1.What is garbage collection in the context of Python, and why is it important? Can you explain how memory management is handled in Python?

- Garbage collection is Python's way of automatically managing memory, ensuring that your applications runs smoothly by freeing up memory that is no longer in use , it does by preventing memory leaks, optimize performance and simplify development.

2.What are the key differences between NumPy arrays and Python lists, and can you explain the advantages of using NumPy arrays in numerical computations?

- Using NumPy over traditional Python lists is one of the added advantages the speed boost it provides for numerical computations. NumPy operations are implemented in C, which makes them significantly faster compared to equivalent operations performed on Python lists.
- This speed boost is particularly noticeable when dealing with large datasets or when performing complex mathematical operations. NumPy's underlying implementation allows it to efficiently utilize hardware resources and optimize memory usage, resulting in faster execution times.
- NumPy is designed for vectorized operations, meaning operations on entire arrays can be performed without explicit loops. This not only leads to more concise and easier-to-read code but also significantly boosts speed due to internal optimizations.
- NumPy can automatically expand the dimensions of arrays during arithmetic operations, making it easier to perform calculations on arrays of different shapes.

3.How does list comprehension work in Python, and can you provide an example of using it to generate a list of squared values or filter a list based on a condition?

- List comprehension in Python is a concise way to create lists by iterating over an iterable and optionally applying a condition. It consists of an expression followed by a for clause, and can also include an if clause to filter items.

- expression is a calculation performed on an element. The element resulting from the expression appends to the newly created list.
- element is an item extracted from an iterable on which the expression applies.
- iterable is an iterable from which the elements are extracted.
- condition is an optional check whether the element meets the provided condition.

Example,uses a **for** loop to generate a list of squares between 0 and 5

```
squares = []

for number in range (0, 5):
```

```
 squares.append(Number**2)

print(squares)
```

Example for  using list comprehension:

```
squares = [number**2 for number in range(0, 5)]

print(squares)
```

4.Can you explain the concepts of shallow and deep copying in Python, including when each is appropriate, and how deep copying is implemented?

- A shallow copy of an object creates a new object, but it does not create copies of the nested objects contained within the original object. Instead, it simply copies references to those nested objects. This means that changes to mutable nested objects in the copied object will affect the original object and vice versa.
- Deep copy is a process in which the copying process occurs recursively. It means first constructing a new collection object and then recursively populating it with copies of the child objects found in the original. In case of deep copy, a copy of object is copied in other object, It means that any changes made to a copy of object do not reflect in the original object. In python, this is implemented using “deepcopy()” function.

#How it is implemented.

```
 # Original object (a list of lists)

original = [[1, 2, 3], [4, 5, 6]]


# Create a deep copy

deep_copied = copy.deepcopy(original)


# Modify the deep copied object

deep_copied[0][0] = 'X'
```

```
print("Original:", original)  # Output: [[1, 2, 3], [4, 5, 6]]

print("Deep Copied:", deep_copied)  # Output: [['X', 2, 3], [4, 5, 6]]
```

5. Explain with examples the difference between list and tuples. cc @channel

Python tuples are a type of data structure that is very similar to lists while a list is a data structure in Python that is a mutable, or changeable, ordered sequence of elements.

- Tuples are immutable objects and lists are mutable objects.
- Once defined, tuples have a fixed length and lists have a dynamic length.
- Tuples use less memory and are faster to access than to lists.
- Tuple syntax uses round brackets or parenthesis, and list syntax uses square brackets.

cc @channel https://app.slack.com/