Answers to Practice Problems — Lesson 3

Solutions

1) square(x) and test with 3 numbers

```
def square(x):
    return x * x

print(square(2))
print(square(-3))
print(square(5))
```

Output

```
4
9
25
```

2) max_of_two(a, b) (no built-in max)

```
def max_of_two(a, b):
    if a >= b:
        return a
    else:
        return b

print(max_of_two(7, 3)) # 7
print(max_of_two(5, 5)) # 5
print(max_of_two(-2, 4)) # 4
```

Output

```
7
5
4
```

3) affordable(price, budget) + test with a loop

```
def affordable(price, budget):
    return price <= budget
```

```
prices = [4, 12, 7, 3]
budget = 8

for p in prices:
    print(p, "->", affordable(p, budget))
```

Output

```
4 -> True
12 -> False
7 -> True
3 -> True
```

4) **route_score(time, fun)** + pick best route

```
def route_score(time, fun):
    return fun - 0.5 * time

routes = [(10, 6), (14, 9), (8, 5)] # (time, fun)
best = None
best_s = -float('inf')

for t, f in routes:
    s = route_score(t, f)
    print("Route:", (t, f), "score =", s)
    if s > best_s:
        best_s = s
        best = (t, f)

print("Best route:", best, "with score =", best_s)
```

Output

```
Route: (10, 6) score = 1.0
Route: (14, 9) score = 2.0
Route: (8, 5) score = 1.0
Best route: (14, 9) with score = 2.0
```

5) (Challenge) **best_item(items, budget)**

```
def best_item(items, budget):
    """
    items: list of (price, fun)
    Returns: (price, fun) with the highest fun under budget.
    Tie-breaker: prefer lower price; if still tied, keep the first seen.
    """
    best = None
    best_fun = -1
    best_price = float('inf')

    for price, fun in items:
        if price <= budget:
            if (fun > best_fun) or (fun == best_fun and price < best_price):
                best = (price, fun)
                best_fun = fun
                best_price = price
    return best

items = [(4, 5), (6, 7), (3, 5), (8, 9)]
budget = 6

ans = best_item(items, budget)
print("Best affordable item:", ans)
```

Output

```
Best affordable item: (6, 7)
```