zlow + z)
21                 blocks[x][y][z] = block
22     return blocks
23
24 prompt = raw_input("Move to the first corner and
      ↪ press enter in this window")
25 pos1 = mc.player.getTilePos()
26
27 prompt = raw_input("Move to the opposite corner
      ↪ and press enter in this window")
28 pos2 = mc.player.getTilePos()
29 print "Please wait"
30
31 structure = getBlocksWithData(pos1.x, pos1.y,
      ↪ pos1.z, pos2.x, pos2.y, pos2.z)
32
33 print "Move to the location you want to place the
      ↪ structure"
34 prompt = raw_input("and press enter in this
      ↪ window.")
35 pos3 = mc.player.getTilePos()
36
37 for x in range(len(structure)):
38     for y in range(len(structure[x])):
39         for z in range(len(structure[x][y])):
40             block = structure[x][y][z]
41             blockType = block.id
42             state = block.data
43             if blockType == 4:
44                 blockType = 49
45             elif blockType == 9 or blockType == 8:
46                 blockType = 10
47             mc.setBlock(pos3.x + x, pos3.y + y,
                  ↪ pos3.z + z, blockType, state)
```

Chapter 9

Loops

9.1 Midas Touch

```
1  import mcpi.minecraft as minecraft
2  import time
3  mc = minecraft.Minecraft.create()
4
5  gold = 41
6  water = 9
7  air = 0
8
9  while True:
10     pos = mc.player.getTilePos()
11     blockBelow = mc.getBlock(pos.x, pos.y - 1,
          ↪ pos.z)
12     if blockBelow != air and blockBelow != water:
13         mc.setBlock(pos.x, pos.y - 1, pos.z, gold)
14     time.sleep(0.1)
```

9.2 Tree Fighter

```
1  import mcpi.minecraft as minecraft
2  import time
3  mc = minecraft.Minecraft.create()
4
5  woodHits = 0
6  woodBlock = 17
7
```

```
8 mc.postToChat("Fight Trees!")
9
10 while woodHits < 20:
11     hits = mc.events.pollBlockHits()
12     for hit in hits:
13         blockType = mc.getBlock(hit.pos.x,
               ↪ hit.pos.y, hit.pos.z)
14         if blockType == woodBlock:
15             woodHits += 1
16             mc.postToChat("You have hit " +
                   ↪ str(woodHits) + " trees")
17     time.sleep(0.1)
18 else:
19     mc.postToChat("Congratulate! Challenge
           ↪ Complete!")
```

9.3 Chat with a Loop

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 userName = raw_input("Enter a user name: ")
5 mc.postToChat(userName + " has entered chat")
6 message = raw_input("Message: ")
7
8 while message != "/exit":
9     mc.postToChat(userName +": " + message)
10     message = raw_input("Message: ")
11 else:
12     mc.postToChat(userName + " has left the chat")
```

9.4 Pyramid

```
1 import mcpi.minecraft as minecraft
2 mc = minecraft.Minecraft.create()
3
4 pos = mc.player.getTilePos()
5 x = pos.x + 2
6 y = pos.y
7 z = pos.z
8
```

```
base = 29
height = 0
sandstone = 24

while base - (2 * height) > 0:
    for block in range(base - (2 * height)):
        for row in range(base - (2 * height)):
            blockx = x + block + height
            blocky = y + height
            blockz = z + row + height
            mc.setBlock(blockx, blocky, blockz,
               ↪ sandstone)
    height += 1
```

9.5 Hot and Cold

```
import mcpi.minecraft as minecraft
import math
import time
import random
mc = minecraft.Minecraft.create()

tileX = random.randint(-127, 127)
tileZ = random.randint(-127, 127)
tileY = mc.getHeight(tileX, tileZ)

diamond = 57
block = diamond
mc.setBlock(tileX, tileY, tileZ, diamond)
mc.postToChat("Block set")

while block == diamond:
    pos = mc.player.getPos()
    playerX = pos.x
    playerY = pos.y
    playerZ = pos.z
    twoDimension = math.sqrt((playerX - tileX) **
       ↪ 2 + (playerZ - tileZ) ** 2)
    threeDimension = math.sqrt((playerY - tileY)
       ↪ ** 2 + twoDimension ** 2)

    if threeDimension > 180:
```

```
25          mc.postToChat("Freezing")
26      elif threeDimension > 90:
27          mc.postToChat("Cold")
28      elif threeDimension > 30:
29          mc.postToChat("Warm")
30      elif threeDimension > 15:
31          mc.postToChat("Warmer")
32      elif threeDimension > 5:
33          mc.postToChat("On fire!")
34      block = mc.getBlock(tileX, tileY, tileZ)
35      time.sleep(0.5)
36  else:
37      mc.postToChat("Found it")
```

9.6 Adapt Exercises

Chapter 10

Advanced Topics in Python

Chapter 11

Binary and Bitwise Operators

Chapter 12

Classes

Chapter 13

File input and Output