Assignment

Q 1 What will be the output of the following code snippet?

def func(a, b):

return b if a == 0 else func(b % a, a) print(func(30, 75)) a) 10 b) 20 c) 15 d) 0

Ans. The output of the code snippet will be 15.

The function **`func`** takes two arguments **a** and **b**. It uses recursion to find the greatest common divisor (GCD) of **a** and **b**.

In each recursive call, the function checks if **a** is equal to 0. If it is, the function returns **b**, which is the GCD. If **a** is not equal to 0, the function calls itself with arguments **b % a** and **a**.

In this example, the initial arguments are 30 and 75. The function makes the following recursive calls:

func(30, 75) = func(15, 30) # b % a = 75 % 30 = 15 = func(0, 15) # b % a = 30 % 15 = 0

Since **a** is now 0, the function returns **b**, which is 15. Therefore, the output of the code snippet is 15.

Q 2 numbers = (4, 7, 19, 2, 89, 45, 72, 22) sorted_numbers = sorted(numbers) even = lambda a: a % 2 == 0 even_numbers = filter(even, sorted_numbers) print(type(even_numbers)) a) Int b) Filter c) List d) Tuple

Ans. The output of the code snippet is of type **`filter`**.

In the code snippet, a tuple of integers **`numbers`** is defined. This tuple is then sorted using the **`sorted()`** function and stored in a new variable called **`sorted_numbers`**.

The **`filter()`** function is used to apply a lambda function called **`even`** to each element of **`sorted_numbers`**. The **`even`** function returns **`True`** if the input number is even and **`False`** otherwise. The **`filter()`** function returns an iterator that contains only the elements of **`sorted_numbers`** that satisfy the **`even`** function.

The output of the **`filter()`** function is an iterator, not a list or tuple. The **`type()`** function is then used to determine the data type of this iterator. Therefore, the output of the code snippet is of type **`filter`**.

Q3 As what datatype are the *args stored, when passed into a) Tuple b) List c) Dictionary d) none

Ans. This code will raise a TypeError because you cannot use the **+** operator to concatenate sets. Instead, you need to use the **`union()`** method to combine the sets into a new set.

Q 4. set1 = {14, 3, 55}

set2 = {82, 49, 62}

set3={99,22,17}

print(len(set1 + set2 + set3))

Ans. This code will raise a TypeError because you cannot use the **+** operator to concatenate sets. Instead, you need to use the **`union()`** method to combine the sets into a new set.

Q 5. What keyword is used in Python to raise exceptions? a) raise b) try c) goto d) except

Ans. The keyword used in Python to raise exceptions is "raise". Therefore, the correct option is a) raise.

The "raise" keyword is used to raise an exception manually in a Python program. When a program encounters an exceptional situation that it cannot handle, it can raise an exception using the "raise" keyword. This allows the program to transfer control to an exception handler, which can then take appropriate actions to handle the exception.

Q 6 Which of the following modules need to be imported to handle date time computations in Python? a) timedate b) date c) datetime d) time

Ans. The module that needs to be imported to handle date time computations in Python is "datetime". Therefore, option c) is the correct answer.

The other modules listed, "timedate", "date", and "time" are not valid Python standard library modules for handling datetime computations.

7) What will be the output of the following code snippet? print(4**3 + (7 + 5)**(1 + 1)) a) 248 b) 169 c) 208 d) 233

Ans. The order of operations in Python follows the acronym PEMDAS, which stands for Parentheses, Exponents, Multiplication and Division (performed from left to right), and Addition and Subtraction (performed from left to right).

Using this order, the expression **`4**3 + (7 + 5)**(1 + 1)`** would be evaluated as follows:

1. Evaluate the expression inside the parentheses: **`(7 + 5) = 12`** and **`(1 + 1) = 2`**.
2. Evaluate the exponentiation inside the parentheses: **`(12 ** 2) = 144`**.
3. Evaluate the exponentiation outside the parentheses: **`(4 ** 3) = 64`**.
4. Add the results of step 2 and step 3: **`144 + 64 = 208`**.

Therefore, the output of the code snippet **`print(4**3 + (7 + 5)**(1 + 1))`** would be **`208`**.

Therefore, the correct answer is option c) 208.

8) Which of the following functions converts date to corresponding time in Python? a) strptime b) strftime c) both a) and b) d) None

Ans. The module that needs to be imported to handle date time computations in Python is "datetime". Therefore, option c) is the correct answer.

The other modules listed, "timedate", "date", and "time" are not valid Python standard library modules for handling datetime computations.

9) The python tuple is _____ in nature. a) mutable b)immutable c)unchangeable d) none

Ans. The correct answer is b) immutable.

In Python, a tuple is an ordered collection of elements, enclosed in parentheses (). Tuples are immutable, which means that once a tuple is created, you cannot add, remove or modify any of its elements. You can only access or iterate over the elements of a tuple.

10) The ___ is a built-in function that returns a range object that consists series of integer numbers, which we can iterate using a for loop. A. range() B. set() C. dictionary{} D. None of the mentioned above

Ans. A. range()

The range() function is a built-in function in Python that returns a range object, which is an ordered sequence of integers. The range() function takes three arguments: start, stop, and step, and generates a sequence of numbers from start to stop (exclusive) with the given step size.

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

A. Del function B. Show function C. Lambda function D. None of the mentioned above

Ans. C. Lambda function

A lambda function, also known as an anonymous function, is a function without a name. It is a small, single-expression function that can take any number of arguments, but can only have one expression. Lambda functions are defined using the lambda keyword in Python. They are often used as a shorthand for simple operations that would otherwise require a full function definition.

12) The module Pickle is used to ___.

Ans. C. Both A and B

The module Pickle in Python is used for serializing and de-serializing a Python object structure. In other words, it converts a Python object into a byte stream for storage or transmission, and converts the byte stream back into a Python object when needed. The Pickle module can be used to save and restore complex Python objects such as lists, dictionaries, functions, and class instances.

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

Ans. B. dump() method

In Python, the module used for writing and reading data in a binary file is called 'pickle'. To convert Python objects for writing data in a binary file, we use the 'dump()' method. The 'dump()' method serializes the Python object and writes it to a binary file. To read the data from a binary file, we use the 'load()' method, which de-serializes the data and returns the original Python object. The 'set()' method is used to create a set object in Python and is not related to binary file operations.

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

A. load() B. set() method C. dump() method D. None of the mentioned above

Ans. A. load() method is used to unpickle data from a binary file in Python.

15.) A text file contains only textual information consisting of ___.

A. Alphabets B. Numbers C. Special symbols D. All of the mentioned above

Ans. A text file can contain alphabets, numbers, and special symbols, so the answer is D. All of the mentioned above.

Ans. The Python code that could replace the ellipsis (...) to get the mentioned output is:

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

Therefore, the correct answer is (a) for ship, captain in captains.items(): print(ship, captain) and (d) both a and b.

16) Which of the following lines of code will create an empty dictionary named captains?

Ans. The correct answer is d) captains = {}.

This line of code creates an empty dictionary named **`captains`**. The curly braces {} are used to denote an empty dictionary in Python. Option a) **`captains = {dict}`** assigns the **`dict`** class to the **`captains`** variable, not an empty dictionary. Option b) **`type(captains)`** returns the type of the **`captains`** variable, but does not create a dictionary. Option c) **`captains.dict()`** tries to call a **`dict()`** method on the **`captains`** variable, but dictionaries do not have such a method, so it would result in an AttributeError.

17) 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? a) captains{"Enterprise" = "Picard"} captains{"Voyager" = "Janeway"} captains{"Defiant" = "Sisko"} b) captains["Enterprise"] = "Picard" captains["Voyager"] = "Janeway" captains["Defiant"] = "Sisko" c) captains = { "Enterprise": "Picard", "Voyager": "Janeway", "Defiant": "Sisko", } d) None of the above

Ans. The correct code snippet to add the key-value pairs to the existing captains dictionary is (b) captains["Enterprise"] = "Picard"; captains["Voyager"] = "Janeway"; captains["Defiant"] = "Sisko".

Explanation:

- Option (a) is incorrect syntax for adding key-value pairs to a dictionary. It should use colons instead of equals signs: captains = {"Enterprise": "Picard"}.
- Option (c) is also correct, but it reassigns the entire dictionary instead of adding to it.
- Option (d) is not correct because (b) is a valid option.

18) 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 a) for ship, captain in captains.items(): print(ship, captain) b) for ship in captains: print(ship, captains[ship]) c) for ship in captains: print(ship, captains) d) both a and b

Ans. d) both a and b

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? a) for item in captains.items(): print(f"The [ship] is captained by [captain].") b) for ship, captain in captains.items(): print(f"The {ship} is captained by {captain}.") c) for captain, ship in captains.items(): print(f"The {ship} is captained by {captain}.") d) All are correct

Ans. The correct way to display the ship and captain names contained in the dictionary with additional context is (b) for ship, captain in captains.items(): print(f"The {ship} is captained by {captain}.").

Explanation:

- Option (a) is incorrect because it uses square brackets instead of curly braces to reference the ship and captain variables, and these variables are not defined in the loop.
- Option (c) is incorrect because it swaps the order of the ship and captain variables in the loop, which will result in incorrect output.
- Option (d) is incorrect because only option (b) is correct.

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 the key "Discovery"? a) del captains b) captains.remove() c) del captains["Discovery"] d) captains["Discovery"].pop()

Ans. c) del captains["Discovery"]