Python

1.How do you get a list of all the keys in a dictionary

```
dictionary = {"a": 1, "b": 2, "c": 3}
keys = list(dictionary.keys()) print(keys)
 # Output: ['a', 'b', 'c']
```

2. Write a program to filter out the number divisible by 3 from a tuple using lambda function

```
numbers = (1, 2, 3, 4, 5, 6, 7, 8, 9)
divisible_by_3 = tuple(filter(lambda x: x % 3 == 0, numbers))
print(divisible_by_3) # Output: (3, 6, 9)
```

3. Why do we need different data types in python

Different data types in Python are necessary to handle various kinds of data appropriately, ensuring efficient storage and operations. For example, integers for counting, floats for decimal numbers, strings for text, lists for ordered collections, and dictionaries for key-value pairs

4.How to check the data type of variable in python

```
variable = 42
print(type(variable)) # Output: <class 'int'>
```

5.Convert a given string to int using a single line of code

```
string_number = "123"
integer_number = int(string_number)
print(integer_number) # Output: 123
```

Python

6.What is the difference between mutable and immutable?
Mutable: Objects that can be changed after creation (e.g., lists, dictionaries).
Immutable: Objects that cannot be changed after creation (e.g., tuples, strings)

7.How can you delete a file in Python?
```
import os
os.remove("filename.txt")
```

8**. How to access an element of a list?**
```
my_list = [1, 2, 3, 4]
print(my_list[2]) # Output: 3
```

9**.Discuss different ways of deleting an element from a list.**
```
del my_list[2]
my_list.remove(3)
my_list.pop(2)
```

10.Write a code snippet to delete an entire list.
```
my_list = [1, 2, 3, 4]
del my_list
```

Python

11.Write a code snippet to reverse an array

```
my_list = [1, 2, 3, 4]
my_list.reverse()
print(my_list) # Output: [4, 3, 2, 1]
```

12.Write a code snippet to get an element, delete an element, and update an element in an array.

```
import array as arr

my_array = arr.array('i', [1, 2, 3, 4])
# Get
print(my_array[2]) # Output: 3
# Update
my_array[2] = 5
# Delete
del my_array[2]
```

13. Write a code snippet to generate the square of every element of a list.

```
numbers = [1, 2, 3, 4]
squares = [x**2 for x in numbers]
print(squares) # Output: [1, 4, 9, 16]
```

14. What is the difference between range and xrange?

range: Returns a list (Python 2) or range object (Python 3).
xrange: Returns an xrange object (Python 2) that generates numbers on the fly (does not exist in Python 3).

Python

15.Which is faster, Python list or Numpy arrays, and why?
NumPy arrays are faster than Python lists because they are implemented in C and optimized for numerical operations, allowing for efficient memory usage and faster execution of mathematical functions.

16.What is the difference between a Python list and a tuple?
List: Mutable, can be changed.
Tuple: Immutable, cannot be changed.

17.What are Python sets? Explain some of the properties of sets.
Set: Unordered collection of unique elements.
Properties:
No duplicates.
Unordered.
Mutable, but elements must be immutable.

18.Explain the logical operations in Python
and: Logical AND
or: Logical OR
not: Logical NOT

```
a = True
b = False
print(a and b) # Output: False
print(a or b) # Output: True
print(not a)  # Output: False
```

19. What are immutable and mutable data types?
Immutable: Strings, tuples, integers, floats.
Mutable: Lists, dictionaries, sets.

20.What are global and local variables in Python?
Global Variables: Declared outside of functions and accessible everywhere.
Local Variables: Declared inside a function and accessible only within that function.

21. What is an ordered dictionary?
An Ordered Dictionary (OrderedDict) is a dictionary subclass that remembers the order in which items were inserted.

22. What are lambda functions in Python, and why are they important?
Lambda functions are **small anonymous functions** defined using the **lambda keyword**. They are used for creating small, one-time, and inline functions.

```
add = lambda x, y: x + y
print(add(2, 3)) # Output: 5
```

23.Explain the difference between modules and packages in Python?
Module: A single file containing Python code.
Package: A directory containing multiple modules, typically with an __init__.py file.

24.Explain the difference between list and array.
List: A collection of items that can hold different data types.
Array: Typically used for numerical operations and can hold elements of the same data type.

25.What are the benefits of using function in python
Code reusability.
Modular code structure.
Easier to maintain and debug.

26.Difference between a list ,tuple and dictionary in python
List: Ordered, mutable collection.
Tuple: Ordered, immutable collection.
Dictionary: Unordered collection of key-value pairs.

27.How to use list comprehension to simplify code that generate a list of square for all even number from 0 to 10?
```
squares = [x**2 for x in range(11) if x % 2 == 0]
print(squares) # Output: [0, 4, 16, 36, 64, 100]
```

28.What are data structure we have in python?
Lists, tuples, dictionaries, sets, arrays, strings.

29.Convert string to list
```
string = "hello"
string_list = list(string)
print(string_list) # Output: ['h', 'e', 'l', 'l', 'o']
```

30.What is class and objection python?
Class: Blueprint for creating objects.
Object: Instance of a class
```
class MyClass:
    def __init__(self, name):
        self.name = name
obj = MyClass("John")
print(obj.name) # Output: John
```

31.What are built in types available in python?
Integers, floats, strings, lists, tuples, sets, dictionaries, booleans, NoneType.

32.What is negative indexing in python?
Negative indexing allows access to elements from the end of a list.

```
my_list = [1, 2, 3, 4]
print(my_list[-1]) # Output: 4
```

33.Explain errors in python
SyntaxError, TypeError, IndexError, KeyError, ValueError, AttributeError, ImportError.

34.Explain oops concept in python
Class and Objects, Inheritance, Polymorphism, Encapsulation, Abstraction

35.Prime

```
def is_prime(n):
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True
```

36.Factorial

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)
```

37.Square root

```
import math
number = 16
sqrt = math.sqrt(number)
print(sqrt) # Output: 4.0
```

38.Leap year

```
def is_leap_year(year):
    return (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0)
```

39.Palindrome

```
def is_palindrome(s):
    return s == s[::-1]
```

40.List comphrension

A concise way to create lists using a single line of code with an expression and a loop.

41.Which is faster, python list or Numpy arrays, and why?

NumPy arrays are faster due to efficient memory storage and optimized C-based operations.

42.What is the difference between a python list and a tuple?

List: Mutable, dynamic size, slower.
Tuple: Immutable, fixed size, faster.

43.What are python sets? Explain some of the properties of sets.

A set in Python is an unordered collection of unique elements. Sets are defined using curly braces {} or the set() function. They are useful for storing items where duplication is not desired and for performing various mathematical set operations like union, intersection, difference, and symmetric difference.

44. Explain the top 5 functions used for python strings.

len(): Get length of string.
str(): Convert to string.
upper(): Convert to uppercase.
lower(): Convert to lowercase.
split(): Split string into list.

45.What are immutable and mutable data types?
Immutable: Data types that cannot be changed after creation.
Examples include:
Strings: str
Tuples: tuple
Integers: int
Floats: float
Mutable: Data types that can be changed after creation. Examples include:

- Lists: list
- Dictionaries: dict
- Sets: set

46.What are python functions, and how do they help in code optimization?
Functions in Python are **reusable blocks of code** designed to perform a specific task. They help in code optimization by:
Reducing redundancy: Write once, use many times.
Enhancing readability and maintainability: Clear separation of functionality.Facilitating debugging and testing: Isolated blocks of functionality.
47.What is an ordered dictionary?
An Ordered Dictionary (OrderedDict) is a dictionary subclass that **maintains the order** in which items are inserted. It is part of the collections module.

48.What are lambda functions in python, and why are they important?
Lambda functions are **small anonymous functions** defined with the **lambda keyword**. They are important for:
Writing concise and readable code. Defining simple functions inline without formally declaring them.

49.Find the second-highest salary
```
import pandas as pd
# Sample data
data = {'Employee': ['A', 'B', 'C', 'D'],
      'Salary': [1000, 2000, 3000, 4000]}
```

```
df = pd.DataFrame(data)
# Find the second-highest salary
second_highest_salary = df['Salary'].nlargest(2).iloc[-1]
print(second_highest_salary) # Output: 3000
```

50. Find the duplicate emails

```
import pandas as pd
# Sample data
data = {'Email': ['a@example.com', 'b@example.com',
'a@example.com', 'c@example.com']}
df = pd.DataFrame(data)
# Find duplicate emails
duplicates = df['Email'].value_counts()[df['Email'].value_counts() > 1]
print(duplicates)
```

51.Given a dataset of test scores, write pandas code to return the cumulative percentage of students that received scores within the buckets of <50, <75, <90, <100.

```
import pandas as pd
# Sample data
data = {'Score': [10, 20, 40, 50, 70, 80, 90, 100]}
df = pd.DataFrame(data)
# Define the bins and labels
bins = [0, 50, 75, 90, 100]
labels = ['<50', '<75', '<90', '<100']
# Create a new column with the bucket labels
df['Bucket'] = pd.cut(df['Score'], bins=bins, labels=labels, right=False)
# Calculate cumulative percentage
cumulative_percentage =
df['Bucket'].value_counts(normalize=True).sort_index().cumsum() *
100
print(cumulative_percentage)
```

52.What is the difference between append() and extend() methods?

```
l = [1, 2, 3]
l.append([4, 5])
print(l) # Output: [1, 2, 3, [4, 5]]
l = [1, 2, 3]
l.extend([4, 5])
print(l) # Output: [1, 2, 3, 4, 5]
```

53.What is the output of the following? x = [‘ab’, ‘cd’]

```
print(len(list(map(list, x))))
Output: 2
```

54.How would you create an empty NumPy array?

```
import numpy as np
empty_array = np.array([])
print(empty_array) # Output: []
```

55.What is the default missing value marker in pandas, and how can you detect all missing values in a DataFrame?

```
import pandas as pd
import numpy as np
# Sample DataFrame with missing values
df = pd.DataFrame({'A': [1, 2, np.nan], 'B': [4, np.nan, 6]})
# Detect missing values
missing_values = df.isnull()
print(missing_values)
```

56. How will you create a series from dict in Pandas?
import pandas as pd

```
data = {'a': 1, 'b': 2, 'c': 3}
series = pd.Series(data)
print(series)
```

57.How will you create an empty DataFrame in Pandas?

```
import pandas as pd
empty_df = pd.DataFrame()
print(empty_df)
```

58.How to get frequency counts of unique items of a series?

```
import pandas as pd
series = pd.Series([1, 2, 2, 3, 3, 3])
frequency_counts = series.value_counts()
print(frequency_counts)
```

59.How to convert a numpy array to a dataframe of given shape?

```
import numpy as np
import pandas as pd
# Example NumPy array
array = np.array([[1, 2, 3], [4, 5, 6]])
# Convert to DataFrame
df = pd.DataFrame(array, columns=['A', 'B', 'C'])
print(df)
```

60.What is Pandas Index?

A Pandas Index is an **immutable array-like object** that labels the axes (rows and columns) of a DataFrame or Series. It helps in accessing data, aligning data, and performing operations like indexing and slicing.

61.Define GroupBy in Pandas?

```
import pandas as pd
# Sample DataFrame
df = pd.DataFrame({'A': ['foo', 'bar', 'foo', 'bar'], 'B': [1, 2, 3, 4]})
# Group by column 'A' and calculate the sum of 'B' for each group
grouped = df.groupby('A').sum()
print(grouped)
```

62.How to convert the index of a series into a column of a dataframe?

```
import pandas as pd
# Sample Series
series = pd.Series([1, 2, 3], index=['a', 'b', 'c'])
# Convert index to column
df = series.reset_index()
print(df)
```

63.Tuple Vs list Vs dictionary

Tuple: Immutable, ordered collection. Cannot be modified after creation

List: Mutable, ordered collection. Can be modified after creation.

Dictionary: Mutable, unordered collection of key-value pairs. Keys must be unique.

64.How would you reverse a numpy array?

```
import numpy as np
array = np.array([1, 2, 3, 4])
reversed_array = array[::-1]
print(reversed_array) # Output: [4, 3, 2, 1]
```

65.What is the difference between mutation and crossover?

Mutation: In genetic algorithms, mutation involves randomly altering the value of a part of an individual solution to introduce genetic diversity.

Crossover: In genetic algorithms, crossover involves combining the genetic information of two parent solutions to generate new offspring.

66.What is difference between range and xrange function in python?

```
for i in range(5):
  print(i)
for i in xrange(5):
  print(i)
for i in xrange(5):
  print(i)
```

67.Explain what is multi-indexing in pandas?

```
import pandas as pd
arrays = [[1, 1, 2, 2], ['a', 'b', 'a', 'b']]
index = pd.MultiIndex.from_arrays(arrays, names=('number',
'letter'))
df = pd.DataFrame({'value': [1, 2, 3, 4]}, index=index)
print(df)
```

68.How would you iterate over rows in a dataframe in pandas?

```
import pandas as pd
df = pd.DataFrame({'A': [1, 2], 'B': [3, 4]})
for index, row in df.iterrows():
 print(index, row['A'], row['B'])
for row in df.itertuples(index=True, name='Pandas'):
 print(row.Index, row.A, row.B)
```