Using list comprehensions and/or mapping and/or zip [Reading 7], write Python code to:

1. Create a list x containing 10 values equally spaced from 0 to 2π inclusive.
2. Create a second list y containing values of .
3. Create a third list z that is True if y > 1.5 and false otherwise.
4. Print a table of values containing x and y, formatted in columns 10 characters wide with 5 decimal places, and z, formatted in

columns 5 characters wide. You can use either a string or an integer format for the booleans. x sin2(3x/π) Exercise 1: list comprehensions, mapping, formatted output

In [27]:

```
import numpy as np
x = np.linspace(0,2 * np.pi, 10)
# print(x)

y = x * np.sin(3 * x / np.pi)

z = [i > 1.5 for i in y]

for i,j,k in zip(x,y,z) :
    k = str(k)"Value of x : {0:.5f}".format(i) , "Value of y : {0:.5f}".format(j),
```

```
Value of x : 0.00000 Value of y : 0.00000 Value of Z False:
Value of x : 0.69813 Value of y : 0.43170 Value of Z False:
Value of x : 1.39626 Value of y : 1.35708 Value of Z False:
Value of x : 2.09440 Value of y : 1.90443 Value of Z True :
Value of x : 2.79253 Value of y : 1.27695 Value of Z False:
Value of x : 3.49066 Value of y : -0.66521 Value of Z False:
Value of x : 4.18879 Value of y : -3.17009 Value of Z False:
Value of x : 4.88692 Value of y : -4.88181 Value of Z False:
Value of x : 5.58505 Value of y : -4.54249 Value of Z False:
Value of x : 6.28319 Value of y : -1.75562 Value of Z False:
```

Now write the output from exercise 1 to a text file called “table.out”. Then read this file back in to three lists a, b, c, and verify that they are the same as x, y, z.

In [39]:

```
with open("table.out", "a") as f:

    for i,j,k in zip(x,y,z) :
        k = str(k)
        f.write(f"{i:10.5f} {j:10.5f} {k:5}\n")
    f.close
```

In [84]:

```
a ,b,c = [],[],[]
with open("table.out", "r") as f:
    # x = f.readline()
    for line in f:
        i, j, k = line.split()
        k = str(k)
        a.append("{0:.5f}".format(float(i)))
        b.append("{0:.5f}".format(float(j)))
        c.append("{0:5s}".format(k))
```

```
a
```

Out[84]:

```
['0.00000',
 '0.69813',
 '1.39626',
 '2.09440',
 '2.79253',
 '3.49066',
 '4.18879',
 '4.88692',
 '5.58505',
 '6.28319']
```

In [88]:

```
flag = True
for i,j,k,l,m,n in zip(x,a,y,b,z,c) :
    i = "{0:.5f}".format(float(i))
    k = "{0:.5f}".format(float(k))
    m = str(m)
    m = ("{0:5s}".format(m))
    if (i != j or k != l or m != n) :
        flag = False
        break
if (flag) :
    print("True")
```

```
True
```

In [103…

```
def process_string(input):
    vowels = ['a','e','i','o','u']
    vowel_count = {vowel: 0 for vowel in vowels}
    for i in input.lower():
        if i in vowels:
            vowel_count[i] += 1
    print("Vowel Counts:")
    for i in vowels:
        print(f"{i} : {vowel_count[i]}")
    number_word_to_num = {
        "zero": "0", "one": "1", "two": "2", "three": "3", "four": "4",
        "five": "5", "six": "6", "seven": "7", "eight": "8", "nine": "9",
        "ten": "10", "eleven": "11", "twelve": "12", "thirteen": "13",
        "fourteen": "14", "fifteen": "15", "sixteen": "16", "seventeen": "17",
        "eighteen": "18", "nineteen": "19"
    }
    words = input.split()

    replaced_string = " ".join([number_word_to_num[word] if word in number_word_to_

    print(replaced_string)




process_string("There are seventeen cats in twelve buildings with four bowls in eac
Vowel Counts:
a : 3
e : 10
i : 5
o : 2
u : 2
There are 17 cats in 12 buildings with 4 bowls in each.
```

In [133…

```python
string = "The quick brown fox jumped over the honest dog’s back"
def pig_latin(input):
    vowels = ['a','e','i','o','u', 'ho']
    words = input.split()
    latin_words = []
    for word in words:
        if (word[0] in vowels or word[:2] in vowels):
            new_word = word + "way"
        else:
            if (word.startswith("qu")):
                end = 2
            else :
                end = 0
                for i in word:
                    if i in vowels:
                        break
                    end += 1
            new_word = word[end:] + word[:end] + "ay"
        latin_words.append(new_word)
    # input[]
    latin_words[0] = latin_words[0].capitalize()
    for i in range(1, len(latin_words)):
        latin_words[i] = latin_words[i].lower()
    return ' '.join(latin_words)

print(pig_latin(string))
```

```
Ethay ickquay ownbray oxfay umpedjay overway ethay honestway og’sday ackbay
```

In [ ]: