Parables and Pythons

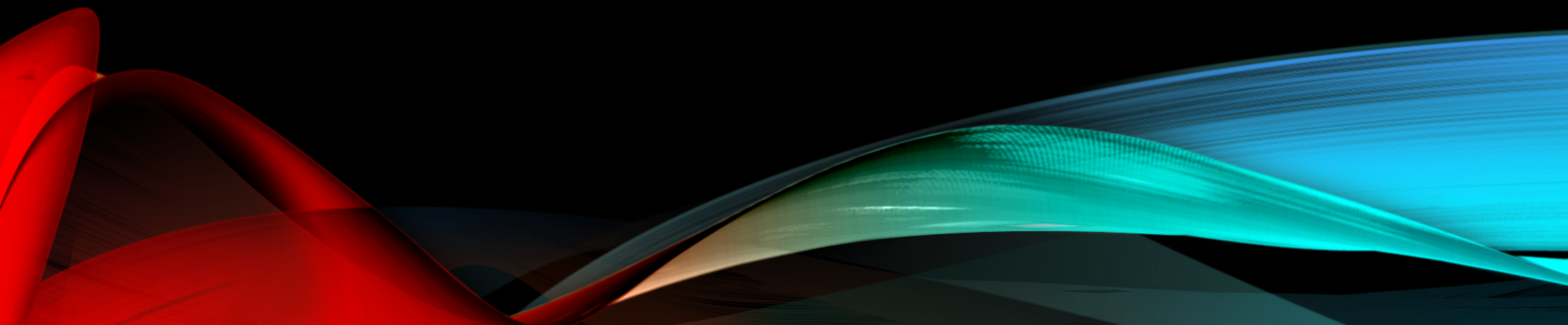
Dr. Charles “Chuck” Bell

Lesson 5: 10 October 2018

CLASS AGENDA

- Bible Study
  - Unmerciful Servant
- The Golden Age of British Comedy
  - It's an ex-parrot!
- Computer Programming with Python
  - New Concepts:
    - (more) Loop Statements
    - Lists
  - Hands-On Practice
    - While loop, lists
- Homework

BIBLE STUDY: THE PARABLES OF JESUS



MATHEW 18:21-33

Then Peter came to Jesus and asked, "Lord, how many times shall I forgive my brother or sister who sins against me? Up to seven times?" Jesus answered, "I tell you, not seven times, but seventy-seven times.

"Therefore, the kingdom of heaven is like a king who wanted to settle accounts with his servants. As he began the settlement, a man who owed him ten thousand bags of gold was brought to him. Since he was not able to pay, the master ordered that he and his wife and his children and all that he had be sold to repay the debt. "At this the servant fell on his knees before him. 'Be patient with me,' he begged, 'and I will pay back everything.' The servant's master took pity on him, canceled the debt and let him go.

THE UNMERCIFUL SERVANT

- Most people see or interpret this parable as how many times we are to forgive someone, but there is far more to the story.
- If you remember this parable, it begins with Peter asking Jesus a seemingly simple question.
- Most people stop reading there, but it goes on to describe a servant who has his debts forgiven but fails to forgive someone who owes him money.
- Furthermore, most see the servant as representing the Israelites, who felt they were above reproach on financial matters and would never consider not paying their debts or expect punishment in the form of torture for violations of debt agreements – it just wasn't done.
- If you listen carefully, you can detect an absurdity that would make it an unbelievable story.
- In fact, there is a dichotomy in the story, which some believe is a contradiction.
- But the dichotomy is intentional. See if you can spot it.

MATHEW 18:21-33

"But when that servant went out, he found one of his fellow servants who owed him a hundred silver coins. He grabbed him and began to choke him. 'Pay back what you owe me!' he demanded. "His fellow servant fell to his knees and begged him, 'Be patient with me, and I will pay it back.' "But he refused. Instead, he went off and had the man thrown into prison until he could pay the debt. When the other servants saw what had happened, they were outraged and went and told their master everything that had happened. "Then the master called the servant in. 'You wicked servant,' he said, 'I canceled all that debt of yours because you begged me to. Shouldn't you have had mercy on your fellow servant just as I had on you?' In anger his master handed him over to the jailers to be tortured, until he should pay back all he owed.

"This is how my heavenly Father will treat each of you unless you forgive your brother or sister from your heart."

WHAT JUST HAPPENED?

- We see that Peter's question is indeed answered as not just 7, but 70 times 7.
- We also see that Jesus tells a related story about a king who has a subject that owes a lot of money, but the king has mercy on him and grace by dismissing the dept.
- However, we see right after that the subject then goes to his debtor, who owes far less, and does not forgive the debt.
- Clearly, this is a warning and it seems the parable is quite straightforward. Or is it?

ANALYSIS OF THE NUMBERS IN THE PARABLE

- The number 7 means perfection or holy completeness. Thus, to forgive someone 7 times is to completely forgive.
- The number 70 times 7 is a word picture for mathematical infinity. Thus, Jesus says to forgive someone as many times as necessary. In other words, always.
- Consider the bags of gold. There were 10,000 of them!
  - It has been suggested Herod's budget for Israel at the time was about 600 talents.
  - If each bag contained a talent, which is roughly 6,000 denarii an each denarii about $48 today, that could equate to over $3 billion. Clearly, this sum of money was an absurd amount of money!
  - With this in mind, the story becomes one of a proof by absurdity. Imagine what would happen to our economy if the government forgave everyone's' tax debt.
- The servant's debt he did not forgive was likely more like $3,000.00 today – something most people with even modest incomes could pay eventually.

LET'S REVIEW

- Given the huge amount of money, is the king in the story really trying to collect or does he know there is no hope for the debtor other than mercy and grace?
- Suppose you loaned someone a lot of money. Would you be so easy to forgive the debt if they begged you?
- If you borrowed money from a bank, would you think the bank manager would be lenient? Even it he were, would you expect cancellation of the debt?
- What about the servant? Was he seeing his debtor in a different light? Did he think the smaller sum accountable but his own insurmountable debt not accountable?

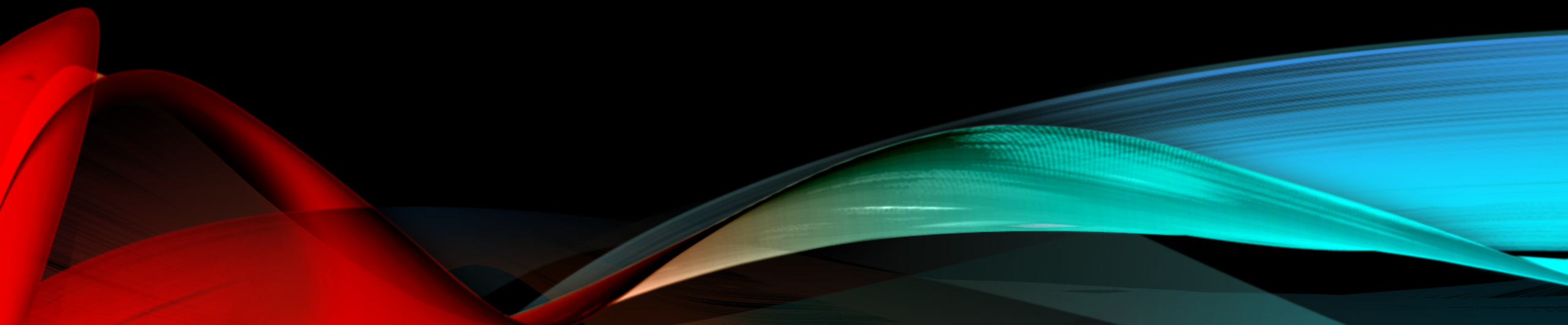
THE DICHOTOMY

- There are two things going on here that seem to be at odds. Forgiveness without consequences, without end and punishment for not forgiving someone
- There is the instruction by Jesus for us to forgive each other "infinitely" (70 times 7).
- There is the king in the story forgiving his servant (the $3B was meant to represent an insurmountable debt).
- There is the servant **not** forgiving his debtor.
- So, this gives us the dichotomy that while we expect God to forgive us, if we do not forgive each other, the consequences are torture.
- Worse, the parable suggests that God will punish all if one of us doesn't forgive another.
  - *"This is how my heavenly Father will treat each of you unless you forgive your brother or sister from your heart."*
  - This is a warning that should be heeded.

CONCLUSIONS

- Most assume the parable is a reminder and a warning that we often don't show the same mercy to others as God shows to us - and it is – but it is more than that!
- The parable uses the absurdity – more money than anyone could fathom – to show the value of God's mercy and grace compared to our own.
- More importantly, we should strive to live in the full-time mercy of God by giving it back to others.
- The message is simply this: the way God forgives is forever. Forever. We should strive to do likewise.

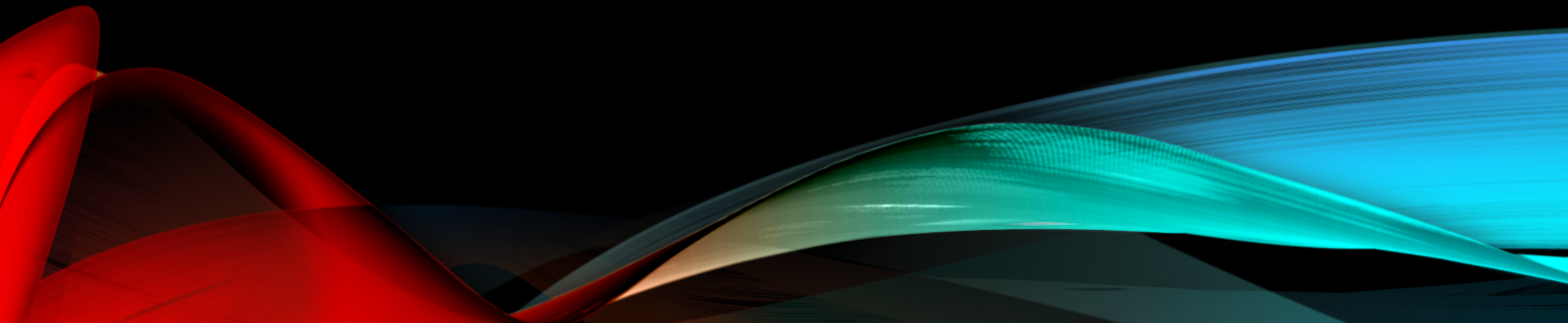
THE GOLDEN AGE OF BRITISH COMEDY



- https://www.youtube.com/watch?v=CIrBMt4eiRk

COMPUTER PROGRAMMING

Hands On Learning



FOR LOOP - REVIEW

- We may need to execute a set of statements several times.
- We have the “for” loop for executing a “for each value in” statement.
- Example: suppose we want to execute a code block 10 times.
- The for loop goes like this:

```
for <variable> in range(<start>, <end>):
    <code to execute>
```

- Note: the range() function end is exclusive. So, if we want to do something 10 times, using (1, 10) won’t work. Why?

LOOPS: THE WHILE LOOP

- Another type of loop is the while loop.
- The loop uses a condition such that as long as the expression evaluates to True, it will execute the body (code block) of the loop.
- The while loop goes like this:

```
while <expression>:
    <code to execute>
```

- The expression can be any combination of inequalities and conditionals.

WHILE LOOP – HANDS ON

- This example is the home work challenge for the dice simulator.
- Open the dice simulator (dice_simulator1.py) and edit it as shown.

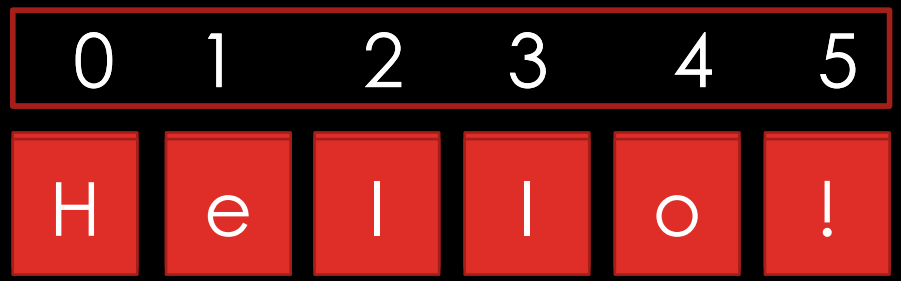
```
import random
repeat = 'y'
while (repeat == 'y') or (repeat == 'Y'):
    print("Welcome to the dice simulator!")
    numSides = int(input("What size dice do you want to use? "))
    if (numSides % 2) == 1:
        print("ERROR: you must use an even number!")
    else:
        numDice = int(input("How many dice do you want to roll? "))
        for i in range(0,numDice):
            diceVal = random.randint(1,numSides)
            print("Roll: {0}".format(diceVal))
    print()
    repeat = input("Roll again? [Y/N] ")
    print()
print("bye!")
```

WHILE LOOP - EXECUTION

```
$ python3 ./dice_simulator2.py
Welcome to the dice simulator!
What size dice do you want to use? 6
How many dice do you want to roll? 4
Roll: 3
Roll: 3
Roll: 6
Roll: 1

Roll again? [Y/N] y

Welcome to the dice simulator!
What size dice do you want to use? 20
How many dice do you want to roll? 3
Roll: 20
Roll: 4
Roll: 4

Roll again? [Y/N] n
bye!
```

DATA OBJECTS IN PYTHON

- So far, we've only used variables, which hold exactly one value.
- What if you need to store more than one value?
- Let's start with the basics: a list.
  - The simplest list is one we've already used: a character string!
  - A list is a contiguous segment of memory that stores a set of values.
  - Lists are "counted" or "indexed" starting at 0.
  - List indexes use the [n] syntax where n is the index (number).
  - A list can hold any number of values.
  - Lists have additional methods that we can use to manipulate the data (more on that next week)



LISTS – HANDS ON EXAMPLE

- Open an editor and enter the following code.

```
# Working with lists.
myString = "Hello, World!”
# Example 1: print the list one character at a time.
for i in range(0, len(myString)):
    print(myString[i], end='')
print()
```

LISTS – HANDS ON EXERCISE

- We can also access parts of the list using special subscript notation.
- myString[<start>:<end>]
- Add these new examples to your file and execute them.

```
# Example 2: print the 3 characters in the middle of the string
num = int(len(myString) / 2)
print(myString[num-1:num+2])
# Example 3: print the first word
print(myString[:5])
# Example 4: print the last word with start to end of string
print(myString[7:])
```

DECLARING LISTS

- Strings are a special case of a list.
- To define a list in Python, say of integers, the [] syntax is as follows.
- Type this into your editor and run the example (list1.py)

```
# Example 5: integer list
myInts = [1,2,3,4,5,6]
print(myInts)
for i in range (0,3):
    print(myInts[i], end=' ')
print()
```

MORE DATA STRUCTURES

- lists are great, but what if you need to store more than one value as a set of data?
  - Example: first name, last name or address (street, city, state, zip)
- Python has other, very powerful data structures (ways to store data).
- These include:
  - Lists
  - Tuples
  - Set
  - Dictionary
- We'll talk about the first three tonight and learn more about them next week.

TUPLES

- A number of values separated by commas
- Immutable
  - Cannot assign values to individual items of a tuple
  - However, tuples can contain mutable objects such as lists

```
>>> t = 12345, 54321, 'hello!'
>>> t[0]
12345
>>> t
(12345, 54321, 'hello!')
>>> # Tuples may be nested:
... u = t, (1, 2, 3, 4, 5)
>>> u
((12345, 54321, 'hello!'), (1, 2, 3, 4, 5))
```

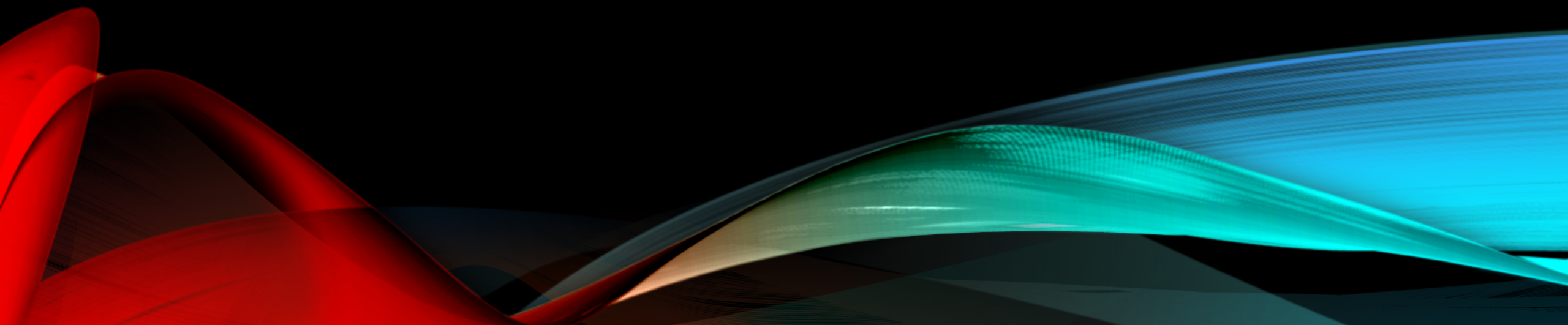
SETS

- An unordered collection with no duplicate elements
- Sets cannot be changed and are often used as lookup or in expressions.
  - Example: (myFruit in myFavoriteFruits)

```
myBasket = {'apple', 'orange', 'pear'}
print(myBasket)
if ('kiwi' not in myBasket):
    print("Sorry, no kiwi for you.")
```

HOMEWORK

All homework assignments can be handed in on hardcopy (with your name at the top) or emailed to me at drcharlesbell@gmail.com.



HOMEWORK #3 - REVIEW

- This is the correct solution to the modified dice simulator assignment:

```
import random
repeat = 'y'
while (repeat == 'y') or (repeat == 'Y'):
    print("Welcome to the dice simulator!")
    numSides = int(input("What size dice do you want to use? "))
    if (numSides % 2) == 1:
        print("ERROR: you must use an even number!")
    else:
        numDice = int(input("How many dice do you want to roll? "))
        for i in range(0,numDice,2):
            diceVal1 = random.randint(1,numSides)
            print("Roll: {0}".format(diceVal1), end='')
            if (i < numDice-1):
                diceVal2 = random.randint(1,numSides)
                print(", {0}".format(diceVal2))
            else:
                print()
    print()
    repeat = input("Roll again? [Y/N] ")
    print()
print("bye!")
```

HOMEWORK #3 - REVIEW

```
$ python3 ./dice_simulator.py
Welcome to the dice simulator!
What size dice do you want to use? 6
How many dice do you want to roll? 4
Roll: 5, 6
Roll: 6, 4

Roll again? [Y/N] y

Welcome to the dice simulator!
What size dice do you want to use? 6
How many dice do you want to roll? 5
Roll: 5, 5
Roll: 4, 3
Roll: 4

Roll again? [Y/N] n

bye!
```

HOMEWORK ASSIGNMENT #4

- (2 points) Make a Yahtzee simulator to roll (5) 6-sided die at one time.
  - https://en.wikipedia.org/wiki/Yahtzee
  - Calculate all of the random values saving each to a list.
  - Print the list of values.
  - Hint: you need only one for loop.
  - Hint:

```
diceRolls = [0,0,0,0,0]
diceRolls[3] = 99
print(diceRolls)
```

- (2 bonus points) Print out the dice rolls in ascending order (hint: you will need to sort the list).

QUESTIONS OR COMMENTS?