1.

$R_0 = Q_0 + C_0$

$Q_0 = B_0 + \sum_{\forall j, P_j < P_0} \left\lceil \frac{Q_0 + 0.1}{T_j} \right\rceil C_j$

Since P_0 is the smallest, $\sum_{\forall j, P_j < P_i} \left\lceil \frac{Q_0 + 0.1}{T_j} \right\rceil C_j = 0$

$\Rightarrow R_0 = B_0 + C_0 = 30 + 10 = 40$

Iteration	LHS (Q_0)	B_0	RHS	Stop?
1	30	30	30	Yes

2.

Iteration	LHS (Q_1)	B_1	j	$Q_1 + \tau$	T_j	$\left\lceil \frac{Q_1 + \tau}{T_j} \right\rceil$	C_j	RHS	Stop?
1	30	30	0	30.1	50	1	10	40	no
2	40	30	0	40.1	50	1	10	40	Yes

Based on the above table,

$R_1 = Q_1 + C_1 = 40 + 30 = 70$

3.

Iteration	LHS (Q_2)	B_2	j	$Q_2 + \tau$	T_j	$\left\lceil \frac{Q_2 + \tau}{T_j} \right\rceil$	C_j	RHS	Stop?
1	20	20	0 1	20.1	50 200	1 1	10 30	60	no
2	60	20	0 1	60.1	50 200	2 1	10 30	70	no
3	70	20	0 1	70.1	50 200	2 1	10 30	70	Yes

Based on the above table,

$R_2 = Q_2 + C_2 = 70 + 20 = 90.$

```cpp
/*
1.44
2.04
2.56
3.16
3.68
4.28
5.2
8.4
9
9.68
10.2
19.36
19.8
20.32
29.4
29.76
30.28
*/

#include <bits/stdc++.h>
using namespace std;

int main()
{
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);

    double n, tau;
    cin >> n >> tau; // n = number of messages, tau = time doubleerval
    vector<double> priority(n), transmission_time(n), period(n), response_time(n, 0);

    for (int i = 0; i < n; i++)
        cin >> priority[i] >> transmission_time[i] >> period[i];

    for (int i = 0; i < n; i++)
    {
        double LHS = -1, RHS = transmission_time[i];
        while (LHS != RHS)
        {
            LHS = RHS;
            double blocking_time = 0, sigma = 0;
            for (int j = 0; j < n; j++)
            {
                if (priority[j] >= priority[i])
                    blocking_time = max(blocking_time, transmission_time[j]);
            }
            for (int j = 0; j < n; j++)
            {
                if (priority[j] < priority[i])
                    sigma += ceil((LHS + tau) / period[j]) * transmission_time[j];
            }
            RHS = blocking_time + sigma;
        }
        response_time[i] = RHS + transmission_time[i];
    }

    for (int i = 0; i < n; i++)
        cout << response_time[i] << "\n";

    return 0;
}
```

1.

Iteration	LHS (R_0)	C_0	RHS	Stop?
1	10	10	10	Yes

$R_0 = C_0 = 10$.

2.

Iteration	LHS (R_1)	C_1	j	R_1	T_j	$\left\lceil \frac{R_1}{T_j} \right\rceil$	C_j	RHS	Stop?
1	30	30	0	30	50	1	10	40	no
2	40	30	0	40	50	1	10	40	Yes

$R_1 = 40$.

3.

Iteration	LHS (R_2)	C_2	j	R_2	T_j	$\left\lceil \frac{R_2}{T_j} \right\rceil$	C_j	RHS	Stop?
1	20	20	0 1	20	50 200	1 1	10 30	60	no
2	60	20	0 1	60	50 200	2 1	10 30	70	no
3	70	20	0 1	70	50 200	2 1	10 30	70	Yes

$R_2 = 70$.

4.

The algorithm for μ overestimates the worst case response time while the algorithm for P doesn't.

1.

(2, 5, 1, 2) → (4, 10, 1, 2, 6, 7)

2.

(4, 10, 0, 3, 5, 6) → (4, 10, 0, 3, 5, 6, 10, 13, 15, 16) a

3.

(4, 10, 1, 2, 6, 7) → (4, 10, 1, 2, 6, 7, 11, 12, 16, 17) s

4.

k	$\max_{1\le j\le n}(s_{j+k} - s_j)$	$=$	$\min_{1\le i\le m}(a_{i+k-1} - a_i)$	$=$	(Column-3) − (Column-5)
1	$\max_{1\le j\le 4}(s_{j+1} - s_j)$	4	$\min_{1\le i\le 4}(a_i - a_i)$	0	4
2	$\max_{1\le j\le 4}(s_{j+2} - s_j)$	5	$\min_{1\le i\le 4}(a_{i+1} - a_i)$	1	4
3	$\max_{1\le j\le 4}(s_{j+3} - s_j)$	9	$\min_{1\le i\le 4}(a_{i+2} - a_i)$	3	6
4	$\max_{1\le j\le 4}(s_{j+4} - s_j)$	10	$\min_{1\le i\le 4}(a_{i+3} - a_i)$	6	4

5.

worst case response time = 1 + 6 = 7

1、

α	β	γ	LHS	$\alpha+\beta-\gamma\leq 1$	$\alpha-\beta+\gamma\leq 1$	$-\alpha+\beta+\gamma\leq 1$	RHS	LHS=RHS?
0	0	0	T	T	T	T	T	T
0	0	1	T	T	T	T	T	T
0	1	0	T	T	T	T	T	T
0	1	1	F	T	T	F	F	T
1	0	0	T	T	T	T	T	T
1	0	1	F	T	F	T	F	T
1	1	0	F	F	T	T	F	T
1	1	1	T	T	T	T	T	T

2、

α	β	γ	LHS	$\alpha+\beta-1\leq\gamma$	$\gamma\leq\alpha$	$\gamma\leq\beta$	RHS	LHS=RHS?
0	0	0	T	T	T	T	T	T
0	0	1	F	T	F	F	F	T
0	1	0	T	T	T	T	T	T
0	1	1	F	T	F	T	F	T
1	0	0	T	T	T	T	T	T
1	0	1	F	T	T	F	F	T
1	1	0	F	F	T	T	F	T
1	1	1	T	T	T	T	T	T

3、

β	LHS	$0\leq y\leq x$	$x-M(1-\beta)\leq y$	$y\leq M\beta$	RHS
0	$0=y$	$0\leq y\leq x$	$x-M\leq y$	$y\leq 0$	$x-M\leq y=0\leq x$
1	$x=y$	$0\leq y\leq x$	$x\leq y$	$y\leq M$	$0\leq y=x\leq M$

from the table, we can know

$x-M\leq 0\leq x$

$0\leq y=x\leq M$ $\Rightarrow$ $M\geq x\geq y\geq 0$

since $x\leq 2022$, we can select M as 2022.

1.

Yes.

To transmit μ_0, we need $8+44+3$ bits.

" μ_1, " $16+44+3$ " .

" μ_0', " $16+44+3$ " .

We can save a lot bits if we use the new design.

2.

No.

The senders are not the same.

3.

Yes.

μ_0' : $(16+44+3)/50 = 1.26$ bpms

μ_2 : $(16+44+3)/50 = 1.26$ bpms

μ_3 : $(16+44+3)/100 = 0.63$ bpms

Since the sender of μ_0' and μ_3 are the same, we can concat μ_3 behind μ_0'.

μ_0'' : $(32+44+3)/50 = 1.58 < 1.26+0.63$.