**CSCI316: Week 1.5 - Programming Basics for Big Data**

---

## 📄Overview

Big Data programming in this subject leverages **Python 3** due to its simplicity and strong ecosystem of data science libraries.

---

## 🔧Why Python?

- Easy to learn and read
- Acts like "executable pseudocode"
- Less verbose than Java or C/C++
- Excellent for prototyping and scaling
- Extensive support for:
- Data analytics (Pandas, NumPy)
- Visualization (Matplotlib)
- Machine Learning (Scikit-Learn)
- Big Data (PySpark)
- Deep Learning (TensorFlow)

---

## 🕰️Platforms and Libraries

| Category | Tools |
|---|---|
| Core Language | Python 3 |
| Scientific Computing | NumPy |
| Data Analytics | Pandas |
| Visualization | Matplotlib |
| Machine Learning | Scikit-Learn |
| Big Data Processing | Apache Spark (PySpark) |
| Deep Learning | TensorFlow |

---

## ⚙️Implementation Levels

1. **Level-1**: Implement ML algorithms from scratch (no libraries)
2. **Level-2**: Use libraries (Scikit-Learn, PySpark, TensorFlow)

---

## 📚Python Core Crash Course

### Indentation

```
for i in [1, 2, 3]:
    print(i)
    for j in [4, 5]:
        print(i + j)
print("done looping")
```

### Whitespace Ignored Inside Brackets

```
nums = (1 + 2 + 3 +
        4 + 5)
```

### Imports

```
import math
from math import ceil
from math import *  # not recommended
```

### Functions

```
def double(x):
    return x * 2
sum = lambda x, y: x + y
```

### Strings

```
s1 = "hello"
s2 = 'world'
multiline = """Line1\nLine2"""
```

### Lists

```
x = [0, 1, 2]
x.append(3)
x[:2]  # slicing
```

### Tuples

```
t = (1, 2)
def add_mul(x, y):
    return (x+y, x*y)
```

### Dictionaries

```
grades = {"Alice": 90, "Bob": 85}
grades.get("Charlie", 0)
```

### Sets

```
s = set([1, 2, 2])  # {1, 2}
s.add(3)
```

### Control Flow

```
if x > 0:
    print("Positive")
elif x == 0:
    print("Zero")
else:
    print("Negative")
```

### Sorting

```
sorted([4,1,2])
sorted(data, key=lambda x: x[1])
```

### List Comprehensions

```
evens = [x for x in range(10) if x % 2 == 0]
squares = [x*x for x in range(5)]
```

### Map and Filter

```
list(map(lambda x: x**2, range(5)))
list(filter(lambda x: x > 0, range(-3, 3)))
```

### OOP in Python

```python
class Set:
    def __init__(self):
        self.data = {}
    def add(self, value):
        self.data[value] = True
```

### Using the class

```python
s = Set()
s.add(1)
```

---

## 🪬NumPy Basics

```python
import numpy as np
arr = np.array([1, 2, 3])
arr.shape  # (3,)
arr[0] = 4
```

### 2D Arrays

```python
matrix = np.array([[1,2],[3,4]])
matrix.T  # transpose
np.sum(matrix, axis=0)  # column-wise sum
```

### Element-wise Operations

```python
np.add(a, b)
np.multiply(a, b)
np.sqrt(a)
```

### Dot Products

```python
np.dot(a, b)  # vectors or matrices
```

---

This covers the essential **Python programming skills** you'll need before diving into Scikit-Learn, TensorFlow, and Spark. Let me know if you'd like exercises or coding labs to reinforce these concepts.