Q1. The float class is a built-in data type in Python that represents floating-point numbers. It is fast and efficient, but can be imprecise when dealing with certain decimal values. On the other hand, the Decimal class provides higher precision for decimal arithmetic operations, but at the cost of performance. The Decimal class is recommended when working with financial applications, where accuracy is critical.

Q2. Decimal('1.200') and Decimal('1.2') are different objects that represent the same value. They correspond to different internal states, as the trailing zero in the first object is significant and is preserved in arithmetic operations.

Q3. The equality of Decimal('1.200') and Decimal('1.2') will return False, as they are different objects that represent the same value.

Q4. Starting a Decimal object with a string is preferred over a floating-point value because the latter can introduce inaccuracies due to the way floating-point numbers are represented internally.

Q5. Combining Decimal objects with integers is simple, as they can be used together in arithmetic expressions without any extra conversion steps.

Q6. Decimal objects and floating-point values can be combined, but it is important to be aware of potential precision issues and to use the appropriate conversion functions as needed.

Q7. An example of a quantity that can be expressed with absolute precision using the Fraction class is 1/3. In decimal notation, this value has an infinite repeating decimal representation, but it can be represented exactly as a fraction.

Q8. A quantity that can be accurately expressed by the Decimal or Fraction classes but not by a floating-point value is 1/10, which has an infinite repeating decimal representation in binary and therefore cannot be represented exactly by a float.

Q9. The internal state of Fraction(1, 2) and Fraction(1, 2, 5, 10) is not the same, as the latter represents a fraction in its reduced form (i.e., 1/2).

Q10. The Fraction class is not a subclass of the int type, but it can be used to represent integers with perfect precision by setting the denominator to 1.