Q2

Queue Q									
1	2	3	4	5	6	7	8	9	10

Q
x = 2
y = 4
z = 6

x	y	z	4	5	6	7	8	9	10
2	4	6							

2. Append (8)

x	y	z	4	5	6	7	8	9	10
2	4	6	8						

3. Append (x-y)

x	y	z	4	5	6	7	8	9	10
2	4	6	8						

2 - 4 = -2

x	y	z	4	5	6	7	8	9	10
2	4	6	8						

4. Append (z)

z value = 6

x	y	z		5	6	7	8	9	10
2	4	6	8						

5. y + = g.serve()
Y + serve()
Y = 4 + 8 = 12

x	y	z		5	6	7	8	9	10
2	12	6	8						

6. g.append (x+z)

x + z = 8

x	y	z		5	6	7	8	9	10
2	12	6	8						

7. z = g.serve () - 4

8 + 2 = 10

Then minus 4

10 - 4 = 6

8. Store value z = 6

x	y	z		5	6	7	8	9	10
2	12	6	8						

9. g.append (z)

x	y	z		z	6	7	8	9	10
2	12	6	8	6					

10. g. append (3)

x	y	z		z	6	7	8	9	10
2	12	6	8	6	3				

11. x - = q.serve() x = x-z serve the 1st element of the queue (z) and subtract it from x

z= 6

2 - 6 = -4

x	y	z		z	6	7	8	9	10
2	12	6	8	6	3				

Q3
Queue Q

M = 6
N = 8
Holds 10 Integer

Front	1	2	3	4	5	6	7	8	9	10	back

1. Enqueue (m)

Front	1	2	3	4	5	6	7	8	9	10	back
	6										

2. Enqueue (n)

Front	1	2	3	4	5	6	7	8	9	10	back
	6	8									

3. n+=q.dequeue()

n = n+dequeue

Front	1	2	3	4	5	6	7	8	9	10	back
	8										

4. q.Enqueue (n)

Front	1	2	3	4	5	6	7	8	9	10	back
	8	8									

5. q.Enqueue(m+n);

m = 6
n = 8

Front	1	2	3	4	5	6	7	8	9	10	back
	8	14									

6. if (q.dequeue() >= m)
m = q.dequeue()

8 >= 6   dequeue 8

Front	1	2	3	4	5	6	7	8	9	10	back
	14										

7. q.Enequeue(m)

Front	1	2	3	4	5	6	7	8	9	10	back
	14	6									

8. q.enqueue (12)

Front	1	2	3	4	5	6	7	8	9	10	back
	14	6	12								

```
9. if (q.Denqueue() < m)
m= q.dequeue()
```

14 < 6 False

Dequeue

Front	1	2	3	4	5	6	7	8	9	10	back
	6	12									

SUBMISSIVE QUESTION

Queue que = Holds 10 intergers

1. ADTqueue que;
initialize Queue for que

Front	1	2	3	4	5	6	7	8	9	10	back

2. int x = 3
3. int y = 6

4. que.append (8)

	1	2	3	4	5	6	7	8	9	10	
	8										

5. que.append (x-y)

3 - 6 = -3

	1	2	3	4	5	6	7	8	9	10	
	8	-3									

6. que.append(y);

	1	2	3	4	5	6	7	8	9	10	
	8	-3	6								

7.y %= que.serve();

6 % 8 = 6

	1	2	3	4	5	6	7	8	9	10	
	-3	6	6								

8. que.append(x);

	1	2	3	4	5	6	7	8	9	10	
	-3	6	3								

9. y = que.serve ( ) * 2;

-3 * 2 = -6

	1	2	3	4	5	6	7	8	9	10	
	6	3	-6								

10. que.append(y);

	1	2	3	4	5	6	7	8	9	10	
	6	3	-6								

11.que.append (3);

	1	2	3	4	5	6	7	8	9	10	
	6	3	-6	3							

12.x *= que.serve( );

3 * 6 = 18

	1	2	3	4	5	6	7	8	9	10	
	3	-6	3	18							

13. que.append(x);

	1	2	3	4	5	6	7	8	9	10	
	3	-6	3	18							

14. que.append(pow(y,2));

y = -6
y = -6 ^ 2 = 36

	1	2	3	4	5	6	7	8	9	10	
	3	-6	3	18	36						

15. cout<<"Elements in the queue are :\n";

	1	2	3	4	5	6	7	8	9	10	
	3	-6	3	18	36						

```
16. while(!que.empty( ))
{
  cout<<que.serve( )*5<<endl;
}
```

15 -30 15 90 180

	1	2	3	4	5	6	7	8	9	10	
	15	-30	15	90	180						

Output =	15	-30	15	90	180