**EX.NO: 2 DATE:**

# Apply Python built-in data types: Strings, List, Tuples, Dictionary, Set and their methods to solve any given problem.

**AIM:**

To apply Python built-in data types (String, List, Tuple, Dictionary, and Set) and their respective methods to solve a given problem.

# PROCEDURE:

**Problem Statement:** Given a list of students' names and their respective marks, perform the following operations:

* Find and display the student with the highest marks.
* Display all students whose marks are above 75.
* Remove duplicate names from the list and display the unique names.
* Concatenate all student names into a single string separated by commas.
* Sort the students by their marks in descending order and display the sorted list.

## Creation of Data Types:

* **String:** To concatenate student names.
* **List:** To store the list of student names and marks.
* **Tuple:** To store each student’s name and marks as a tuple.
* **Dictionary:** To store student names as keys and their marks as values.
* **Set:** To remove duplicates from the student list.
* ​

## Operations using Methods:

**List Methods:**

* max() to find the student with the highest marks.
* List comprehension to filter students with marks above 75.
* sort() to sort the list based on marks.
* Set Methods:
* set() to remove duplicates from the student list.

## String Methods:

* join() to concatenate all student names.

## Dictionary Methods:

* Use dictionary methods like items() to access keys and values.

# PROGRAM:

## List of tuples with student names and their marks

students = [ ("Alice", 85),

("Bob", 67),

("Charlie", 92),

("David", 76),

("Alice", 85), Duplicate entry ("Eva", 88)

]

## 1. Dictionary Creation (Student names as keys, marks as values)

student\_dict = {name: marks for name, marks in students}

## 2. Find the student with the highest marks

highest\_student = max(student\_dict, key=student\_dict.get)

print(f"Student with highest marks: {highest\_student}, Marks:

{student\_dict[highest\_student]}")

## 3. Display all students with marks above 75

students\_above\_75 = [name for name, marks in student\_dict.items() if marks > 75] print(f"Students with marks above 75: {students\_above\_75}")

**4. Remove duplicate names (if any) using set** unique\_students = set([name for name, marks in students]) print(f"Unique student names: {unique\_students}")

## 5. Concatenate all student names into a single string separated by commas

student\_names = ', '.join(unique\_students) print(f"Concatenated student names: {student\_names}")

**6. Sort the students by their marks in descending order** sorted\_students = sorted(students, key=lambda x: x[1], reverse=True) print("Students sorted by marks (descending):")

for name, marks in sorted\_students: print(f"{name}: {marks}")

# OUTPUT:

**Student with highest marks:** Charlie, Marks: 92

**Students with marks above 75:** ['Alice', 'Charlie', 'David', 'Eva'] **Unique student names:** {'Eva', 'David', 'Charlie', 'Bob', 'Alice'} **Concatenated student names:** Eva, David, Charlie, Bob, Alice **Students sorted by marks (descending):**

## Charlie: 92

**Eva:** 88

**Alice:** 85

**David:** 76

**Bob:** 67

# RESULT:

The problem was successfully solved by applying various Python built-in data types and their methods. The highest-scoring student was identified, students with marks above 75 were listed, duplicates were removed, student names were concatenated, and students were sorted by their marks in descending order.