Advanced Assignment

# Advanced Assignment 1

Q1. What is the purpose of Python's OOP?

Q2. Where does an inheritance search look for an attribute?

Q3. How do you distinguish between a class object and an instance object?

Q4. What makes the first argument in a class’s method function special?

Q5. What is the purpose of the \_\_init\_\_ method?

Q6. What is the process for creating a class instance?

Q7. What is the process for creating a class?

Q8. How would you define the superclasses of a class?

# Advanced Assignment 2

Q1. What is the relationship between classes and modules?

Q2. How do you make instances and classes?

Q3. Where and how should be class attributes created?

Q4. Where and how are instance attributes created?

Q5. What does the term "self" in a Python class mean?

Q6. How does a Python class handle operator overloading?

Q7. When do you consider allowing operator overloading of your classes?

Q8. What is the most popular form of operator overloading?

Q9. What are the two most important concepts to grasp in order to comprehend Python OOP code?

# Advanced Assignment 3

1. What is the concept of an abstract superclass?

2. What happens when a class statement's top level contains a basic assignment statement?

3. Why does a class need to manually call a superclass's \_\_init\_\_ method?

4. How can you augment, instead of completely replacing, an inherited method?

5. How is the local scope of a class different from that of a function?

# Advanced Assignment 4

Q1. Which two operator overloading methods can you use in your classes to support iteration?

Q2. In what contexts do the two operator overloading methods manage printing?

Q3. In a class, how do you intercept slice operations?

Q4. In a class, how do you capture in-place addition?

Q5. When is it appropriate to use operator overloading?

# Advanced Assignment 5

Q1. What is the meaning of multiple inheritance?

Q2. What is the concept of delegation?

Q3. What is the concept of composition?

Q4. What are bound methods and how do we use them?

Q5. What is the purpose of pseudoprivate attributes?

# Advanced Assignment 6

Q1. Describe three applications for exception processing.

Q2. What happens if you don't do something extra to treat an exception?

Q3. What are your options for recovering from an exception in your script?

Q4. Describe two methods for triggering exceptions in your script.

Q5. Identify two methods for specifying actions to be executed at termination time, regardless of whether or not an exception exists.

# Advanced Assignment 7

Q1. What is the purpose of the try statement?

Q2. What are the two most popular try statement variations?

Q3. What is the purpose of the raise statement?

Q4. What does the assert statement do, and what other statement is it like?

Q5. What is the purpose of the with/as argument, and what other statement is it like?

# Advanced Assignment 8

Q1. What are the two latest user-defined exception constraints in Python 3.X?

Q2. How are class-based exceptions that have been raised matched to handlers?

Q3. Describe two methods for attaching context information to exception artefacts.

Q4. Describe two methods for specifying the text of an exception object's error message.

Q5. Why do you no longer use string-based exceptions?

# Advanced Assignment 9

Q1. In Python 3.X, what are the names and functions of string object types?

Q2. How do the string forms in Python 3.X vary in terms of operations?

Q3. In 3.X, how do you put non-ASCII Unicode characters in a string?

Q4. In Python 3.X, what are the key differences between text-mode and binary-mode files?

Q5. How can you interpret a Unicode text file containing text encoded in a different encoding than your platform's default?

Q6. What is the best way to make a Unicode text file in a particular encoding format?

Q7. What qualifies ASCII text as a form of Unicode text?

Q8. How much of an effect does the change in string types in Python 3.X have on your code?

# Advanced Assignment 10

Q1. What is the difference between \_\_getattr\_\_ and \_\_getattribute\_\_?

Q2. What is the difference between properties and descriptors?

Q3. What are the key differences in functionality between \_\_getattr\_\_ and \_\_getattribute\_\_, as well as properties and descriptors?

# Advanced Assignment 11

Q1. What is the concept of a metaclass?

Q2. What is the best way to declare a class's metaclass?

Q3. How do class decorators overlap with metaclasses for handling classes?

Q4. How do class decorators overlap with metaclasses for handling instances?

# Advanced Assignment 12

Q1. Does assigning a value to a string's indexed character violate Python's string immutability?

Q2. Does using the += operator to concatenate strings violate Python's string immutability? Why or why not?

Q3. In Python, how many different ways are there to index a character?

Q4. What is the relationship between indexing and slicing?

Q5. What is an indexed character's exact data type? What is the data form of a slicing-generated substring?

Q6. What is the relationship between string and character "types" in Python?

Q7. Identify at least two operators and one method that allow you to combine one or more smaller strings to create a larger string.

Q8. What is the benefit of first checking the target string with in or not in before using the index method to find a substring?

Q9. Which operators and built-in string methods produce simple Boolean (true/false) results?

# Advanced Assignment 13

Q1. Can you create a programme or function that employs both positive and negative indexing? Is there any repercussion if you do so?

Q2. What is the most effective way of starting with 1,000 elements in a Python list? Assume that all elements should be set to the same value.

Q3. How do you slice a list to get any other part while missing the rest? (For example, suppose you want to make a new list with the elements first, third, fifth, seventh, and so on.)

Q4. Explain the distinctions between indexing and slicing.

Q5. What happens if one of the slicing expression's indexes is out of range?

Q6. If you pass a list to a function, and if you want the function to be able to change the values of the list—so that the list is different after the function returns—what action should you avoid?

Q7. What is the concept of an unbalanced matrix?

Q8. Why is it necessary to use either list comprehension or a loop to create arbitrarily large matrices?

# Advanced Assignment 14

Q1. Is an assignment operator like += only for show? Is it possible that it would lead to faster results at the runtime?

Q2. What is the smallest number of statements you'd have to write in most programming languages to replace the Python expression a, b = a + b, a?

Q3. In Python, what is the most effective way to set a list of 100 integers to 0?

Q4. What is the most effective way to initialise a list of 99 integers that repeats the sequence 1, 2, 3? S If necessary, show step-by-step instructions on how to accomplish this.

Q5. If you're using IDLE to run a Python application, explain how to print a multidimensional list as efficiently?

Q6. Is it possible to use list comprehension with a string? If so, how can you go about doing it?

Q7. From the command line, how do you get support with a user-written Python programme? Is this possible from inside IDLE?

Q8. Functions are said to be “first-class objects” in Python but not in most other languages, such as C++ or Java. What can you do in Python with a function (callable object) that you can't do in C or C++?

Q9. How do you distinguish between a wrapper, a wrapped feature, and a decorator?

Q10. If a function is a generator function, what does it return?

Q11. What is the one improvement that must be made to a function in order for it to become a generator function in the Python language?

Q12. Identify at least one benefit of generators.

# Advanced Assignment 15

What are the new features added in Python 3.8 version?

What is monkey patching in Python?

What is the difference between a shallow copy and deep copy?

What is the maximum possible length of an identifier?

What is generator comprehension?

# Advanced Assignment 16

Q1. What is the benefit of regular expressions?

Q2. Describe the difference between the effects of "(ab)c+" and "a(bc)+." Which of these, if any, is the unqualified pattern "abc+"?

Q3. How much do you need to use the following sentence while using regular expressions?

import re

Q4. Which characters have special significance in square brackets when expressing a range, and under what circumstances?

Q5. How does compiling a regular-expression object benefit you?

Q6. What are some examples of how to use the match object returned by re.match and re.search?

Q7. What is the difference between using a vertical bar (|) as an alteration and using square brackets as a character set?

Q8. In regular-expression search patterns, why is it necessary to use the raw-string indicator (r)? In   replacement strings?

# Advanced Assignment 17

Q1. Explain the difference between greedy and non-greedy syntax with visual terms in as few words as possible. What is the bare minimum effort required to transform a greedy pattern into a non-greedy one? What characters or characters can you introduce or change?

Q2. When exactly does greedy versus non-greedy make a difference?  What if you're looking for a non-greedy match but the only one available is greedy?

Q3. In a simple match of a string, which looks only for one match and does not do any replacement, is the use of a nontagged group likely to make any practical difference?

Q4. Describe a scenario in which using a nontagged category would have a significant impact on the program's outcomes.

Q5. Unlike a normal regex pattern, a look-ahead condition does not consume the characters it examines. Describe a situation in which this could make a difference in the results of your programme.

Q6. In standard expressions, what is the difference between positive look-ahead and negative look-ahead?

Q7. What is the benefit of referring to groups by name rather than by number in a standard expression?

Q8. Can you identify repeated items within a target string using named groups, as in "The cow jumped over the moon"?

Q9. When parsing a string, what is at least one thing that the Scanner interface does for you that the re.findall feature does not?

Q10. Does a scanner object have to be named scanner?

# Advanced Assignment 18

Q1. Describe the differences between text and binary files in a single paragraph.

Q2. What are some scenarios where using text files will be the better option? When would you like to use binary files instead of text files?

Q3. What are some of the issues with using binary operations to read and write a Python integer directly to disc?

Q4. Describe a benefit of using the with keyword instead of explicitly opening a file.

Q5. Does Python have the trailing newline while reading a line of text? Does Python append a newline when you write a line of text?

Q6. What file operations enable for random-access operation?

Q7. When do you think you'll use the struct package the most?

Q8. When is pickling the best option?

Q9. When will it be best to use the shelve package?

Q10. What is a special restriction when using the shelve package, as opposed to using other data dictionaries?

# Advanced Assignment 19

Q1. Define the relationship between a class and its instances. Is it a one-to-one or a one-to-many partnership, for example?

Q2. What kind of data is held only in an instance?

Q3. What kind of knowledge is stored in a class?

Q4. What exactly is a method, and how is it different from a regular function?

Q5. Is inheritance supported in Python, and if so, what is the syntax?

Q6. How much encapsulation (making instance or class variables private) does Python support?

Q7. How do you distinguish between a class variable and an instance variable?

Q8. When, if ever, can self be included in a class's method definitions?

Q9. What is the difference between the \_ \_add\_ \_ and the \_ \_radd\_ \_ methods?

Q10. When is it necessary to use a reflection method? When do you not need it, even though you support the operation in question?

Q11. What is the \_ \_iadd\_ \_ method called?

Q12. Is the \_ \_init\_ \_ method inherited by subclasses? What do you do if you need to customize its behavior within a subclass?

# Advanced Assignment 20

1. Compare and contrast the float and Decimal classes' benefits and drawbacks.

2. 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?

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

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

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

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

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

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

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?

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

# Advanced Assignment 21

Q1. What is a probability distribution, exactly? If the values are meant to be random, how can you predict them at all?

Q2. Is there a distinction between true random numbers and pseudo-random numbers, if there is one? Why are the latter considered “good enough”?

Q3. What are the two main factors that influence the behaviour of a "normal" probability distribution?

Q4. Provide a real-life example of a normal distribution.

Q5. In the short term, how can you expect a probability distribution to behave? What do you think will happen as the number of trials grows?

Q6. What kind of object can be shuffled by using random.shuffle?

Q7. Describe the math package's general categories of functions.

Q8. What is the relationship between exponentiation and logarithms?

Q9. What are the three logarithmic functions that Python supports?

# Advanced Assignment 22

Q1. What are the benefits of the built-in array package, if any?

Q2. What are some of the array package's limitations?

Q3. Describe the main differences between the array and numpy packages.

Q4. Explain the distinctions between the empty, ones, and zeros functions.

Q5. In the fromfunction function, which is used to construct new arrays, what is the role of the callable argument?

Q6. What happens when a numpy array is combined with a single-value operand (a scalar, such as an int or a floating-point value) through addition, as in the expression A + n?

Q7. Can array-to-scalar operations use combined operation-assign operators (such as += or \*=)? What is the outcome?

Q8. Does a numpy array contain fixed-length strings? What happens if you allocate a longer string to one of these arrays?

Q9. What happens when you combine two numpy arrays using an operation like addition (+) or multiplication (\*)? What are the conditions for combining two numpy arrays?

Q10. What is the best way to use a Boolean array to mask another array?

Q11. What are three different ways to get the standard deviation of a wide collection of data using both standard Python and its packages? Sort the three of them by how quickly they execute.

12. What is the dimensionality of a Boolean mask-generated array?

# Advanced Assignment 23

Q1. If you have any, what are your choices for increasing the comparison between different figures on the same graph?

Q2. Can you explain the benefit of compound interest over a higher rate of interest that does not compound after reading this chapter?

Q3. What is a histogram, exactly? Name a numpy method for creating such a graph.

Q4. If necessary, how do you change the aspect ratios between the X and Y axes?

Q5. Compare and contrast the three types of array multiplication between two numpy arrays: dot product, outer product, and regular multiplication of two numpy arrays.

Q6. Before you buy a home, which numpy function will you use to measure your monthly mortgage payment?

Q7. Can string data be stored in numpy arrays? If so, list at least one restriction that applies to this data.

# Advanced Assignment 24

Q1. Is it permissible to use several import statements to import the same module? What would the goal be? Can you think of a situation where it would be beneficial?

Q2. What are some of a module's characteristics? (Name at least one.)

Q3. Circular importing, such as when two modules import each other, can lead to dependencies and bugs that aren't visible. How can you go about creating a program that avoids mutual importing?

Q4. Why is \_ \_all\_ \_ in Python?

Q5. In what situation is it useful to refer to the \_ \_name\_ \_ attribute or the string '\_ \_main\_ \_'?

Q6. What are some of the benefits of attaching a program counter to the RPN interpreter application, which interprets an RPN script line by line?

Q7. What are the minimum expressions or statements (or both) that you'd need to render a basic programming language like RPN primitive but complete— that is, capable of carrying out any computerised task theoretically possible?

# Advanced Assignment 25

Q1. What is the distinction between a numpy array and a pandas data frame? Is there a way to convert between the two if there is?

Q2. What can go wrong when an user enters in a stock-ticker symbol, and how do you handle it?

Q3. Identify some of the plotting techniques that are used to produce a stock-market chart.

Q4. Why is it essential to print a legend on a stock market chart?

Q5. What is the best way to limit the length of a pandas data frame to less than a year?

Q6. What is the definition of a 180-day moving average?

Q7. Did the chapter's final example use "indirect" importing? If so, how exactly do you do it?