Ans-1 :-

Differences Between Lists and Tuples in Python

Feature	List (list)	Tuple (tuple)
Mutability	Mutable (can be modified)	Immutable (cannot be modified after creation)
Performance	Slower (because of dynamic resizing and mutability overhead)	Faster (optimized due to immutability)
Memory Usage	Higher (extra memory for modifications)	Lower (no modification overhead)
Methods Available	Many methods (append(), remove(), etc.)	Fewer methods (count(), index())
Iteration Speed	Slower (due to dynamic nature)	Faster (fixed structure)
Use Case	When modifications are needed (e.g., dynamic data structures)	When values should remain constant (e.g., fixed configurations, dictionary keys)

Performance

- **Lists: Generally have a slight performance overhead because they need to accommodate potential changes in size and element values.**
- **Tuples: Because they are immutable, they can be more efficient in terms of memory and performance. This immutability allows Python to optimize tuples better than lists.**

Choosing Between Lists and Tuples

- **Choose lists when you need:**
  - **Flexibility to modify the collection.**
  - **Frequent changes to the data, such as appending or removing elements.**
- **Choose tuples when you need:**
  - **Data to remain unchanged.**
  - **Better performance and memory efficiency.**
  - **To use the collection as a dictionary key.**

Ans-2 :-

Type Conversions in Python

Conversion Type	Function	Example	Result
Int → Float	float(x)	float(5)	5.0
Float → Int	int(x)	int(3.9)	3
Number → String	str(x)	str(100)	"100"
String → Int	int(x)	int("42")	42
String → Float	float(x)	float("3.14")	3.14
String → List	list(x)	list("abc")	['a', 'b', 'c']
List → String	"".join(list)	"".join(['a', 'b', 'c'])	"abc"