1. What is the big-O simplification of:
   \[ f(n) = 3n^2 + 4n + 2 \]
   A  O(n)
   B  O(n^2)
   C  O(n^2 + n)
   D  O(3n^2 + 4n + 2)

2. What is the big-O simplification of:
   \[ f(n) = 3n^2 + 4n + 2 \]
   A  O(n)
   B  O(n^2)
   C  O(n^2 + n)
   D  O(3n^2 + 4n + 2)

3. What is the time complexity of the Python max function, e.g.
   ```python
   numbers = list(range(100))
   the_max = max(numbers)
   ```
   A  O(1)
   B  O(100)
   C  O(n)
   D  O(n^2)

4. What is the time complexity of the Python max function, e.g.
   ```python
   numbers = list(range(100))
   the_max = max(numbers)
   ```
   A  O(1)
   B  O(100)
   C  O(n)
   D  O(n^2)
5. In Python you can implement matrix multiply on two nxn matrices (stored as lists of lists) with the following algorithm.

```python
for i in range(n):
    for j in range(n):
        val = 0.0
        for k in range(n):
            val += x[i][k] * y[k][j]
        res[i][j] = val
```

What is the time complexity of this algorithm?

- **A** O(1)
- **B** O(log n)
- **C** O(n)
- **D** O(n^2)
- **E** O(n^3)

6. In Python you can implement matrix multiply on two nxn matrices (stored as lists of lists) with the following algorithm.

```python
for i in range(n):
    for j in range(n):
        val = 0.0
        for k in range(n):
            val += x[i][k] * y[k][j]
        res[i][j] = val
```

What is the time complexity of this algorithm?

- **A** O(1)
- **B** O(log n)
- **C** O(n)
- **D** O(n^2)
- **E** O(n^3)

The PDF unfortunately squashes the indentation, but the three loops are nested, thus we will do n*n*n multiplications, or O(n^3)