Real-Life Python Problems with Solutions (Tuple-Based)

Problem: Store student details (name, age, grade) using tuples and display them.

```
students = [("Alice", 20, "A"), ("Bob", 19, "B")]
for s in students:
    print(f"Name: {s[0]}, Age: {s[1]}, Grade: {s[2]}")
```

Problem: Store GPS coordinates as tuples and calculate distance between two.

```
import math
coord1 = (12.97, 77.59)
coord2 = (28.61, 77.20)
distance = math.sqrt((coord2[0]-coord1[0])**2 + (coord2[1]-coord1[1])**2)
print("Distance:", distance)
```

Problem: Store (account_no, amount, type) in tuples and extract only deposits.

```
transactions = [(101, 2000, "deposit"), (102, 1500, "withdraw"), (103, 3000, "deposit")]
deposits = [t for t in transactions if t[2] == "deposit"]
print(deposits)
```

Problem: Each tuple has (day, temperature). Print the hottest day.

```
temps = [("Mon", 34), ("Tue", 38), ("Wed", 36)]
print("Hottest Day:", max(temps, key=lambda x: x[1]))
```

Problem: Extract all flight destinations from a list of flight info tuples.

```
flights = [("DEL", "Mumbai"), ("BLR", "Chennai")]
destinations = [f[1] for f in flights]
print(destinations)
```

Problem: Sort employee tuples based on salary.

```
employees = [("John", 50000), ("Anna", 60000)]
print(sorted(employees, key=lambda x: x[1]))
```

Problem: Tuple of (student_name, subject, score), find highest scorer per subject.

```
records = [("Alice", "Math", 92), ("Bob", "Math", 85), ("Alice", "Sci", 90)]
subjects = {}
for name, subject, score in records:
    if subject not in subjects or score > subjects[subject][1]:
        subjects[subject] = (name, score)
print(subjects)
```

Problem: Use a tuple to represent a config setting and explain immutability.

```
config = ("localhost", 8080)
print("Immutable config:", config)
```

Problem: Given (city, STD code), find city for a given code.

```
cities = [("Delhi", "011"), ("Mumbai", "022")]
code = "011"
print(next((c[0] for c in cities if c[1] == code), "Not Found"))
```

Problem: From tuple (product_id, name, price), display affordable products.

```
products = [(1, "Pen", 10), (2, "Book", 150)]
affordable = [p for p in products if p[2] <= 100]
print(affordable)
```

Problem: Get all movie titles released after 2010.

```
movies = [("Movie1", 2012), ("Movie2", 2008)]
recent = [m[0] for m in movies if m[1] > 2010]
print(recent)
```

Problem: Store contacts as (name, number) and search by name.

```
contacts = [("Alice", "98765"), ("Bob", "87654")]
name = "Alice"
print(next((c[1] for c in contacts if c[0] == name), "Not Found"))
```