Section 2

Exercise 23.2.1

P_0 = (0.25, 0.25, 0.25, 0.25,)
P_1 = (0.125, 0.25, 0.25, 0.375)
P_2 = (0.125, 0.25, 0.3125, 0.3125)
P_3 = (0.1562, 0.2188, 0.2812, 0.3438)
P_4 = (0.1406, 0.25, 0.2812, 0.3281)
P_5 = (0.1406, 0.2344, 0.2891, 0.3359)
P_6 = (0.1445, 0.2383, 0.2852, 0.3320)
P_7 = (0.1426, 0.2383, 0.2852, 0.3340)
…
P_{11} = (0.1429, 0.2381, 0.2857, 0.3333)
P_{12} = (0.1429, 0.2381, 0.2857, 0.3333)

P = (3/21, 5/21, 6/21, 7/21)

Exercise 23.2.2

a)
P_0 = (0.25, 0.25, 0.25, 0.25,)
P_1 = (0.1375, 0.25, 0.25, 0.3625)
P_2 = (0.1375, 0.25, 0.3006, 0.3119)
P_3 = (0.1603, 0.2272, 0.2778, 0.3347)
P_4 = (0.15, 0.2477, 0.2778, 0.3244)
P_5 = (0.15, 0.2385, 0.2825, 0.3290)
...
P_9 = (0.1513, 0.2406, 0.2806, 0.3276)
P_{10} = (0.1513, 0.2406, 0.2806, 0.3276)

P = (0.1513, 0.2406, 0.2806, 0.3276)

b)
P_0 = (0.25, 0.25, 0.25, 0.25)
P_1 = (0.15, 0.25, 0.25, 0.35,
P_2 = (0.15, 0.25, 0.29, 0.31)
P_3 = (0.166, 0.234, 0.274, 0.3260)
P_4 = (0.1596, 0.2468 0.274, 0.3196)
P_5 = (0.1596, 0.2417, 0.2766, 0.3222)
P_6 = (0.1606, 0.2427, 0.2755, 0.3211)
P_7 = (0.1602, 0.2427, 0.2755, 0.3215)
P_8 = (0.1602, 0.2427, 0.2757, 0.3214)
P_9 = (0.1603, 0.2426, 0.2756, 0.3214)
P_{10} = (0.1603, 0.2426, 0.2756, 0.3214)

P = (0.1603, 0.2426, 0.2756, 0.3214)

Exercise 23.2.3

i a)
P_0 = (0.25, 0.25, 0.25, 0.25)
P_1 = (0.025, 0.025, 0.1375, 0.1375)
P_2 = (0.025, 0.025, 0.0363, 0.0363)
P_3 = (0.025, 0.025, 0.0363, 0.0363)

P = (0.025, 0.025, 0.0363, 0.0363)

i b)
P_0 = (0.25, 0.25, 0.25, 0.25)
P_1 = (0.05, 0.05, 0.15, 0.15)
P_2 = (0.05, 0.05, 0.07, 0.07)
P_3 = (0.05, 0.05, 0.07, 0.07)

P = (0.05, 0.05, 0.07, 0.07)

ii a)
P_0 = (0.25, 0.25, 0.25, 0.25)
P_1 = (0.1375, 0.25, 0.025, 0.3625)
P_2 = (0.0363, 0.3513, 0.0250, 0.2613)
P_3 = (0.0363, 0.2601, 0.0250, 0.3524)
P_4 = (0.0363, 0.3421, 0.0250, 0.2704)
P_5 = (0.0363, 0.2683, 0.0250, 0.3442)
P_6 = (0.0363, 0.3348, 0.0250, 0.2777)
P_7 = (0.0363, 0.2750, 0.0250, 0.3375)
...
P_{69} = (0.0363, 0.3032, 0.0250, 0.3093)
P_{70} = (0.0363, 0.3033, 0.0250, 0.3092)
P_{71} = (0.0363, 0.3033, 0.0250, 0.3092)

P = (0.0363, 0.3033, 0.0250, 0.3092)

ii b)
P_0 = (0.25, 0.25, 0.25, 0.25)
P_1 = (0.15, 0.25, 0.05, 0.35)
P_2 = (0.07, 0.33, 0.05, 0.27)
P_3 = (0.07, 0.266, 0.05, 0.3340)
P_4 = (0.07, 0.3172, 0.05, 0.2828)
P_5 = (0.07, 0.2762, 0.05, 0.3238)
P_6 = (0.07, 0.3090, 0.05, 0.2910)
P_7 = (0.07, 0.2828, 0.05, 0.3172)
...

P_{40} = (0.07, 0.2945, 0.05, 0.3055)
P_{41} = (0.07, 0.2944, 0.05, 0.3056)
P_{42} = (0.07, 0.2944, 0.05, 0.3056)

P = (0.07, 0.2944, 0.05, 0.3056)

Exercise 23.2.4

Map function:
The map function takes in, as parameters, the page rank estimate of a certain page along with its corresponding column in the transition matrix; the column maybe partitioned to accommodate the number of processors. The map function will output key-value pairs where the key of each pair corresponds to a page from the provided column and the value is the page rank for that page, which is the product of its value in the provided column and the page rank estimate.

Reduce function:
The reduce function simply sums up key-value pairs that have a common key.

Section 3

Exercise 23.3.1

Compute page rank for fig 23.5
a)
P_0 = (0.25, 0.25, 0.25, 0.25)
P_1 = (0.3, 0.2, 0.2, 0.3)
P_2 = (0.28, 0.24, 0.2, 0.28)
P_3 = (0.28, 0.224, 0.208, 0.288)
P_4 = (0.2832, 0.2272, 0.2048, 0.2848)
P_5 = (0.2819 0.2272 0.2048 0.2861)
...
P_{10} = (0.2821, 0.2271, 0.2051, 0.2857)
P_{11} = (0.2821, 0.2271, 0.2051, 0.2857)

P = (0.2821, 0.2271, 0.2051, 0.2857)

b)
P_0 = (0.25, 0.25, 0.25, 0.25)
P_1 = (0.2, 0.3, 0.2, 0.3)
P_2 = (0.18, 0.30, 0.24, 0.28)
P_3 = (0.196, 0.284, 0.232, 0.288)
P_4 = (0.1928, 0.2936, 0.2288, 0.2848)
P_5 = (0.1915, 0.2910, 0.2314, 0.2861)
P_6 = (0.1925, 0.2910, 0.2308, 0.2856)
P_7 = (0.1923, 0.2912, 0.2306, 0.2858)
P_8 = (0.1923, 0.2912, 0.2308, 0.2857)

$P_9 = (0.1923, 0.2912, 0.2308, 0.2857)$

$P = (0.1923, 0.2912, 0.2308, 0.2857)$

Exercise 23.3.2
a)
$P_0 = (0.25, 0.25, 0.25, 0.25)$
$P_1 = (0.2, 0, 0.1, 0.1)$
$P_2 = (0.2, 0, 0, 0)$
$P_3 = (0.2, 0, 0, 0)$

$P = (0.2, 0, 0, 0)$

b)
$P_0 = (0.25, 0.25, 0.25, 0.25)$
$P_1 = (0.1, 0.1, 0.1, 0.1)$
$P_2 = (0.1, 0.1, 0.04, 0.04)$
$P_3 = (0.1, 0.1, 0.04, 0.04)$

$P = (0.1, 0.1, 0.04, 0.04)$

Exercise 23.3.3
a)
$P_0 = (0.25, 0.25, 0.25, 0.25)$
$P_1 = (0.3, 0.2, 0, 0.3)$
$P_2 = (0.2, 0.24, 0, 0.16)$
$P_3 = (0.2, 0.128, 0, 0.1920)$
$P_4 = (0.2, 0.1536, 0, 0.1024)$
...
$P = (0.2, 0, 0, 0)$

b)
$P_0 = (0.25, 0.25, 0.25, 0.25)$
$P_1 = (0.2, 0.3, 0, 0.3)$
$P_2 = (0.1, 0.34, 0, 0.24)$
$P_3 = (0.1, 0.292, 0, 0.272)$
$P_4 = (0.1, 0.3176, 0, 0.2336)$
$P_5 = (0.1, 0.2869, 0, 0.2541)$
...
$P_{36} = (0.1, 0.2778, 0, 0.2222)$
$P_{37} = (0.1, 0.2778, 0, 0.2222)$

$P = (0.1, 0.2778, 0, 0.2222)$

Exercise 23.3.4

Proof by construction

$$\beta \begin{bmatrix} G_{11} \dots G_{1a} \dots G_{1b} \dots G_{1n} \\ \dots \\ G_{a1} \dots G_{aa} \dots G_{ab} \dots G_{an} \\ \dots \\ G_{b1} \dots G_{ba} \dots G_{bb} \dots G_{bn} \\ \dots \\ G_{n1} \dots G_{na} \dots G_{nb} \dots G_{nn} \end{bmatrix} \begin{bmatrix} R_1 \\ \dots \\ R_a \\ \dots \\ R_b \\ \dots \\ R_n \end{bmatrix} + T \begin{bmatrix} 0_1 \\ \dots \\ 1_a \\ \dots \\ 0_b \\ \dots \\ 0_n \end{bmatrix} = \begin{bmatrix} (G_{11}R_1 + \dots + G_{1a}R_a + \dots + G_{1b}R_b + \dots + G_{1n}R_n)\beta \\ \dots \\ (G_{a1}R_1 + \dots + G_{aa}R_a + \dots + G_{ab}R_b + \dots + G_{an}R_n)\beta + T \\ \dots \\ (G_{b1}R_1 + \dots + G_{ba}R_a + \dots + G_{bb}R_b + \dots + G_{bn}R_n)\beta \\ \dots \\ (G_{n1}R_1 + \dots + G_{na}R_a + \dots + G_{nb}R_b + \dots + G_{nn}R_n)\beta \end{bmatrix} \quad (A)$$

$$\beta \begin{bmatrix} G_{11} \dots G_{1a} \dots G_{1b} \dots G_{1n} \\ \dots \\ G_{a1} \dots G_{aa} \dots G_{ab} \dots G_{an} \\ \dots \\ G_{b1} \dots G_{ba} \dots G_{bb} \dots G_{bn} \\ \dots \\ G_{n1} \dots G_{na} \dots G_{nb} \dots G_{nn} \end{bmatrix} \begin{bmatrix} R_1 \\ \dots \\ R_a \\ \dots \\ R_b \\ \dots \\ R_n \end{bmatrix} + T \begin{bmatrix} 0_1 \\ \dots \\ 0_a \\ \dots \\ 1_b \\ \dots \\ 0_n \end{bmatrix} = \begin{bmatrix} (G_{11}R_1 + \dots + G_{1a}R_a + \dots + G_{1b}R_b + \dots + G_{1n}R_n)\beta \\ \dots \\ (G_{a1}R_1 + \dots + G_{aa}R_a + \dots + G_{ab}R_b + \dots + G_{an}R_n)\beta \\ \dots \\ (G_{b1}R_1 + \dots + G_{ba}R_a + \dots + G_{bb}R_b + \dots + G_{bn}R_n)\beta + T \\ \dots \\ (G_{n1}R_1 + \dots + G_{na}R_a + \dots + G_{nb}R_b + \dots + G_{nn}R_n)\beta \end{bmatrix} \quad (B)$$

$$\beta \begin{bmatrix} G_{11} \dots G_{1a} \dots G_{1b} \dots G_{1n} \\ \dots \\ G_{a1} \dots G_{aa} \dots G_{ab} \dots G_{an} \\ \dots \\ G_{b1} \dots G_{ba} \dots G_{bb} \dots G_{bn} \\ \dots \\ G_{n1} \dots G_{na} \dots G_{nb} \dots G_{nn} \end{bmatrix} \begin{bmatrix} R_1 \\ \dots \\ R_a \\ \dots \\ R_b \\ \dots \\ R_n \end{bmatrix} + T \begin{bmatrix} 0_1 \\ \dots \\ .5_a \\ \dots \\ .5_b \\ \dots \\ 0_n \end{bmatrix} = \begin{bmatrix} (G_{11}R_1 + \dots + G_{1a}R_a + \dots + G_{1b}R_b + \dots + G_{1n}R_n)\beta \\ \dots \\ (G_{a1}R_1 + \dots + G_{aa}R_a + \dots + G_{ab}R_b + \dots + G_{an}R_n)\beta + .5T \\ \dots \\ (G_{b1}R_1 + \dots + G_{ba}R_a + \dots + G_{bb}R_b + \dots + G_{bn}R_n)\beta + .5T \\ \dots \\ (G_{n1}R_1 + \dots + G_{na}R_a + \dots + G_{nb}R_b + \dots + G_{nn}R_n)\beta \end{bmatrix} \quad (C)$$

$$[(A) + (B)]/2 = \begin{bmatrix} (G_{11}R_1 + \dots + G_{1a}R_a + \dots + G_{1b}R_b + \dots + G_{1n}R_n)\beta \\ \dots \\ (G_{a1}R_1 + \dots + G_{aa}R_a + \dots + G_{ab}R_b + \dots + G_{an}R_n)\beta + .5T \\ \dots \\ (G_{b1}R_1 + \dots + G_{ba}R_a + \dots + G_{bb}R_b + \dots + G_{bn}R_n)\beta + .5T \\ \dots \\ (G_{n1}R_1 + \dots + G_{na}R_a + \dots + G_{nb}R_b + \dots + G_{nn}R_n)\beta \end{bmatrix}$$

(A) – when *a* is the only node in the teleport set
(B) – when *b* is the only node in the teleport set
(C) – when the teleport set is $\{a, b\}$, which equals to the average between (A) and (B)

Exercise 23.3.5

We can multiply each result by the number of elements in its teleport set, then sum the modified results together and divide by the number of elements in both teleport sets.

Section 4

Exercise 23.4.1

a)

```
SELECT *
FROM
        Sensors [Rows 1000]
WHERE
         time IN (SELECT MIN(time) FROM Sensors[Rows 1000]);
```

b)

```
SELECT s.sensID
FROM
        Sensors [Range 1 Minute] s
WHERE
        EXISTS (SELECT * FROM Sensors [Range 1 Minute] WHERE
        sensID = s.sensID AND time <> s.time);
```

c)

```
SELECT a.sensID
FROM
        (SELECT sensID, count(sensID) as cnt FROM sensors [Range 1 minute] GROUP BY
        sensID) a,
        (SELECT sensID, count(sensID) as cnt FROM sensors [Range 2  minute] GROUP BY
        sensID) b
WHERE
        a.sensID = b.sensID AND (a.cnt * 2) > (b.cnt)
```

Exercise 23.4.2

Readings	(80, 0)	(70, 50)	(60, 70)	(65, 100)
Istream(R)	0	50	70	100
Dstream(R)	60	110	100	160

Exercise 23.4.3

a)

```
SELECT item FROM
        (SELECT item, COUNT(basket) AS cnt FROM Basket[Range 1 hour] GROUP BY item)
WHERE
        cnt * 100 >= (SELECT COUNT (DISTINCT basket) FROM Basket[Range 1 hour]);
```

b)

```
SELECT P1.item1, P1.item2
FROM
        (SELECT a.item AS item1, b.item AS item2, COUNT(*) AS cnt FROM Basket[Range
        30 minutes] a, Basket[Range 30 minutes] b WHERE a.basket = b.basket and item1 <>
        item2 GROUP BY item1, item2) P1,
        (SELECT a.item AS item1, b.item AS item2, COUNT(*) AS cnt FROM Basket[Range 1
        hour] a, Basket[Range 1 hour] b WHERE a.basket = b.basket and item1 <> item2
        GROUP BY item1, item2) P2
WHERE
        P1.cnt >= P2.cnt * (2/3);
```

c)

```
SELECT item1, item2
FROM
        (SELECT a.item AS item1, b.item AS item2, COUNT(*) AS cnt FROM Basket[Range 1
        hour] a, Basket[Range 1 hour] b WHERE a.basket = b.basket AND item1 <> item2
        GROUP BY item1, item2)
WHERE
        cnt = (SELECT MAX(cnt2) FROM (SELECT COUNT(*) AS cnt2 FROM Basket[Range
        1 hour] c, Basket[Range 1 hour] d WHERE c.basket = d.basket AND c.item <> d.item
        GROUP BY c.item, d.item);
```

Section 5

Exercise 23.5.1

16, 8, 8, 4, 4, 2, 1, 1

1: 16, 8, 8, 4, 4, 2, 1, 1, 1 => 16, 8, 8, 4, 4, 2, 2, 1
2: 16, 8, 8, 4, 4, 2, 2, 1, 1
3: 16, 8, 8, 4, 4, 2, 2, 1, 1, 1 => 16, 8, 8, 4, 4, 2, 2, 2, 1 => 16, 8, 8, 4, 4, 4, 2, 1=>
16, 8, 8, 8, 4, 2, 1 => 16, 16, 8, 4, 2, 1
4: 16, 16, 8, 4, 2, 1, 1
5: 16, 16, 8, 4, 2, 1, 1, 1 => 16, 16, 8, 4, 2, 2, 1
6: 16, 16, 8, 4, 2, 2, 1, 1
7: 16, 16, 8, 4, 2, 2, 1, 1, 1 => 16, 16, 8, 4, 2, 2, 2, 1 => 16, 16, 8, 4, 4, 2, 1
8: 16, 16, 8, 4, 4, 2, 1, 1
9: 16, 16, 8, 4, 4, 2, 1, 1, 1 => 16, 16, 8, 4, 4, 2, 2, 1
10: 16, 16, 8, 4, 4, 2, 2, 1, 1

Exercise 23.5.2

k	10	15	20
buckets	2, 1, 1	4, 2, 1, 1	4, 4, 2, 1, 1
# of 1's (estimate)	4	6	10

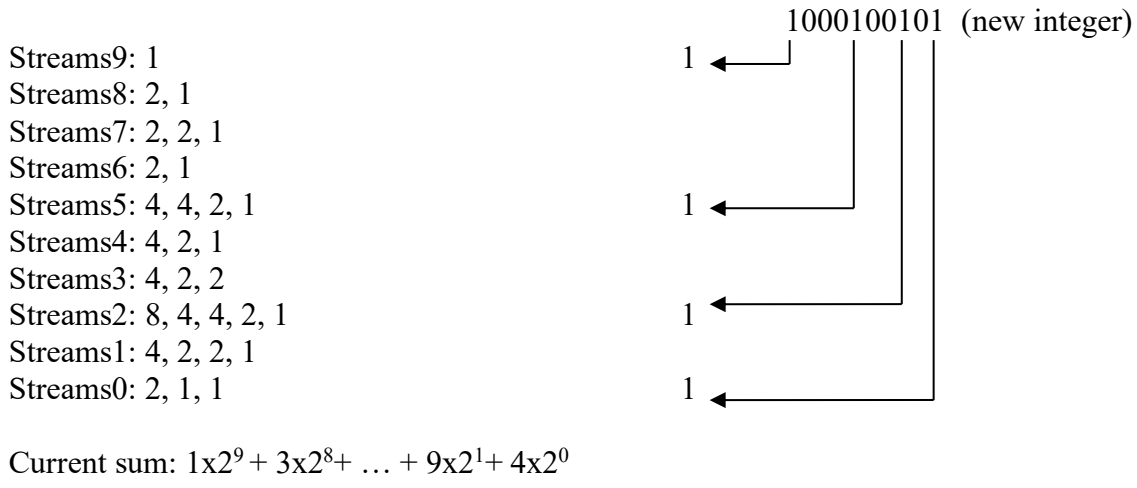
For this particular example, the estimates match their exact values.

Exercise 23.5.3

Create 10 streams, one for each bits of the 10-bit integer. When a new integer arrives, send all the bits that are 1's to their corresponding streams. Each stream employs the bit-counting method of section 23.5.2. The sum of integers from the last k time units is calculated as

$$S_0(k) \times 2^0 + S_1(k) \times 2^1 + S_2(k) \times 2^2 + \ldots + S_7(k) \times 2^7 + S_8(k) \times 2^8 + S_9(k) \times 2^9,$$

where $S_i(k)$ is the estimated number of bits from ith stream, which corresponds to the ith bit of the integer.



Current sum: $1x2^9 + 3x2^8 + \ldots + 9x2^1 + 4x2^0$

Exercise 23.5.4

a)

Whenever there are p+2 buckets of any size, combine the two least recent buckets of that size to form a new one; the new bucket will take on the timestamp of the more recent bucket. Repeat this process if the newly formed bucket causes there to be p+2 buckets of that size.

b)

The error of the algorithm presented in section 23.5.2 is at most B/2 where B is the size of the last bucket. If there are n buckets, then the true count is at least

$$1 + B_{n-1} + B_{n-2} + B_{n-3} + \ldots + B_2 + B_0$$

since the last bucket contributes at least one 1, and rest contributes their exact size. Therefore, the maximum error is

$$(B_n/2)/(1 + B_{n-1} + B_{n-2} + B_{n-3} + \ldots + B_2 + B_0).$$

Since bucket size increases by powers of 2, the sum last p bucket of the same size as B_n cannot be larger than the sum of the remaining buckets plus p (at least one 1 from each bucket contributes to the total). Thus the maximum error is reduced to a factor of 1/p since the approximation error is only from the last bucket. The new approximation error is

$$1/p[(B_n/2)/(1 + B_{n-1} + B_{n-2} + B_{n-3} + \ldots + B_2 + B_0)],$$

which reduces to

$$1/p(1/2) = 1/2p$$

when the sum of the last p bucket is equal to the sum of the remaining buckets plus p.

Exercise 23.5.5

a)

v	**$h_1(v)$**	**r**	**R**
24	24(11000)	3	3
45	45(101101)	0	3
102	102(1100110)	1	3
24	24(11000)	3	3
78	78(1001110)	1	3
222	222(11011110)	1	3
45	45(101101)	0	3
24	24(11000)	3	3
670	158(10011110)	1	3
78	78(1001110)	1	3
999	487(111100111)	0	3
576	64(1000000)	6	6
222	222(11011110)	1	6
24	24(11000)	3	6

$2^6 = 64$

b)

v	**$h_2(v)$**	**r**	**R**
24	183(10110111)	0	0
45	204(11001100)	2	2
102	261(100000101)	0	2
24	183(10110111)	0	2
78	237(11101101)	0	2
222	381(101111101)	0	2
45	204(11001100)	2	2
24	183(10110111)	0	2
670	317(100111101)	0	2
78	134(10000110)	1	2
999	237(11101101)	0	2
576	223(11011111)	0	2
222	381(101111101)	0	2
24	183(10110111)	0	2

$2^2 = 4$

c)

v	**h_3(v)**	**r**	**R**
24	365(101101101)	0	0
45	386(110000010):1	1	1
102	443(110111011):0	0	1
24	365(101101101):0	0	1
78	419(110100011):0	0	1
222	51(110011):0	0	1
45	386(110000010):1	1	1
24	365(101101101):0	0	1
670	499(111110011):0	0	1
78	419(110100011):0	0	1
999	316(100111100):2	2	2
576	405(110010101):0	0	2
222	51(110011):0	0	2
24	365(101101101):0	0	2

$2^2 = 4$

Exercise 23.5.6

a)

No, this rule does not always compress the data in the window; it will not achieve any compression when all the temperature readings are either the same or always decreasing.

b)

Since the temperature readings are real numbers chosen uniformly and at random, the chance of a chosen reading being smaller or larger than a previous reading is 50%. There are infinite numbers of real numbers in any fixed range. Each new temperature reading will have a 50% chance of reducing the number of tuples by half. Thus, on average, $\log_2 N$ tuples will be retained.