Spark - Exercises

Exercise #58

- Full station identification in real-time
- Input:
  - A stream of readings about the status of the stations of a bike sharing system
    - Each reading has the format
      - stationId, # free slots, #used slots, timestamp

Exercise #58

- Output:
  - For each reading with a number of free slots equal to 0
    - print on the standard output timestamp and stationId
  - Emit new results every 2 seconds by considering only the data received in the last 2 seconds

Exercise #59

- Full situation count in real-time
- Input:
  - A stream of readings about the status of the stations of a bike sharing system
    - Each reading has the format
      - stationId,# free slots,#used slots,timestamp

Exercise #59

- Output:
  - For each batch, print on the standard output the number of readings with a number of free slots equal to 0
  - Emit new results every 2 seconds by considering only the data received in the last 2 seconds

Exercise #6o

- Full distinct stations identification in real-time
- Input:
  - A stream of readings about the status of the stations of a bike sharing system
    - Each reading has the format
      - stationId,# free slots,#used slots,timestamp

Exercise #6o

- Output:
  - For each batch, print on the standard output the **distinct stationIds** associated with a reading with a number of free slots equal to 0 in each batch
  - Emit new results every 2 seconds by considering only the data received in the last 2 seconds

Exercise #61

- Maximum number of free slots in real-time
- Input:
  - A stream of readings about the status of the stations of a bike sharing system
    - Each reading has the format
      - stationId, # free slots, #used slots, timestamp

Exercise #61

- Output:
  - For each batch, print on the standard output the maximum value of the field "# free slots" by considering all the readings of the batch (independently of the stationId)
  - Emit new results every 2 seconds by considering only the data received in the last 2 seconds
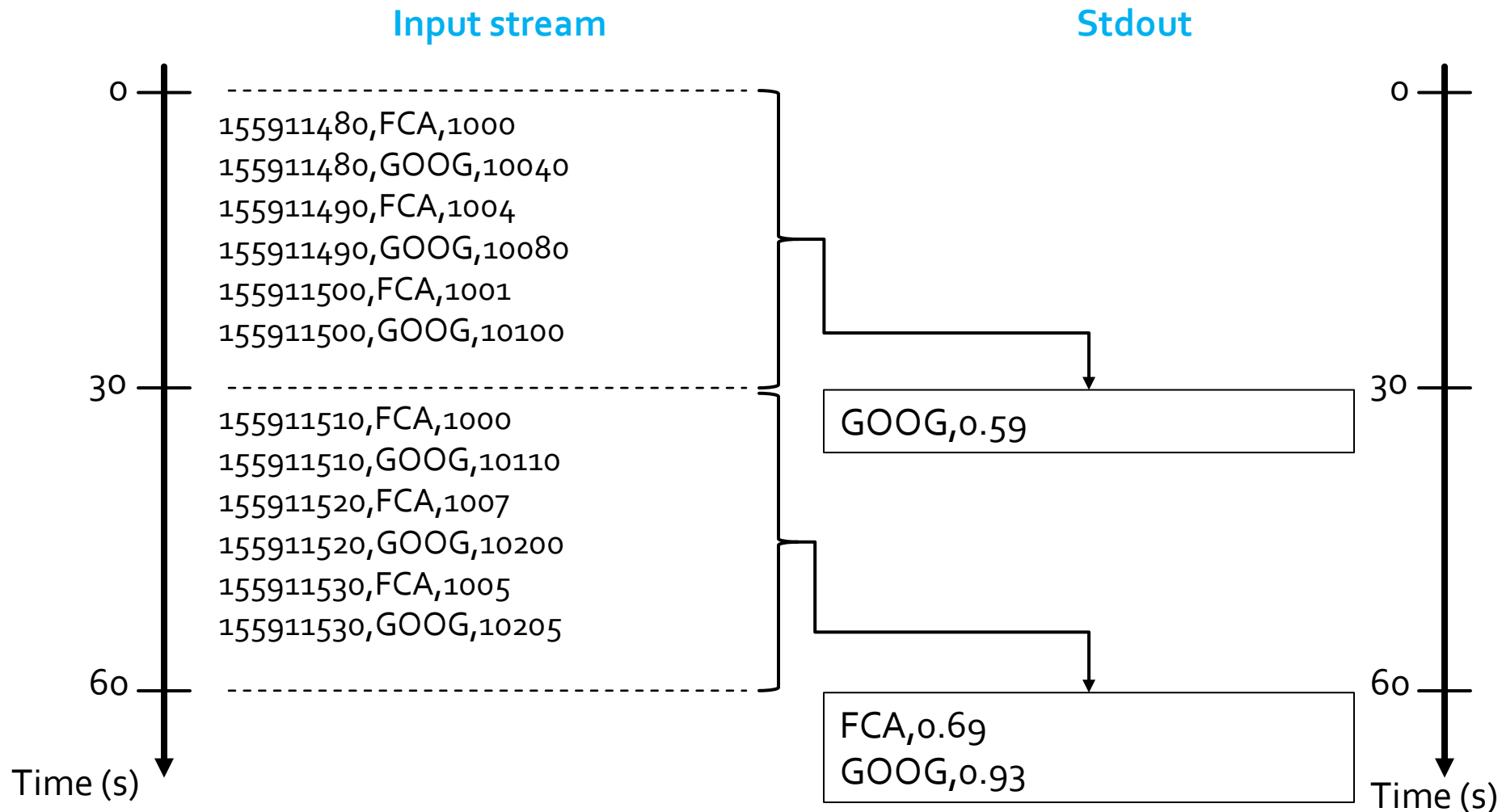
Exercise #62

- High stock price variation identification in real-time
- Input:
  - A stream of stock prices
    - Each input record has the format
      - Timestamp,StockID,Price

Exercise #62

- Output:
  - Every 30 seconds print on the standard output the **StockID and the price variation (%) in the last 30 seconds** of the stocks with a **price variation greater than 0.5%** in the last 30 seconds
    - Given a stock, its price variation during the last 30 seconds is:

$$\frac{max(price)-min(price)}{max(price)}$$

Exercise #62- Example

Input stream

Stdout

0

155911480,FCA,1000
155911480,GOOG,10040
155911490,FCA,1004
155911490,GOOG,10080
155911500,FCA,1001
155911500,GOOG,10100

30

155911510,FCA,1000
155911510,GOOG,10110
155911520,FCA,1007
155911520,GOOG,10200
155911530,FCA,1005
155911530,GOOG,10205

60

Time (s)

0

30

GOOG,0.59

60

FCA,0.69
GOOG,0.93

Time (s)

Exercise #62 Bis

- High stock price variation identification in real-time
- Input:
  - A stream of stock prices
    - Each input record has the format
      - Timestamp,StockID,Price

Exercise #62 Bis

- Output:
  - **Every 30 seconds** print on the standard output the **StockID and the price variation (%)** in the last 60 seconds of the stocks with a **price variation greater than 0.5%** in the last 60 seconds
    - Given a stock, its price variation during the last 60 seconds is:

$$\frac{max(price)-min(price)}{max(price)}$$

Exercise #63

- Full station identification in real-time
- Input:
  - A textual file containing the list of stations of a bike sharing system
    - Each line of the file contains the information about one station

      id\tlongitude\tlatitude\tname

  - A stream of readings about the status of the stations
    - Each reading has the format
      - StationId,# free slots,#used slots,timestamp

Exercise #63

- Output:
  - For each reading with a number of free slots equal to 0
    - print on the standard output timestamp and name of the station
  - Emit new results every 2 seconds by considering only the data received in the last 2 seconds

Exercise #64

- Anomalous stock price identification in real-time
- Input:
  - A textual file containing the historical information about stock prices in the last year
    - Each input record has the format
      - Timestamp,StockID,Price
  - A real time stream of stock prices
    - Each input record has the format
      - Timestamp,StockID,Price

Exercise #64

- Output:
  - Every 1 minute, by considering only the data received in the last 1 minute, print on the standard output the StockIDs of the stocks that satisfy one of the following conditions
    - price of the stock (received on the real-time input data stream) < historical minimum price of that stock (based only on the historical file)
    - price of the stock (received on the real-time input data stream) > historical maximum price of that stock (based only on the historical file)
  - If a stock satisfies the conditions multiple times in the same batch, return the stockId only one time for each batch

Exercise #64- Example

- Textual file containing the historical information about stock prices in the last year

  130000000,FCA,1000

  130000000,GOOG,10040
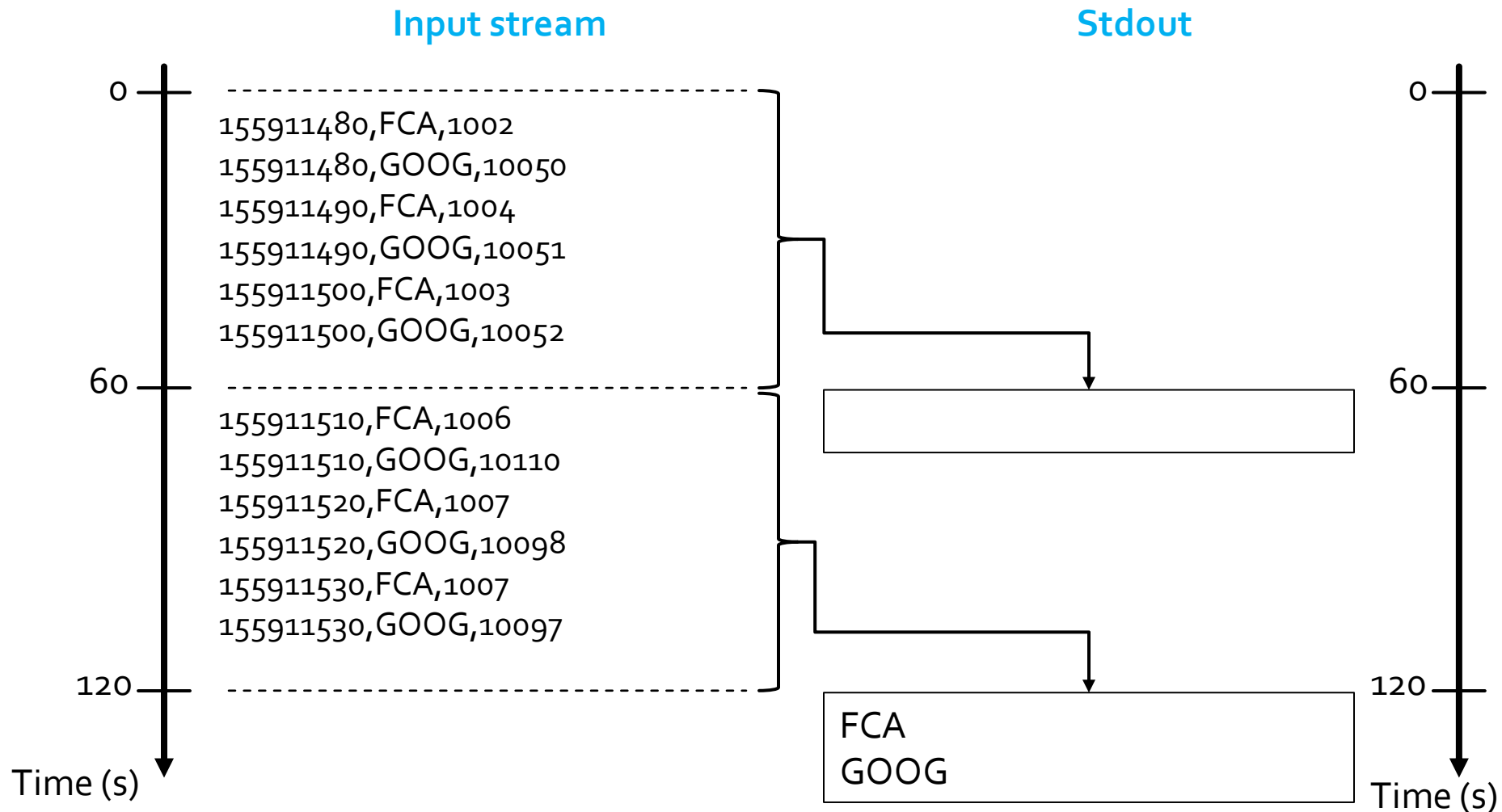
  130000060,FCA,1004

  130000060,GOOG,10080

  130000120,FCA,1001

  130000120,GOOG,10100

Exercise #64- Example

Input stream

Stdout

0

155911480,FCA,1002
155911480,GOOG,10050
155911490,FCA,1004
155911490,GOOG,10051
155911500,FCA,1003
155911500,GOOG,10052

60

155911510,FCA,1006
155911510,GOOG,10110
155911520,FCA,1007
155911520,GOOG,10098
155911530,FCA,1007
155911530,GOOG,10097

120

Time (s)

0

60

FCA
GOOG

120

Time (s)

Exercise #65

- Anomalous stock price identification in real-time
- Input:
  - A textual file containing the historical information about stock prices in the last year
    - Each input record has the format
      - Timestamp,StockID,Price
  - A real time stream of stock prices
    - Each input record has the format
      - Timestamp,StockID,Price

Exercise #65

- Output:
  - Every 30 seconds, by considering only the data received in the last 1 minute, print on the standard output the StockIDs of the stocks that satisfy one of the following conditions
    - price of the stock (received on the real-time input data stream) < historical minimum price of that stock (based only on the historical file)
    - price of the stock (received on the real-time input data stream) > historical maximum price of that stock (based only on the historical file)
  - If a stock satisfies the conditions multiple times in the same batch, return the stockId only one time for each batch

Exercise #65- Example

- Textual file containing the historical information about stock prices in the last year
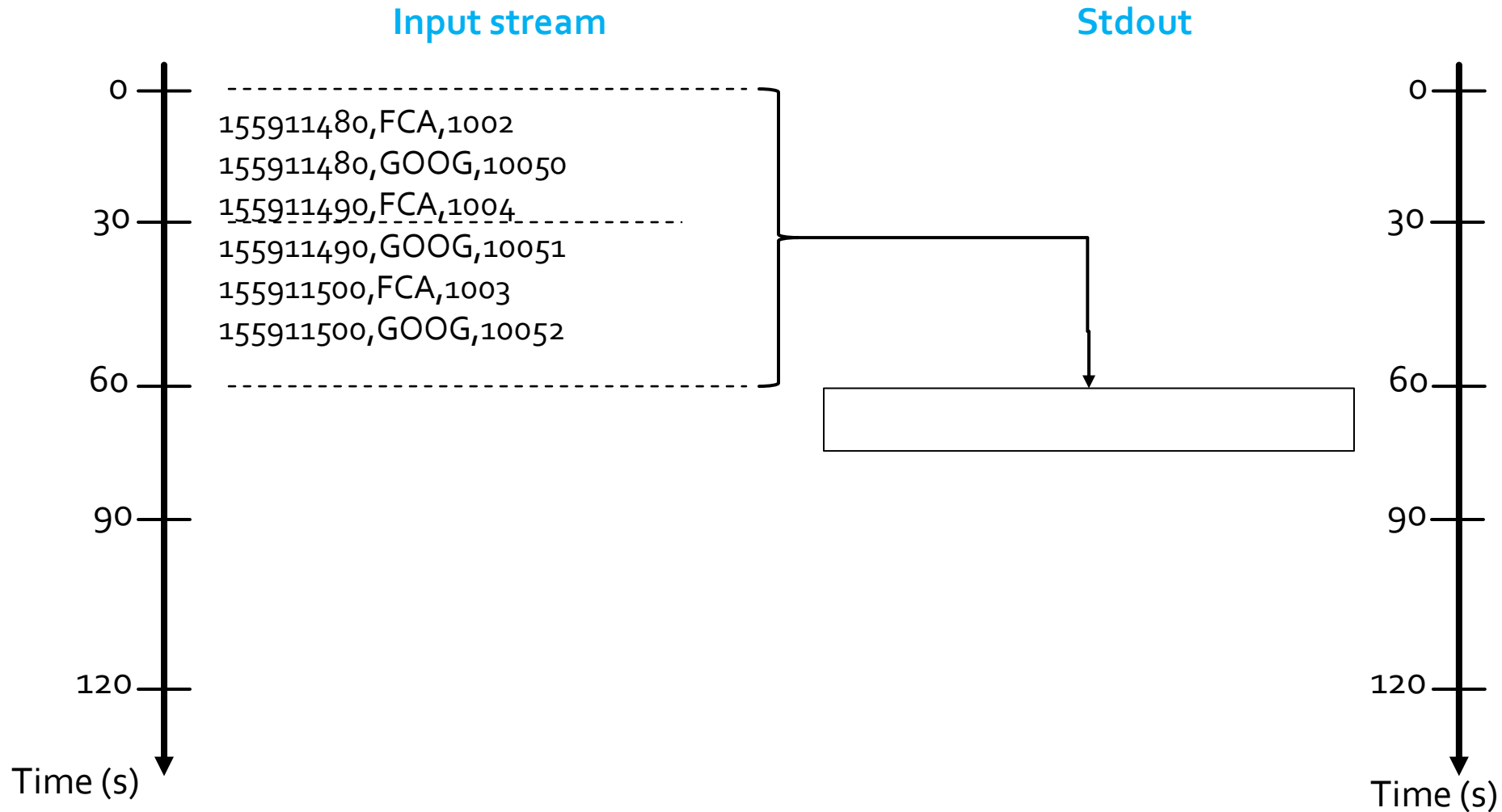
130000000,FCA,1000

130000000,GOOG,10040

130000060,FCA,1004

130000060,GOOG,10080

130000120,FCA,1001
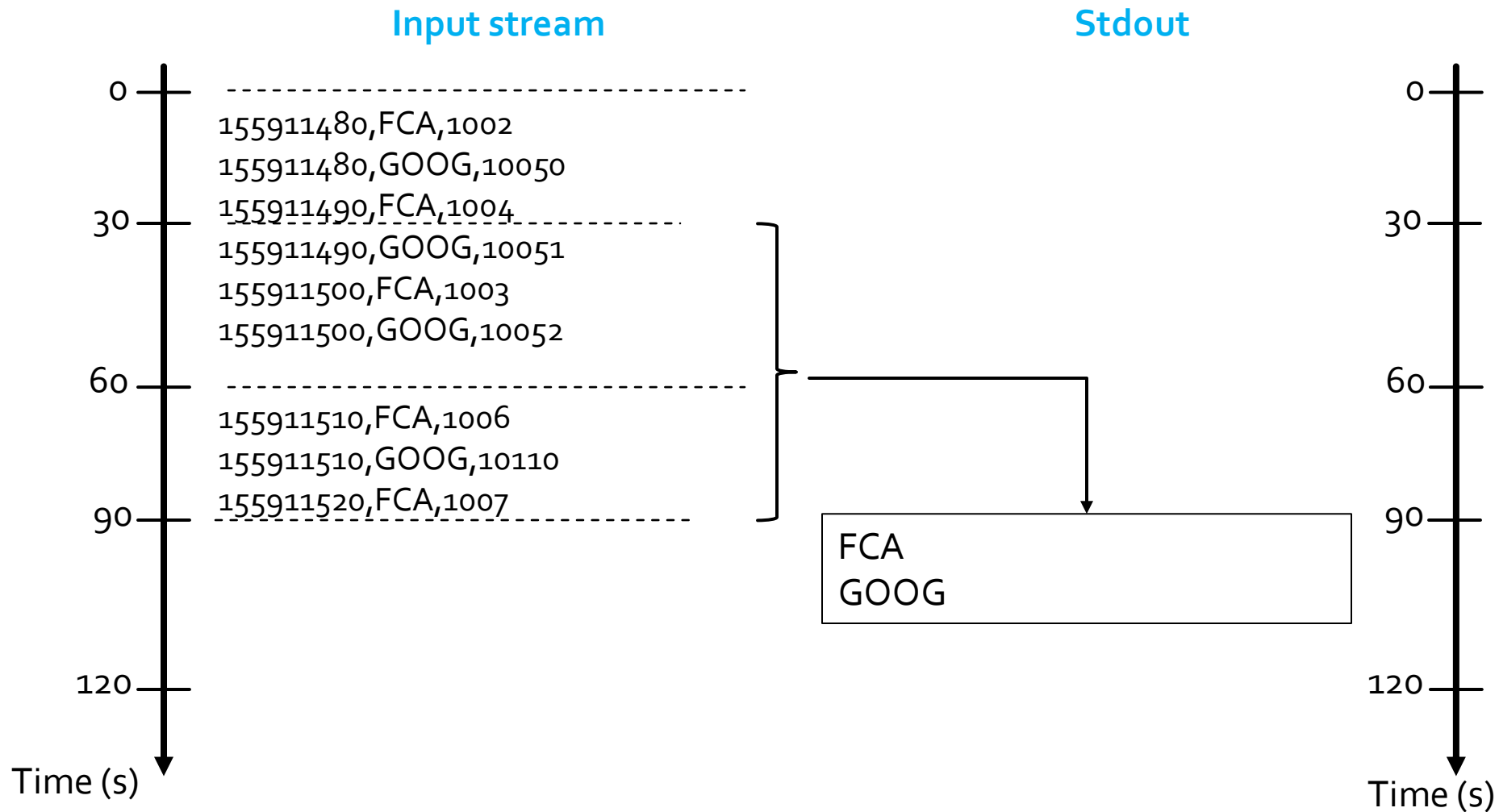
130000120,GOOG,10100

Exercise #65- Example

Input stream

1559114180,FCA,1002
1559114180,GOOG,10050
1559114190,FCA,1004
1559114190,GOOG,10051
1559114200,FCA,1003
1559114200,GOOG,10052

Stdout

Time (s)

Time (s)

Exercise #65- Example

Exercise #65- Example

Input stream

Stdout

0

155911480,FCA,1002
155911480,GOOG,10050
155911490,FCA,1004
30
155911490,GOOG,10051
155911500,FCA,1003
155911500,GOOG,10052

60

155911510,FCA,1006
155911510,GOOG,10110
155911520,FCA,1007
90
155911520,GOOG,10098
155911530,FCA,1007
155911530,GOOG,10097

120

Time (s)

0

30

60

90

120

FCA
GOOG

Time (s)