# Exercises: Lambda Functions and Arrays in Python

These exercises help you practice Python's lambda functions and arrays. Each task is designed to enhance your understanding of applying lambda expressions and working with arrays.

## Exercises:

1. Create a lambda function to find the cube of a number. Use it with the `map()` function to compute the cubes of elements in the array `[2, 4, 6, 8, 10]`.

2. Use a lambda function with the `filter()` function to extract elements greater than 5 from the array `[3, 5, 7, 9, 11]`.

3. Create a lambda function to add two numbers. Use it with `reduce()` from the `functools` module to find the sum of all elements in the array `[1, 2, 3, 4, 5]`.

4. Sort the array `[(3, 4), (1, 2), (5, 0)]` based on the second element of each tuple using a lambda function.

5. Reverse an array `[10, 20, 30, 40, 50]` using the `reverse()` method and print the result.

6. Use a lambda function within `map()` to calculate the square root of elements in the array `[4, 16, 36, 64, 100]`.

7. Filter all even numbers from an array `[11, 22, 33, 44, 55]` using a lambda function with `filter()`.

8. Create an array of integers `[2, 3, 5, 7, 11, 13]` and find its length using the `len()` function.

9. Use a lambda function to check if a number is a multiple of 3. Apply it to the array `[3, 6, 9, 12, 15]` using `filter()` and print the filtered numbers.

10. Create an array `[1, 2, 3, 4, 5]` and append the value `6` to it. Then, remove the last element using the `pop()` method.

## Hints:

- Use `lambda` functions to create small anonymous functions.  
- Use `map()` for element-wise operations.  
- Use `filter()` to extract elements that satisfy a condition.  
- Use the `functools.reduce()` method for cumulative operations.  
- Use the `array` module for array operations.