Assignment 20 Solutions

Q1. Compare and contrast the float and Decimal classes' benefits and drawbacks ?

Ans: Both the `float` and `decimal` types store numerical values in Python.

We use `floats` when convenience and speed matter. A float gives us an approximation of the number we declare.

We use `decimals` when precision matters. `Decimals` can suffer from their own precision issues, but generally, `decimals` are more precise than `floats` . The performance difference between `float` and `decimal` , with Python 3, is not outlandish, and in my experience, the precision benefits of a `decimal` outweigh the performance benefits of a `float` .

Q2. Decimal('1.200') and Decimal('1.2') are two objects to consider. In what sense are these the same

object? Are these just two ways of representing the exact same value, or do they correspond to different internal states?

Ans: the output of both od them will be same this is just a way of representing precision upto 3 decimal points basic mathematics can be seen here in below example. the deciam class while outputting the result rund off the value to `1.2` from `1.200`

In [1]:

```
c  = 1.200
d = 1.2
print(c,d)
```

```
1.2 1.2
```

Q3. What happens if the equality of Decimal('1.200') and Decimal('1.2') is checked ?

Ans: Both values are checked to be equal since they only differ in precision.

Q4. Why is it preferable to start a Decimal object with a string rather than a floating-point value ?

Ans: Floating-point value is converted to Decimal format. Decimal can store float value with absolute precision. But when float value is given as Decimal object,it first has to be converted from floating point value which might already have rounding error.

Hence it is preferable to start a Decimal object with a string.

In [2]:

```
from decimal import *
getcontext().prec = 6
print(Decimal(1) / Decimal(7))
getcontext().prec = 28
print(Decimal(1) / Decimal(7))
```

```
0.142857
0.1428571428571428571428571429
```

Q5. In an arithmetic phrase, how simple is it to combine Decimal objects with integers ?

Ans: We can do it with use of `Decimal()` .

In [3]:

```
.6 + 1
```

Out[3]:

```
1.6
```

Q6. Can Decimal objects and floating-point values be combined easily ?

Ans: Arithmetic operfations like Adding,subtracting or multiplying a Decimal object by a floating-point value generates an error.

To do these operations, the floating point has to be converted to a Decimal.

Q7. Using the Fraction class but not the Decimal class, give an example of a quantity that can be expressed with absolute precision ?

Ans: Value of `0.5` will be represented as ½ .

In [4]:

```python
from fractions import Fraction

print (Fraction(11, 35))
# returns Fraction(11, 35)
print (Fraction(10, 18))
# returns Fraction(5, 9)
print (Fraction())
# returns Fraction(0, 1)

```

```
11/35
5/9
0
```

In [5]:

```python
from fractions import Fraction
print (Fraction('3.14159265358979323846'))
print (Fraction('3.14159265358979323846').limit_denominator(10000))
print (Fraction('3.14159265358979323846').limit_denominator(100))
print (Fraction('3.14159265358979323846').limit_denominator(10))
print (Fraction(125, 50).numerator)
print (Fraction(125, 50).denominator)
```

```
157079632679489661923/50000000000000000000
355/113
311/99
22/7
5
2
```

Q8. Describe a quantity that can be accurately expressed by the Decimal or Fraction classes but not by a floating-point value.

Answ: 0.00011 is a finite representation of an infinite number of digits. That doesn't help us with floating-point. Floating-point does not represent numbers using repeat bars; it represents them with a fixed number of bits it can be represented by decimal or fraction class

Q9.Consider the following two fraction objects: Fraction(1, 2) and Fraction(1, 2). (5, 10). Is the internal state of these two objects the same? Why do you think that is?

Ans: The internal state is same will give fraction of `1/2`

Q10. How do the Fraction class and the integer type (int) relate to each other? Containment or inheritance ?

Ans: Fraction class and integer type(int) are related in form of a container. It contains two ints, one in the numerator and the other in the denominator