FLIP ROBO TECHNOLOGIES

INTERNSHIP – DS0523

PROJECT - BATCH DS2311

MOHAN G

MCQ

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

Solution: **Option C – 15**

The given code is similar to a recursive function that takes 2 variables a and b.
The initial call is func(30, 75). Since a is not equal to 0, it goes to the else part and the function is called as (75 % 30, 30) i.e., 75 modulus of 30 which means the remainder value when the 75 is divided by 30 so it will be 15, which is equivalent to func(15, 30).
Again, a is not equal to 0, so it goes to else part (30 % 15, 15), i.e., 30 modulus of 15 which means the remainder value when the 30 is divided by 15 so it will be 0 which is equivalent to func(0, 15).
Now, a is equal to 0, so the function returns b, which is 15.

Hence, the output of the code snippet will be 15.

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

Solution: **Option B – Filter**

The numbers function is defined in tuples data structures and sort function is used to sort the list of number given in the tuple.
Lambda function even is used to check whether a given number is even or not. (All the even numbers will be divisible by 2 – Criteria for even numbers).
The filter function is used to create a filter object called even numbers, which contains only the even number in the sorted numbers list.
Lastly, type function is used to print the type of the even numbers.

Hence, the output of the is <class 'filter'> (built in python type for filtered objects).

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

Solution: **Option A – Tuple**

By default, *args will be stored as a tuple.

4. **set1 = {14, 3, 55}**
   set2 = {82, 49, 62}
   set3={99,22,17}
   print(len(set1 + set2 + set3))
   a) 105
   b) 270
   c) 0
   d) Error

Solution: **Option D – Error**

The + operator is not defined for sets in Python, so it will result in a Type Error.

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

Solution: **Option A – Raise**

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

Solution: **Option C – datetime**

7. **What will be the output of the following code snippet?**
   print(43 + (7 + 5)**(1 + 1))**
   a) 248
   b) 169
   c) 208
   d) 233

Solution: **Option C – 208**

Detailed expression breakup is as follows:
4**3 indicates 4 power of 3, which will be 4x4x4 = 64

(7+5)**(1+1) = 12**2 = 12 power of 2 = 12x12 = 144

So, 64+144 = 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

Solution: **Option A – strptime**

strptime function is used to parse a string representing a date and time and convert it into a corresponding datetime object.
And strftime is vice versa.

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

Solution: **Option B – Immutable**

The key characteristic of a tuple. Once a tuple is created, its elements cannot be changed or modified.

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

Solution: **Option A – range()**

The range() function in Python is a built-in function that returns a sequence of numbers, often used for iterating over a sequence of numbers in a loop. The range() function can take one, two, or three arguments, and it generates a range object representing a sequence of numbers.

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

Solution: **Option C – Lambda function**

Lambda functions in Python are also known as anonymous functions because they don't have a name like regular functions defined using the def keyword. Lambda functions are defined using the lambda keyword, followed by a list of parameters, a colon, and an expression.

12. **The module Pickle is used to ___.**
    A. Serializing Python object structure
    B. De-serializing Python object structure
    C. Both A and B
    D. None of the mentioned above

Solution: **Option C – Both A and B**

The pickle module in Python is used for serializing and deserializing Python object structures. Serialization is the process of converting a Python object into a byte stream, and deserialization is the process of reconstructing the original Python object from a byte stream. The pickle module provides functions like pickle.dump() for serialization and pickle.load() for deserialization.

13. **Amongst which of the following is / are the method of convert Python objects for writing data in a binary file?**
    A. set() method
    B. dump() method
    C. load() method
    D. None of the mentioned above

Solution: **Option B – dump() method**

Converting Python objects for writing data to a binary file, the dump() method is associated with the pickle module. The pickle.dump() method is used to serialize a Python object and write it to a binary file. It takes an object and a file-like object as arguments and writes the serialized data to the file.

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

Solution: **Option A – load() method**

Unpickling data from a binary file, the load() method is associated with the pickle module. The pickle.load() method is used to deserialize data from a binary file. It reads the serialized data from the file and reconstructs the original Python object.

15. **A text file contains only textual information consisting of ___.**
    A. Alphabets
    B. Numbers
    C. Special symbols
    D. All of the mentioned above

Solution: **Option D – All of the mentioned above**

A text file can contain a combination of alphabets, numbers, and special symbols. Therefore, it includes all the mentioned types of characters.

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

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

Solution: **Option D – Both a and b**

a. for ship, captain in captains.items():
```
        print(ship, captain)
```
This code iterates over the key-value pairs in the captains dictionary using the items() method. In each iteration, it unpacks the key-value pair into the variables ship and captain, and then prints them.

b. for ship in captains:
```
        print(ship, captains[ship])
```
This code iterates over the keys in the captains dictionary. In each iteration, it prints the key (ship) and retrieves the corresponding value (captains[ship]) from the dictionary.

17. Which of the following lines of code will create an empty dictionary named captains?
a) captains = {dict}
b) type(captains)
c) captains.dict()
d) captains = {}

Solution: **Option D – captain {}**

The {} syntax is used to represent an empty dictionary literal in Python.

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?

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

Solution: Option B –

```
captains["Enterprise"] = "Picard"
captains["Voyager"] = "Janeway"
captains["Defiant"] = "Sisko"
```

The syntax used in option (a) is not for adding key-value pairs to a dictionary in Python. The correct syntax uses square brackets, not curly braces.
Option (c) is a way to create a new dictionary with the specified key-value pairs, but it doesn't add these pairs to an existing dictionary. It creates a new dictionary and assigns it to the variable captains.

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

Solution: **Option B –**

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

Option (a) uses the placeholder [ship] and [captain] inside the f-string, but this syntax is not valid in Python. The correct syntax uses curly braces {}.
Option (b) correctly uses the placeholders {ship} and {captain} inside the f-string, providing the desired context.
Option (c) swaps the order of captain and ship in the f-string, which may lead to confusion and incorrect output.

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()

Solution: **Option C – del captains[“Discovery”]**

(a) del captains: This statement would delete the entire captains dictionary, not just the entry for the key "Discovery."
b) captains.remove(): The remove() method is not applicable to dictionaries. It is used for lists to remove a specified element.
d) captains["Discovery"].pop(): This statement would raise an AttributeError because dictionaries do not have a pop() method. Additionally, if "Discovery" is not present in the dictionary, it would raise a KeyError