**Python Programming and Databricks**

**Tasks**

**1. FOR Loop**

Iterate through a list and print elements.

Calculate the sum of numbers within a range.

Generate a multiplication table using nested loops.

A screenshot of a computer program

Description automatically generated A table of multiplication with numbers

Description automatically generated

**2. IF Condition**

Check if a number is positive, negative, or zero:

A screenshot of a computer program

Description automatically generated

Compare two numbers and determine the larger one.

A screenshot of a computer program

Description automatically generated Categorize a score into grades using conditional statements.

A screenshot of a computer

Description automatically generated

**3. BREAK and PASS**

Exit a loop upon encountering a specific value.

A screenshot of a computer

Description automatically generated

Skip a condition within a loop while continuing execution

A screenshot of a computer

Description automatically generated

**Combine BREAK and PASS for advanced loop control**

A screenshot of a computer

Description automatically generated

**Functions**

Write a function to calculate the factorial of a number. A screenshot of a computer

Description automatically generated

2. **Reverse a string using a defined function:** A screenshot of a computer program

Description automatically generated 3. **Identify the maximum value in a list through a function** A screenshot of a computer

Description automatically generated

**5. Date in Databricks**

Retrieve Current Timestamp

A screenshot of a computer

Description automatically generated **Extract components like day, month, and year from a date column:** A screenshot of a computer

Description automatically generated **Add days to a date column and create a new column:**

A screenshot of a computer

Description automatically generated**Filter**

1. **Filter rows in a DataFrame based on a single condition:**

### You can use the filter() or where() method to filter rows based on a single condition. A screenshot of a computer Description automatically generated **2. Apply multiple conditions to filter rows:**

You can combine multiple conditions using the logical operators & (AND), | (OR), and ~ (NOT).

A screenshot of a computer

Description automatically generated **Filter rows matching a specific string value:** **A screenshot of a computer

Description automatically generated**

**7. New Column**

### 1. **Create a column based on arithmetic operations:** **A screenshot of a computer program Description automatically generated** 2. **Categorize data into groups using conditional logic:**

You can categorize data into groups based on conditions using when() and otherwise() from pyspark.sql.functions.

### A screenshot of a computer Description automatically generated 3. **Add a column with transformed string values:**

You can use string functions to modify the values in an existing column and create a new column based on the transformation. For example, converting string values to uppercase.

### A screenshot of a computer program Description automatically generated 8. **Literals ('DATAFLOW')**

#### 1. **Create a DataFrame using a string literal:**

You can create a DataFrame using a string literal by passing a list of tuples with the string values to the createDataFrame() function.

#### A screenshot of a computer program Description automatically generated 2. **Filter rows based on a specific string literal:**

You can filter rows based on a string literal using filter() or where().

#### A screenshot of a computer Description automatically generated 3. **Insert a string literal as a constant value in a new column:**

You can insert a string literal as a constant value in a new column using withColumn().

A screenshot of a computer

Description automatically generated

### 9. **Joins**

#### 1. **Perform inner, left, and full outer joins on DataFrames:**

Assume we have two DataFrames, df1 and df2, that share a common column to join on.

#### A screenshot of a computer program Description automatically generated A screenshot of a computer Description automatically generated 2. **Merge DataFrames while handling null values:**

When performing joins, you may encounter null values for non-matching rows. You can handle these null values by using fillna() or coalesce() to replace null with a default value.

A screenshot of a computer

Description automatically generated