ASSIGNMENT – III

TITLE :

Implement Matrix Multiplication Using MapReduce.

NAME : Shinde Shubham Dnyandev. **ROLL NO :** 23107121. **CLASS :** TY-B

PROGRAM :

INPUT :

```
A 1 1 1
A 1 2 2
A 2 1 3
A 2 2 4
B 1 1 5
B 1 2 6
B 2 1 7
B 2 2 8
```

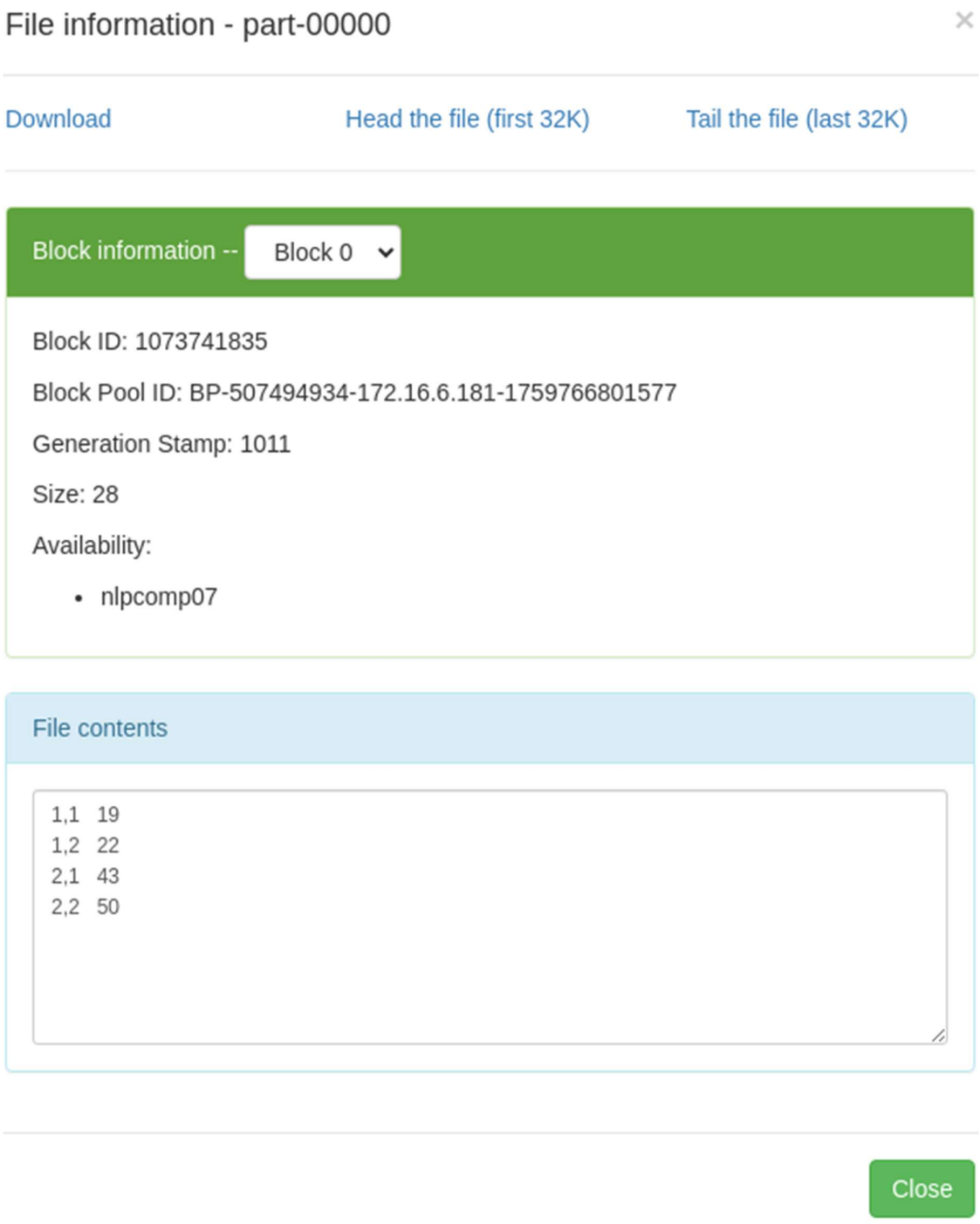
MAPPER :

```
#!/usr/bin/env python3
import sys

# Dimensions of matrices (2x2)
M = 2  # rows in A
N = 2  # columns in A / rows in B
P = 2  # columns in B

for line in sys.stdin:
    line = line.strip()
    if not line:
        continue
    matrix, i, j, value = line.split()
    i, j, value = int(i), int(j), int(value)

    if matrix == 'A':
        # Emit (i,k) as key for k in [1..P]
        for k in range(1, P+1):
            # key: (i,k), value: ('A', j, A[i,j])
            print(f"{i},{k}\tA,{j},{value}")
    elif matrix == 'B':
        # Emit (i,k) as key for i in [1..M], but matrix B has dimension j,k so here
        # We use j as row index in B (j), and k is column index
        # According to formula, emit ((i,k), (B,j,Bjk))
        for i_b in range(1, M+1):
            # key: (i_b,k), value: ('B', j, B[j,k])
            print(f"{i_b},{j}\tB,{i},{value}")
```

REDUCER :

```
#!/usr/bin/env python3
import sys
from collections import defaultdict

current_key = None
A_values = defaultdict(int)  # j -> value
B_values = defaultdict(int)  # j -> value

def emit_result(i, k, total):
    print(f"{i},{k}\t{total}")

for line in sys.stdin:
    line = line.strip()
    if not line:
        continue
    key, value = line.split('\t')
    i, k = key.split(',')
    i, k = int(i), int(k)

    if current_key != (i, k):
        # Process previous key
        if current_key is not None:
            total = 0
            for j in range(1, 3):  # j from 1 to N=2
                total += A_values[j] * B_values[j]
            emit_result(current_key[0], current_key[1], total)
        current_key = (i, k)
        A_values.clear()
        B_values.clear()

    # Parse value
    matrix, j, val = value.split(',')
    j, val = int(j), int(val)

    if matrix == 'A':
        A_values[j] = val
    else:
        B_values[j] = val

# Emit last key
if current_key is not None:
    total = 0
    for j in range(1, 3):
        total += A_values[j] * B_values[j]
    emit_result(current_key[0], current_key[1], total)
```

OUTPUT :


File information - part-00000
Download
Head the file (first 32K)
Tail the file (last 32K)
Block information --
Block 0
Block ID: 1073741835
Block Pool ID: BP-507494934-172.16.6.181-1759766801577
Generation Stamp: 1011
Size: 28
Availability:
nlpcomp07
File contents
1,1 19
1,2 22
2,1 43
2,2 50
Close