Find and read 20 interview questions for Data Types, Operators, Conditional Statements, Looping Statements, Functions.

Data types Questions:

#1. What are the built-in data types in Python?
Ans: The built-in data types in python are ***int, float, complex, bool, list, tuple, set, dict, str.***

#2. What's the difference between list and tuple?
Ans: The main difference between list and tuple is ***"the list is mutable objects in case of the tuple is immutable objects".***

#3. What are mutable and immutable data types?
Ans: Mutable data types are ***list, set, dict*** in python which can modified after creating them. In case of Immutable data types are ***int, float, str and tuple*** which cannot be modified after creating them.

#4. What are some methods of the list used in python?
Ans: Some of the methods used in list python are:

- Python List **append()** Add a single element to the end of the list
- Python List **sort()** Sort elements of a List
- Python List **remove()** Removes items from the list
- Python List **pop()** removes element at the given index
- Python List **reverse()** Reverse the list
- Python List **count()** Returns count of the element in the list

#5. Which data types allow slicing?
Ans: We can use slicing on ***list, tuple, and str*** data types in python.

#6. What are the unpacking operators in python?
Ans: You can say ***and** operators are unpacking operators in python. The * unpacking operators is used to assign multiple values to different values at a time from sequence data types.

#7. What are some methods in string?
Ans: 1. Split: the methods used to split the at desired points. It returns the list as a result. By default it splits the string at spaces

2. **Join:** the method is used to combine the list of string objects. It combines the string objects with the delimiter we provide.

Note: Some other methods of strings are: ***capitalize, isalnum, isalpha, isdigit, lower, upper, center,*** **etc..,**

#8. What is the negative indexing in lists?
Ans: The index is used to access the element from the lists. Normal indexing of the list starts from *0*. The start of the negative indexing is *-1*. And it keeps on increasing like *-2*, *-3*, *-4*, etc.., till the length of the list. It allows us to access the index from the end of the list. Example:

```
neg_index = [1, 2, 3, 4, 5]
print(neg_index[-1])
Output is 5
```

#9. What are some methods in dict?

Ans: 1. Items: the method returns *key: value* pairs of dictionaries as a list of tuples.

```
dict1 = {1: 'Rajkumar', 2: 'Kavirajan', 3: 'Sachin'}
print(dict1.items())

Output is
dict_items([(1, 'Rajkumar'), (2, 'Kavirajan'), (3, 'Sachin')])
```

2. Pop: the method is used to remove the *key: value* pair from the dictionary. It accepts the key as an argument and removes it from the dictionary.

```
dict1 = {1: 2, 2: 3}
print(dict1.pop(2))

output is 3
```

Note: Some other methods of *dict* are: ***get, keys, values, clear,*** **etc.**

Condition and loops Questions:

#10. Does Python have switch statement?
Ans: No Python don't have switch statement.

#11. What's the difference between for loop and while loop in Python?
Ans: The main difference between for loop and while loop is ***"the for loop is used when the range can be known and in case of the while loop user don't know the range to iterate".***

#12. What are break, pass and continue statements?
Ans: Break: It is used to terminate the running loop and the execution will jump out of the break loop.

Pass: It is used as place holder for future order and when executed, nothing happens, but you avoid getting an error when empty code is not allowed.

Continue: It's used to skip the execution of the remaining code and doesn't execute in the current iteration and goes to next iteration.

#13. When is the code in else executed with while and for loops?
Ans: The code inside the ***else*** block with ***while*** and ***for*** loops is executed after executing all iterations. And the code inside the ***else*** block doesn't execute when we break the loops.

#14. What are list and dictionary comprehension?
Ans: List comprehensions and dictionary comprehensions are **a powerful substitute to for-loops and also lambda functions** and they are also faster than for loops.

Example for list and dictionary comprehension:

```
lst = [i for i in range(10)]

lst = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

print(lst)
```

```
dict1 = {i: i + 1 for i in range (10)}
dict1 = {0: 1, 1: 2, 2: 3, 3: 4, 4: 5, 5: 6, 6: 7, 7: 8, 8: 9, 9: 10}
print(dict1)
```

#15. How do you implement the functionality of switch statements in Python?
Ans: We can implement the functionality of switch statements using *if* and ***elif*** statements. We **start with an 'if' statement followed by 'if-elif' statements and in the end, we add the 'else' statement**.

Example:

```
if switch == 1:
    print(...)
elif switch == 2:
    print(....)
```

#16. How does the range function work?
Ans: The range function returns the sequence of numbers between the start to stop with a step increment. The syntax of the range function is ***range(start, stop[, step])***.

The ***stop*** argument is mandatory. The arguments ***start*** **and** ***step*** are optional. The default value of *start* and *step* are *0* and *1,* respectively. Example:

```
list(range(10))
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
list(range(1, 10))
[1, 2, 3, 4, 5, 6, 7, 8, 9]
list(range(1, 10, 2))
[1, 3, 5, 7, 9]
```

Functions Questions:

#17. What are the parameters and argument in python?
Ans: The parameters are the names listed in the function definition and argument are the values passed to the function while invoking.

#18. What are the different types of arguments in python?
Ans: There are mainly four types of arguments. They are ***positional arguments, default arguments, keyword arguments, and arbitrary arguments.***

- **positional arguments** are used when normal arguments are defined in user defined functions and it's required while invoking the function.
- **default arguments** are used when the user didn't pass the value the function will consider the default value.
- **Keyword arguments** are used when we can specify the name of the arguments while invoking the function and assign values to them and its helps us to avoid ordering whereas positional arguments is mandatory.
- **Arbitrary arguments** are used to collect a bunch of values at a time when we don't know the number of arguments the function will get, *and** operator is use to collect

#19. What is the lambda function in Python?

Ans: Lambda functions are small anonymous functions in Python. It has single expressions and accepts multiples arguments. Example:

```
add = lambda a, b: a + b
print(add(1, 3))
output is 4
```

#20. What is recursive function?

Ans: The function calling itself is called a recursive function. So, you can loop through data to reach a result.

Example of a recursive function:

Following is an example of a recursive function to find the factorial of an integer,

```
def factorial(recur):
    if recur == 1:
        return 1
    else:
        return (recur * factorial(recur-1))


num = 3
print("The factorial of", num, "is", factorial(num))

Output is
The factorial of 3 is 6
```