Python Games

Session 1

By Declan Fox

Rules

“Above all, be cool.”

General Information

Wi-Fi Name: CoderDojo

Password: coderdojowireless

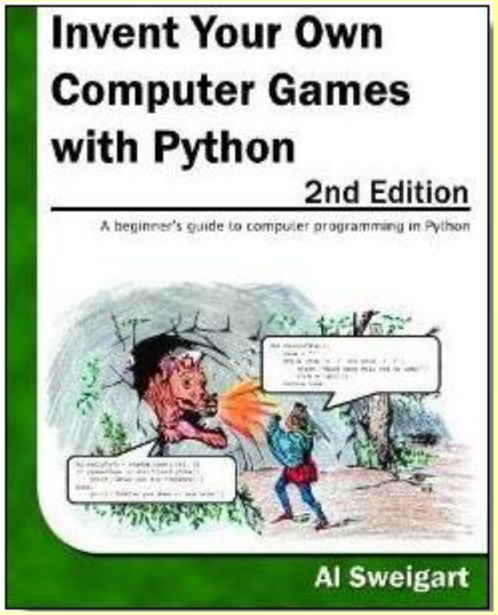
Website: http://cdathenry.wordpress.com/

Plans for this year

- Command line interface at first
  - Number guessing game
- GUI games with Pygame
  - Bunnies and badgers
- Possibly Hardware
  - Lego Mindstorms

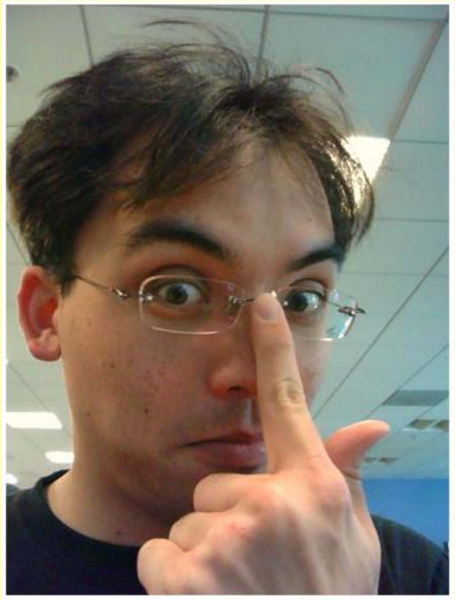
Sources

We will be using some code from the book "Invent your own computer games with Python" by Al Sweigart.


Invent Your Own Computer Games with Python
2nd Edition
A beginner's guide to computer programming in Python
Al Sweigart

Al Sweigart

Albert Sweigart (but you can call him Al), is a software developer in San Francisco, California he is originally from Houston, Texas.



The book is free online at

http://inventwithpython.com

Installation

As we will be moving on to graphical games we will need to install both Python and Pygame*

* If you have Python 3.x.x and Pygame installed you can ignore the next two slides

Installation

We will be using version 3.2 of Python as this is latest release that's compatible with Pygame

Go to

https://www.python.org/download/releases/3.2.5/

Scroll down to the list of downloads

Click on Windows x86 MSI Installer (3.2.5)

When this has downloaded click on it to install.

Installation

To install Pygame

Go to

http://pygame.org/download.shtml

Scroll down to the list of downloads

Click on

pygame-1.9.2a0.win32-py3.2.msi

When this has downloaded click on it to install.

Programming Languages

Input, Output & Store Data

- E.g. text, numbers

Operate on Data

- E.g. add numbers, change text

Loops

- Repeat commands several times

Decisions

- Do something IF something else is true

IDLE

Stands for **I**nteractive **D**eve**L**opment **E**nvironment.

The Interactive Shell

The window that appears when you first run IDLE is called the interactive shell.

A shell is a program that lets you type instructions into the computer.

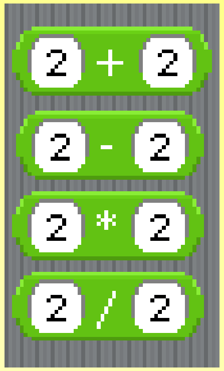
Exercise

Try typing some of these math problems into the shell, pressing Enter key after each one.

```
2+2+2+2+2
8*6
10-5+6
2  +      2
10 / 2
```

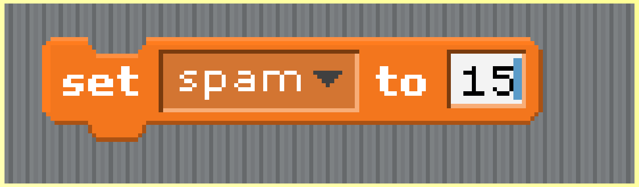
Math Operators

2 + 2	addition
2 - 2	subtraction
2 * 2	Multiplication
2 / 2	division



Storing Values in Variables

spam = 15


```
set spam to 15
```

types of numbers in Python

int which is an integer example 3, 4, 7

float which is a floating point number 3.4, 7.2, 5.0

What is a Variable?

In computer programming, a variable is a storage **location** and an associated **symbolic name** (an identifier) which contains some known or **unknown quantity** or information, a value.

Strings

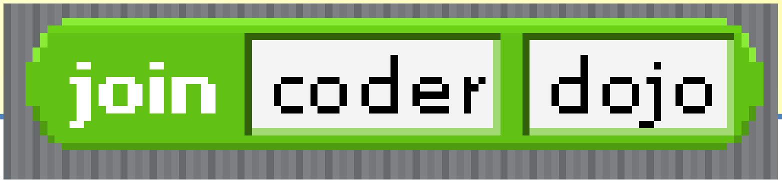
Another important data type is a string which are pieces of text.

spam = ‘hello’

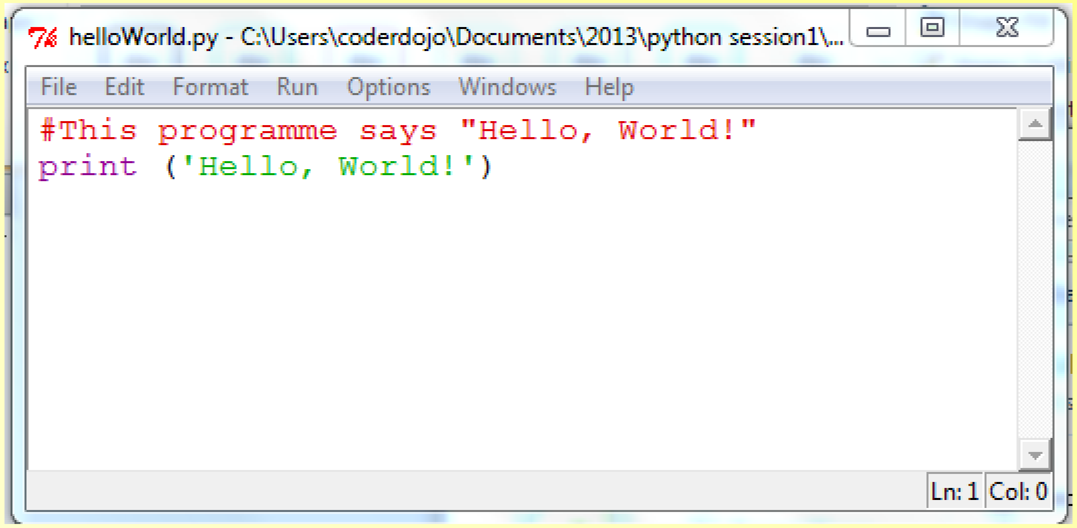
String concatenation

Complicated way of saying you can join string together using the + operator.

‘coder’ + ‘dojo’ will give you ‘coderdojo’



Our first program


helloWorld.py - C:\Users\coderdojo\Documents\2013\python session1\...
File Edit Format Run Options Windows Help
Ln: 1 Col: 0

```python
#This programme says "Hello, World!"
print ('Hello, World!')
```

The File Editor

Let's write our first program!

To write a programme we need to use the IDLE file editor.

Click on the File menu at the top of the Python Shell window, and select New Window. A new blank window will appear for us to type our program in.

This window is the file editor.

Comments

```
#This programme says "Hello, World!“
```

This line is a comment

Comments are used to explain the code to yourself or to anybody else who looks at it

Print Function

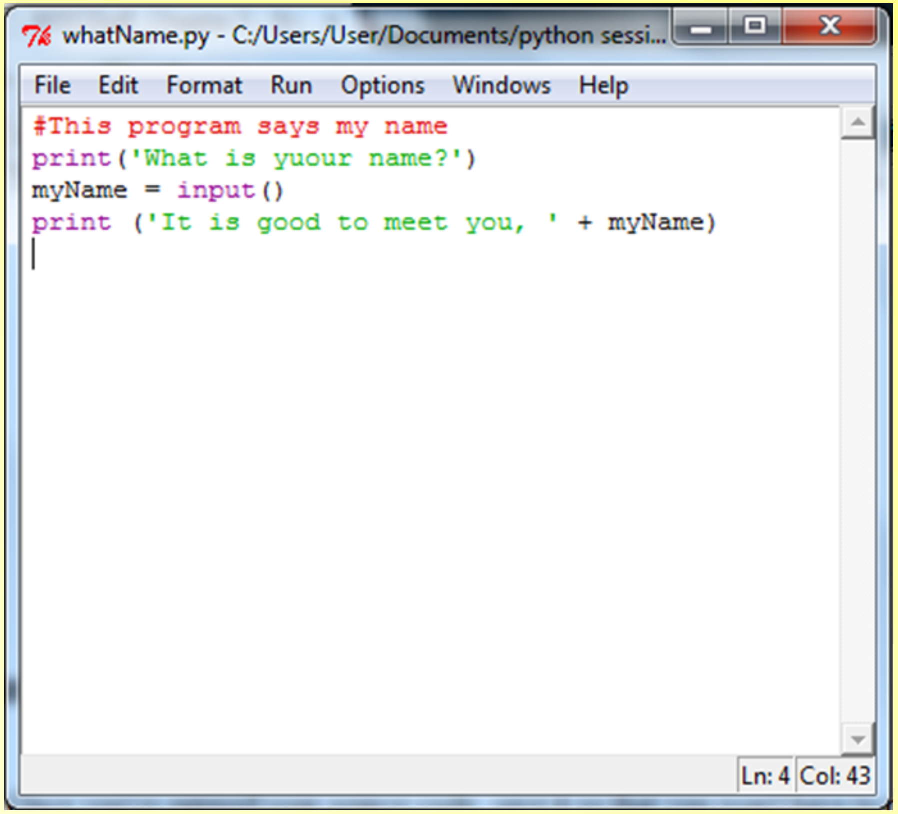
```
print ('Hello, World!‘)
```

This line calls the print function

Whatever is to be printed goes inside the parentheses

We use parentheses to make it clear we are talking about a function called print not a variable called print

Input


whatName.py - C:/Users/User/Documents/python sessi...
File Edit Format Run Options Windows Help
Ln: 4 Col: 43

```
#This program says my name
print('What is yuour name?')
myName = input()
print ('It is good to meet you, ' + myName)
```

Input

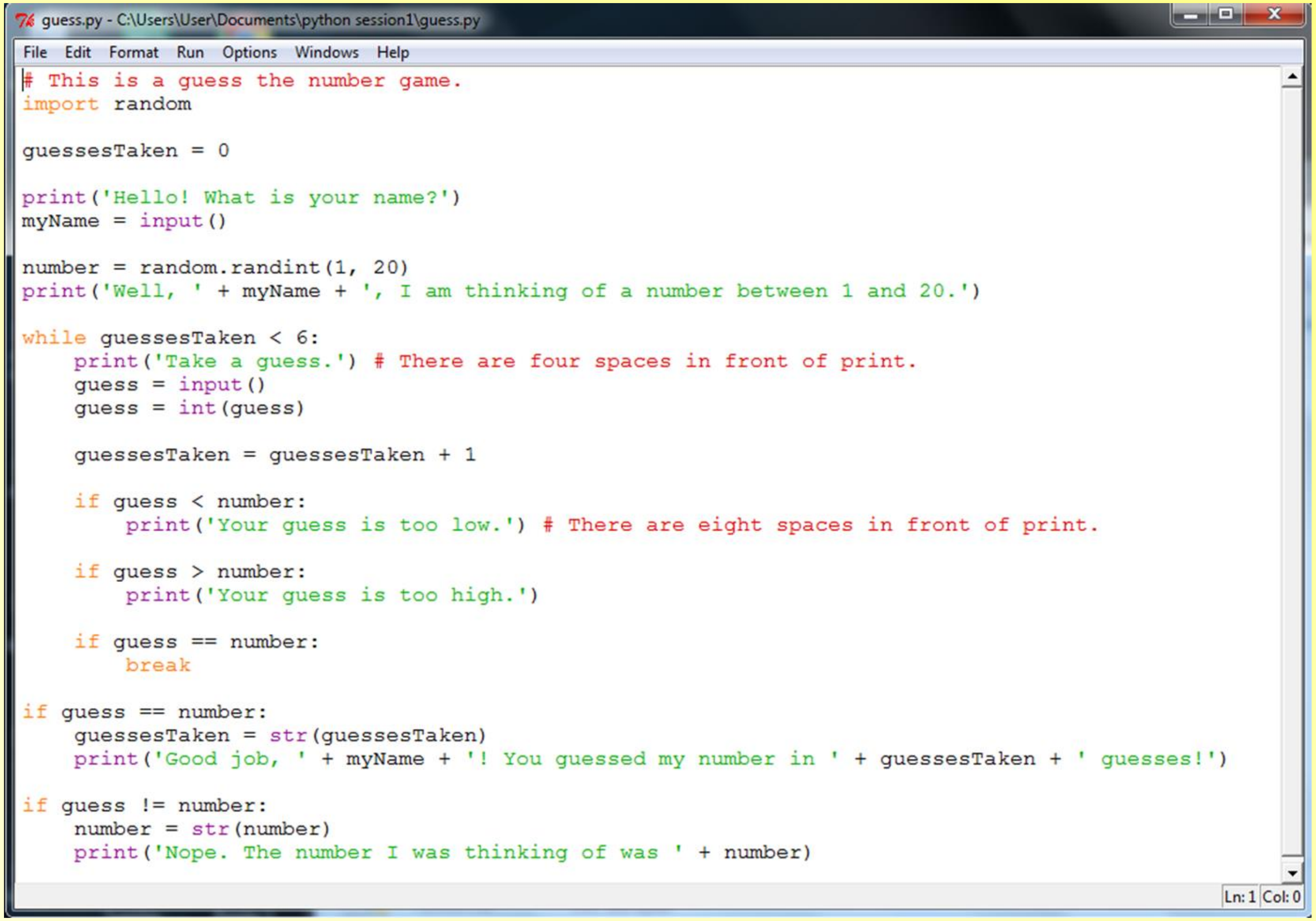
```
myName = input()
```

This line has an assignment statement with a variable myName and a function call input()

When input() is called, the program waits for the user to enter text.

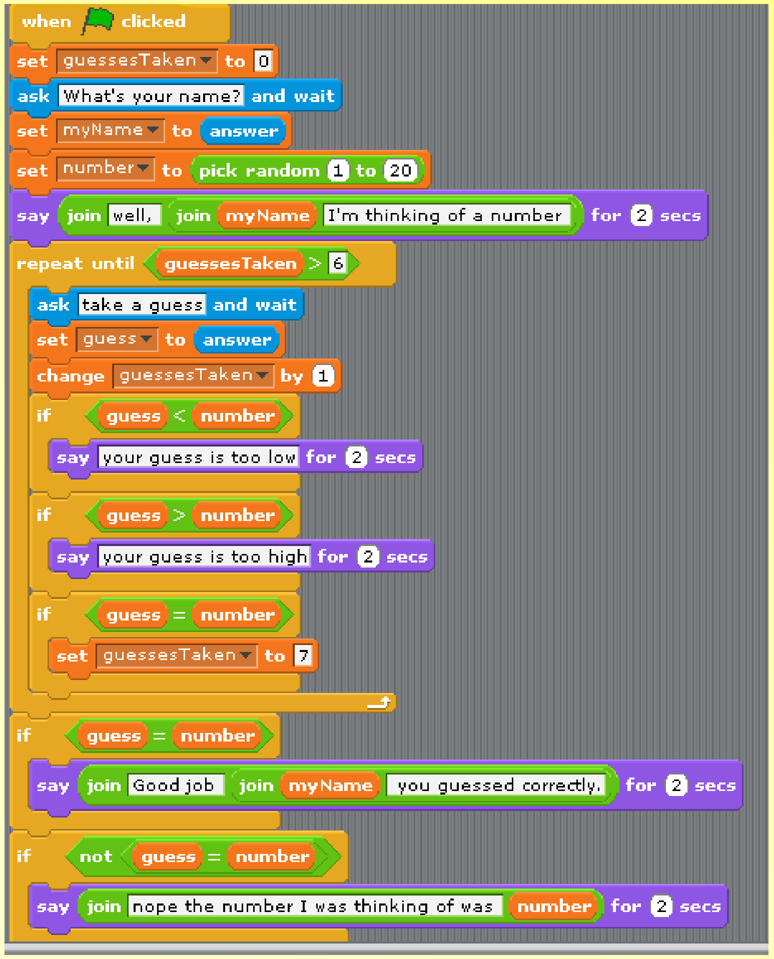
The text string that the user enters (your name) becomes the function's output value

Guess the number game


guess.py - C:\Users\User\Documents\python session1\guess.py
File Edit Format Run Options Windows Help
Ln: 1 Col: 0

```
# This is a guess the number game.
import random

guessesTaken = 0

print('Hello! What is your name?')
myName = input()

number = random.randint(1, 20)
print('Well, ' + myName + ', I am thinking of a number between 1 and 20.')

while guessesTaken < 6:
    print('Take a guess.') # There are four spaces in front of print.
    guess = input()
    guess = int(guess)

    guessesTaken = guessesTaken + 1

    if guess < number:
        print('Your guess is too low.') # There are eight spaces in front of print.

    if guess > number:
        print('Your guess is too high.')

    if guess == number:
        break

if guess == number:
    guessesTaken = str(guessesTaken)
    print('Good job, ' + myName + '! You guessed my number in ' + guessesTaken + ' guesses!')

if guess != number:
    number = str(number)
    print('Nope. The number I was thinking of was ' + number)
```

Scratch Version



```
when [flag] clicked
set guessesTaken to 0
ask What's your name? and wait
set myName to answer
set number to pick random 1 to 20
say join well, join myName I'm thinking of a number for 2 secs
repeat until guessesTaken > 6
    ask take a guess and wait
    set guess to answer
    change guessesTaken by 1
    if guess < number
        say your guess is too low for 2 secs
    if guess > number
        say your guess is too high for 2 secs
    if guess = number
        set guessesTaken to 7
if guess = number
    say join Good job join myName you guessed correctly. for 2 secs
if not guess = number
    say join nope the number I was thinking of was number for 2 secs
```

Import Statement

```
import random
```

This line is an **import statement**

Python includes many built-in functions.

Modules are Python programs that contain additional functions.

In this case, we're importing the random module.

Random Function

```
number = random.randint(1, 20)
```

In this line we call a new function named randint(), and then store the return value in a variable named number.

Because the randint() function is provided by the random module, we precede it with random. (don't forget the dot) to tell our program that the function randint() is in the random module.

Passing Arguments to Functions

The numbers in the parentheses in the random.randint(1, 20) function call are called arguments.

Arguments are the values that are passed to a function when the function is called. Arguments tell the function how to behave.

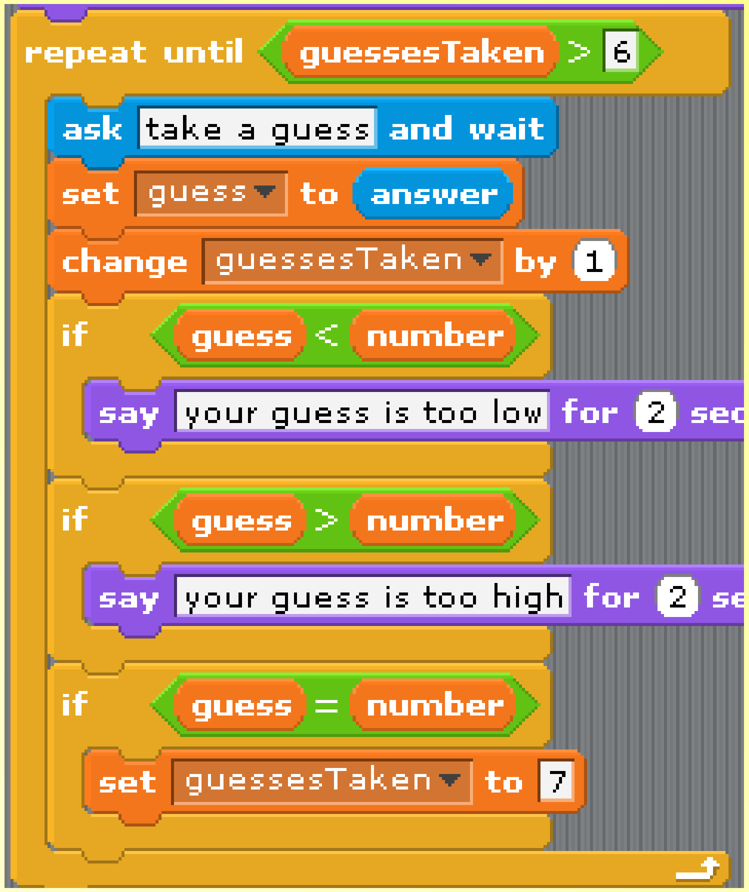
Some functions require that you pass them values when you call them.

Loops

Line 12 has something called a while statement, which indicates the beginning of a while loop.

Loops are parts of code that are executed over and over again.

Loop in Scratch



```
repeat until guessesTaken > 6
    ask take a guess and wait
    set guess to answer
    change guessesTaken by 1
    if guess < number
        say your guess is too low for 2 sec
    if guess > number
        say your guess is too high for 2 se
    if guess = number
        set guessesTaken to 7
```

Code Blocks

A **block** is one or more lines of code grouped together with the same amount of indentation. You can tell where a block begins and ends by looking at the line's **indentation** (that is, the number of spaces in front of the line).

A block begins when a line is indented by four spaces. Any following line that is also indented by four spaces is part of the block.

Blocks within Blocks

A block within a block begins when a line is indented with another four spaces (for a total of eight spaces in front of the line). The block ends when there is a line of code with the same indentation before the block started

Blocks within Blocks

```
while guessesTaken < 6:
    print('Take a guess.') # There are four spaces in front of print.
    guess = input()
    guess = int(guess)

    guessesTaken = guessesTaken + 1

    if guess < number:
        print('Your guess is too low.') # There are eight spaces in front of print.

    if guess > number:
        print('Your guess is too high.')

    if guess == number:
        break
```

The Boolean Data Type

The Boolean data type has only two values

True or **False**.

These values are case-sensitive

They are not string values; do not put a ' quote character around them.

We use Boolean values (also called bools) with comparison operators to form conditions.

Comparison Operators

```
while guessesTaken < 6:
```

The comparison operator is used to compare two values and evaluate to a True or False Boolean value.

Comparison Operators

Comparison operators	
Operator Sign	**Operator Name**
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Is equal to
!=	Not equal to

Conditions

A condition is an expression that combines two values with a comparison operator

(such as < or >)

and evaluates to a Boolean value.

A condition is just another name for an expression that evaluates to True or False.

Conditions

<u>Conditions always evaluate to a Boolean value True or False.</u>

In the case of our Guess the Number program, in line 4 we stored the value 0 in guessesTaken. Because 0 is less than 6, this condition evaluates to the Boolean value of True.

Remember, a condition is just a name for an expression that uses comparison operators such as < or !=.

The int() Function

```
guess = int(guess)
```

In line 15, we call a new function called int().

The input() function returned a string of text that player typed. But in our program, we will want an integer, not a string. When the player enters their guess, the input() function will return the string value '5' and not the integer value 5.

The int() Function

Remember that Python considers the string '5' and the integer 5 to be different values.

So the int() function will convert the string value we give it and return the integer value of it.

Incrementing a Variable

```
guessesTaken = guessesTaken + 1
```

The if statement

Lines 19 and 20 check if the number that the player guessed is less than the secret random number that the computer came up with.

If so, then we want to tell the player that their guess was too low by printing this message to the screen.

```
if guess < number:
print('Your guess is too low.')
```

The break Statement

```
if guess == number:
    break
```

The line inside the if-block is a **break** statement that tells the program to immediately jump out of the while-block to the first line after the end of the while-block.

Next Week

Ideas for improving our guessing game and we’re going to look at an adventure game called Dragon’s Lair