All programming languages have

- Sequence
- Selection
- Iteration

**Sequence** in Python is pretty easy. Things that occur first in the script are executed first. So, in the following code fragment, **dog** is defined *before* **cat.**

```
            dog=”Spot”
            cat=”Fluffy”
```

To follow along, type the following into IDLE:

```
from system import *

dog=”Spot”
cat=”Fluffy”
```

The first line is called an **import statement**. We use import statements to tell Python to find resources it needs to work. We usually put `from system import *` at the top of all of our programs. You can have as many import statements as you want, but need to import content before you can use it.

**Selection** describes how you separate one set of things from another. In Python, we use if statements. If statements work like this:

```
            if dog==”Spot”:
                  print(“The dog’s name is Spot.”)
            else:
                  print(“The dog’s name is not Spot.”)
```

Here, we use a **print statement** inside of our if statements. Print statements ‘print’ words to the output terminal on your screen. Try it!

To follow along, your code in IDLE should look like this:

```
from system import *

dog=”Spot”
cat=”Fluffy”
if dog==”Spot”:
      print(“The dog’s name is Spot.”)
else:
      print(“The dog’s name is not Spot.”)
```

**Iteration** is a way of telling the computer to keep on doing something, without having to repeat our commands. Python has **two** different ways to iterate through things. Usually, you can use either way. One of the ways is often easier than the other, though.

A **while loop** is a statement that tells Python to continue doing something while a particular condition is true. For example:

```
while dog==”Spot”:
      cat=”Fluffy”
```

Because `dog` is always `Spot`, the loop will execute forever. You can **manually stop** a program from executing with the Ctrl+C command (in Windows and most Linux OSs, not sure about Macs).

Here’s an example of a while loop that won’t execute forever:

```
while cat==”Fluffy”:
      cat=”Wet”
```

Because we changed `cat` from `Fluffy` to `Wet` inside of the loop, this loop will only execute (run) once. Then, the `cat` will be `Wet`, and when it tries to execute again, it won’t be able to.

A **for loop** is a statement that tells python to keep on doing something for as many times as you tell it to.

```
for num in range(6):
      print(num)
```

You probably won’t need to use these very often in this class. They’re good for when you only want to do something a certain number of times.

Some useful functions in Python:

**Print** function

```
print(“Print the ball’s velocity ”, ball.velocity)
```

**Float** function: use this function to turn a string (usually a word) into a float (a decimal number).

```
number=float(“456”)
```