While Loops

- Iteration statements are critical to a functioning ‘while loop.’ Iteration statements result in the statements within the iteration are executed repeatedly. The amount of times the code is looped depends on the iteration statement parameters.
- While loops are associated with bool expressions, because they execute and loop a code block when the bool statement is true. Additionally, the bool expression is assessed before the code body within the expression is executed.
- public bool canWalk = true;

```
void Start () {
        if(canWalk) {
                StartCoroutine(GoWalk());
        } else {
                print(“Learn to walk doofus”);
        }
}



IEnumerator GoWalk () {

        while (canWalk) {
                print(“Walking”);
                yield return new WaitForSeconds(0.1f);
        }
        print(“Planking”);
    }
```

- Do-while loops are another type of iteration statement, and they are very similar to while loops. The only difference between them and while loops is that do-while loops check the expression after the code block is executed. This results in the statements within the iteration statement being executed at least one.

-
  ```
  public bool canBlink = false;
  public Text input;
  public string blink = “I can blink”;


      void Start () {
          if(input.text == blink) {
          do {
              StartCoroutine(BlinkEyes());
              canBlink = true;
          }
          } else {
              print(“uhhhh you can’t blink? You weird lol”);
          }
      }
      IEnumerator  BlinkEyes () {
          while(canBlink) {
  ```

```
            print(“Blinking. Good Job!”);
            yield return new WaitForSeconds(0.1f);
    }
    print(“are you watching clockwork orange?”);
}
```

- For loops are another version of iteration statements. These loops contain particular clauses that change the initialization and iteration of a looped code block. Other than that, they are the same as while loops. The clauses include the init, condition, and iteration clauses.
-
  ```
  public int example = 4;
      void Update () {
          while (example < 8) {
              print(example += 1);
          }
  }
  ```
-
  ```
  public Text input;
  public string typeThis = “this”;
  void Start () {
      StartCoroutine(TypeStuff());
  }
  IEnumerator TypeStuff () {
      while(text.input == typeThis) {
          print(“this. haha get it?”);
  ```

```
        yield return new WaitForSeconds(0.1f);
    }
    print(“this is confusing”);
}
```

- Iteration code blocks must always return something, unlike void functions. If you don’t return something, you will err out.
- public bool sayHi = true;

```
void Start () {
    StartCoroutine(SayGreeting());
}
IEnumerator SayGreeting () {
    while (sayHi) {
        print(“Hi!”);
        return null;
    }
    print(“Rude!”);
}
```