Question 1: Spark Program on Employee Dataset

Dataset contains employee information. Assume the dataset includes a Salary column (or Salary derived from PaymentTier).

```
from pyspark.sql import SparkSession
from pyspark.sql.functions import col

# Create Spark session
spark = SparkSession.builder.appName("EmployeeProcessing").getOrCreate()

# Read CSV file
df = spark.read.csv("employees.csv", header=True, inferSchema=True)

# Filter employees with salary > 50,000
filtered_df = df.filter(col("Salary") > 50000)

# Increase salary by 10%
updated_df = filtered_df.withColumn(
    "UpdatedSalary",
    col("Salary") * 1.10
)

# Show top 5 highest salaries
updated_df.orderBy(col("UpdatedSalary").desc()).show(5)

# Count qualifying employees
count = updated_df.count()
print("Total qualifying employees:", count)
```

```
+----------+------+----------+---------------+------+-------+-----------------+---------------+------------------+
|First Name|Gender|Start Date|Last Login Time|Salary|Bonus %|Senior Management|           Team|     UpdatedSalary|
+----------+------+----------+---------------+------+-------+-----------------+---------------+------------------+
| Katherine|Female| 8/13/1996|       12:21 AM|149908| 18.912|            false|        Finance|164898.80000000002|
|      Rose|Female| 5/28/2015|        8:40 AM|149903|   5.63|            false|Human Resources|164893.30000000002|
|   Cynthia|Female| 7/12/2006|        8:55 AM|149684|  7.864|            false|        Product|164652.40000000002|
|      NULL|Female| 2/23/2005|        9:50 PM|149654|  1.825|             NULL|          Sales|164619.40000000002|
|     Kathy|Female| 3/18/2000|        7:26 PM|149563| 16.991|             true|        Finance|164519.30000000002|
+----------+------+----------+---------------+------+-------+-----------------+---------------+------------------+
only showing top 5 rows
Total qualifying employees: 854
```

Question 2: Pair RDD Operations

```
# Input Data
data = [("A",10), ("B",20), ("A",30), ("B",40), ("C",50)]
rdd = spark.sparkContext.parallelize(data)

# (a) Total value per key
total_per_key = rdd.reduceByKey(lambda x, y: x + y)

# (b) Average value per key
avg_per_key = rdd.mapValues(lambda x: (x,1)) \
                 .reduceByKey(lambda a,b: (a[0]+b[0], a[1]+b[1])) \
                 .mapValues(lambda x: x[0]/x[1])

# (c) Sorted by key
avg_per_key.sortByKey().collect()
```

```
[('A', 20.0), ('B', 30.0), ('C', 50.0)]
```

Question 3: Department Marks using Spark

```
# Input Format
data = [("CS",80), ("AI",90), ("IT",70), ("IT",85), ("EE",75)]
rdd = spark.sparkContext.parallelize(data)

# (a) Max marks per department
max_marks = rdd.reduceByKey(lambda x, y: max(x, y))

# (b) Average marks per department
avg_marks = rdd.mapValues(lambda x:(x,1)) \
               .reduceByKey(lambda a,b:(a[0]+b[0], a[1]+b[1])) \
               .mapValues(lambda x:x[0]/x[1])

# (c) Departments with average > 75
result = avg_marks.filter(lambda x: x[1] > 75)

# Actions to display output
print("MAX MARKS:", max_marks.collect())
print("AVG MARKS:", avg_marks.collect())
print("RESULT:", result.collect())
```

```
MAX MARKS: [('CS', 80), ('AI', 90), ('IT', 85), ('EE', 75)]
AVG MARKS: [('CS', 80.0), ('AI', 90.0), ('IT', 77.5), ('EE', 75.0)]
RESULT: [('CS', 80.0), ('AI', 90.0), ('IT', 77.5)]
```

Start coding or generate with AI.

Name – Rohit Kumar MTECH-AI Roll-no:- 25901334 Assignment -1

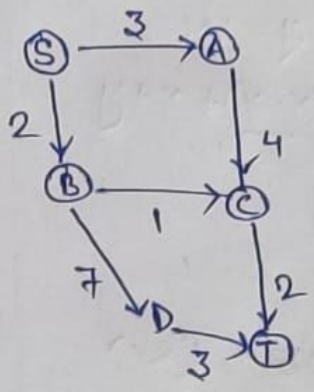
ADEAI

Q.4.) Given :-

S → A (3)
S → B (2)
A → C (4)
B → C (1)
B → D (7)
C → T (2)
D → T (3)

Nodes ↑ Node ↑
Edges weights

→



(i) Shortest path :- (S to T)

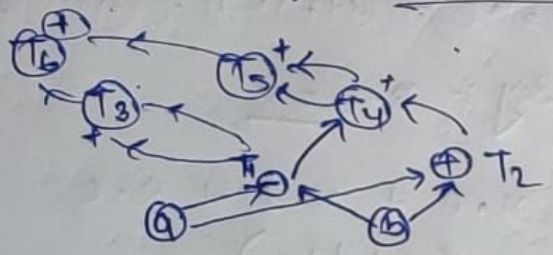
(i) S → B → C → T ⇒ 2+1+2 = 5
(ii) S → A → C → T ⇒ 3+4+2 = 9
(iii) S → B → D → T ⇒ 2+7+3 = 12

} All possible paths from S → T

• Shortest path :- S → B → C → T ⇒ 2+1+2 = 5

Q.5) Construct and optimize the DAG :-

a.) (a-b) * (a-b) + (a-b) * (a+b) + (a+b) * (a-b)



$T_1 = a-b$, $T_2 = a+b$

$T_3 = T_1 * T_1$, $T_4 = T_1 * T_2$

$T_5 = 2 * T_4 = T_4 + T_4$

$T_6 = T_5 + T_3$

So, $T_6 = (a-b)*(a-b) + 2((a-b)*(a+b))$

But when we optimize then, let say $(a-b) = x$, $(a+b) = y$

So, Expression ⇒ $x \cdot x + x \cdot y + y \cdot x$

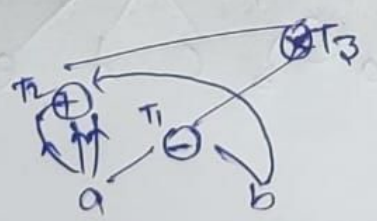
⇒ $x^2 + xy + yx = x^2 + 2xy$

$= x(x+2y)$

(Adj :- $xy = yx$)

So, final Expression ⇒ $(a-b)(3a+b)$

So,



So, $T_1 = a-b$, $T_2 = 3a+b$

$T_3 = T_1 * T_2$

5.b) Given:- Expression

$(a*b+b*c)*(a*b-b*c)+(a*b)+(b*c)$

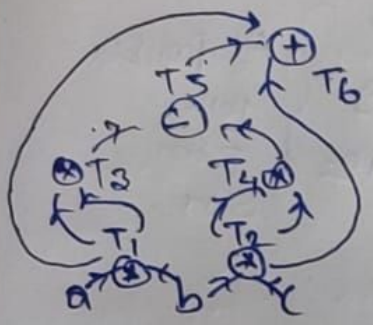
⇒ let say $x=a*b$, $y=b*c$

so, Expression:- $(x+y)(x-y)+x+y$

⇒ $(x^2-y^2)+x+y$

so, final Expression:-

$((a*b)^2-(b*c)^2)+(a*b)+(b*c)$



$T_1 = a*b$

$T_2 = b*c$

$T_3 = T_1 * T_1$

$T_4 = T_2 * T_2$

$T_5 = T_3 - T_4$

$T_6 = T_5 + T_1 + T_2$

5.c) Given:- Expression:-

$(a+b+c)*(a+b)+(a+b)*(b+c)+(a+b+c)$

⇒ let say, as $(a+b)$ appears 2 times

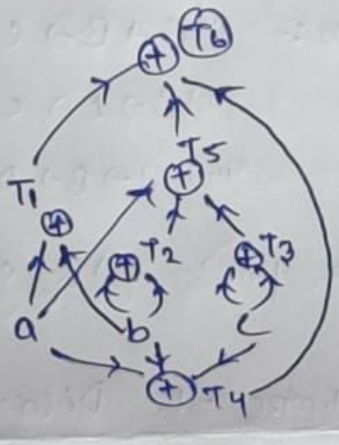
so,

⇒ $(a+b)[(a+b+c)+(b+c)]+(a+b+c)$

⇒ $(a+b)(a+2b+2c)+(a+b+c)$

Final Expression

Then:-



$T_1 = a+b$

$T_2 = b+b$

$T_3 = c+c$

$T_4 = a+b+c$

$T_5 = a+T_2+T_3$

$T_6 = T_1+T_5+T_4$

5.d) $(a-b)*(c-d)+(a-b)+(c-d)/(a-b)$

⇒ so, let say:- $x=a-b$, $y=c-d$

then:-

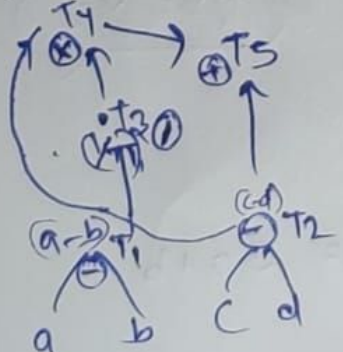
$x \cdot y + x + \frac{y}{x} = xy + \frac{y}{x} + x$

$y\left(x+\frac{1}{x}\right)+x$

Final Expression:-

so, $(c-d)\left((a-b)+\frac{1}{(a-b)}\right)+(c-d)$

⇒ then



$T_1 = a-b$

$T_2 = c-d$

$T_3 = \frac{1}{T_1}$

$T_4 = T_3 * T_2$

$T_5 = T_1 + T_4$

5.e) $((a+b)*c)+((a+b)*d)+((a+b)*e)$

⇒ so, let say :- $a+b=x$

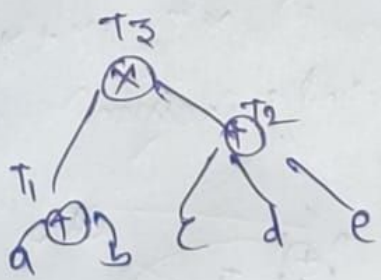
then $x \cdot c + x \cdot d + x \cdot e$

⇒ $x(c+d+e)$

so, final Expression

$(a+b)(c+d+e)$

⇒ then



$T_1 = a+b$

$T_2 = c+d+e$

$T_3 = T_1 * T_2$