MCQ Answers

1. Option (c) 15
2. Option (c) List

3. a) When passed into a Tuple: The *args arguments will be stored as a tuple.

   b) When passed into a List: The *args arguments will still be stored as a tuple, but if you explicitly convert it to a list using list(args), it will become a list.

   c) When passed into a Dictionary: The *args arguments will still be stored as a tuple. However, if you pass it into a dictionary directly, it will be used to unpack the key-value pairs. In this case, the *args must contain an even number of elements, where every pair of elements represents a key-value pair. The resulting dictionary will be constructed using the elements of *args as keys and values.

   d) If *args is not passed into any data structure explicitly, it will remain as a tuple.

4. Option (d) Error.

The given code will raise a TypeError because the + operator is not supported for set concatenation in Python. The + operator is used for concatenating sequences like lists and strings, but it does not work directly on sets. To combine multiple sets into a single set, you can use the union operation (|) or the union() method. Here's the corrected code:

```
set1 = {14, 3, 55}

set2 = {82, 49, 62}

set3 = {99, 22, 17}

combined_set = set1 | set2 | set3

print(len(combined_set))
```

5. Option (a) raise

   The raise keyword is used to raise exceptions explicitly. It allows you to generate an exception manually at any point in your code. When an exception is raised, it interrupts the normal flow of the program and jumps to the nearest exception handler that can handle that particular exception.

6. Option (c) datetime.
7. Option (c) 208
8. Option (d) None

9. Option (b) Immutable
10. Option (A) range()

Question 11

Amongst which of the following is a function which does not have any name?

Answer: Option (c) Lambda Function

A lambda function, also known as an anonymous function, is a function that does not have a name. It is defined using the lambda keyword, followed by the function's parameters and a colon, and then the function's body. Lambda functions are typically used for small, one-time functions that don't require a formal definition.

Question 12

The module Pickle is used to ___.

Answer: Option (c) Both A and B

The pickle module in Python is used for both serializing and de-serializing Python object structures.

Question 13

Amongst which of the following is / are the method of convert Python objects for writing data in a binary file?

Answer: Option (b) dump() method

The dump() method is used to convert Python objects for writing data in a binary file. It is a method provided by the pickle module in Python. The dump() method takes two arguments: the object to be serialized and the file object where the serialized data will be written.

Question 14

Amongst which of the following is / are the method used to unpickling data from a binary file?

Answer: Option (A) load()
The load() method is used to unpickle data from a binary file. The load() method takes a file object as an argument and returns the deserialized Python object.

Question 15

A text file contains only textual information consisting of ___.

Answer: Option (d) All of the mentioned above.

A text file can contain textual information consisting of alphabets, numbers, and special symbols. Text files are commonly used to store and represent human-readable data, which can include a combination of letters, digits, and various special characters.

Question 16

Which Python code could replace the ellipsis (...) below to get the following output? (Select all that apply.)

```
 captains = {
    "Enterprise": "Picard",
    "Voyager": "Janeway",
    "Defiant":
"Sisko",  }
```

```
    Enterprise Picard,
      Voyager Janeway
      Defiant Sisko
```

Answer: Option (d) both a and b

Explanation: Option (a):

```
for ship, captain in captains.items():
    print(ship, captain)
```

In this option, we are using the items() method of the captains dictionary to iterate over its key-value pairs. This allows us to directly access both the ship and the captain in each iteration and print them.

Option(b):

```
for ship in captains:
    print(ship, captains[ship])
```

In this option, we are iterating over the keys of the captains dictionary using a simple for loop. For each ship, we access the corresponding captain using captains[ship] and print both ship and captain. Both options will produce the desired output

Question 17

Which of the following lines of code will create an empty dictionary

named captains?

Answer: Option (d) captains = {}

Question 18

Now you have your empty dictionary named `captains`. It's time to add some data! Specifically, you want to add the key-value pairs "Enterprise": "Picard", "Voyager": "Janeway", and `"Defiant": "Sisko"`. Which of the following code snippets will successfully add these key-value pairs to the existing `captains` dictionary?

Answer: Option (b)

```
captains["Enterprise"] = "Picard"

captains["Voyager"] = "Janeway"

captains["Defiant"] = "Sisko"
```

In this option, we use square brackets [] to access the dictionary captains and assign values to specific keys. Each line adds a key-value pair to the dictionary.

Question 19

You're really building out the Federation Starfleet now! Here's what you have:

```
captains = {
  "Enterprise": "Picard",
  "Voyager": "Janeway",
  "Defiant": "Sisko",
```

```
    "Discovery": "unknown",
```

}Now, say you want to display the ship and captain names contained in the dictionary, but you also want to provide some additional context. How could you do it?

Answer: Option (b)

```
for ship, captain in captains.items():
    print(f"The {ship} is captained by {captain}.")
```

Option (b) correctly utilizes the items() method of the captains dictionary to iterate over its key-value pairs. By unpacking each pair into the variables ship and captain, we can access and print the ship and captain names using string interpolation ({}). This option provides the desired output by displaying the ship and captain names with additional context.

Question 20

You've created a dictionary, added data, checked for the existence of keys, and iterated over it with a for loop. Now you're ready to delete a key from this dictionary:

```
captains = {
    "Enterprise": "Picard",
    "Voyager": "Janeway",
    "Defiant": "Sisko",
    "Discovery": "unknown",
}
```

What statement will remove the entry for `"Discovery"` the key ?

Answer: Option (c) del captains["Discovery"]

This statement uses the del keyword followed by the dictionary name and the key in square brackets to delete the specified key-value pair from the dictionary. In this case, it will remove the entry for the key "Discovery" from the captains dictionary.