Question **1**

Correct

You are given a two-dimensional 3*3 array starting from A [0][0]. You should add the alternate elements of the array and print its sum. It should print two different numbers the first being sum of A 0 0, A 0 2, A 1 1, A 2 0, A 2 2 and A 0 1, A 1 0, A 1 2, A 2 1.

Input Format

First and only line contains the value of array separated by single space.

A00	A01	A02
4	6	9
A10	**A11**	**A12**
2	5	8
A20	**A21**	**A22**
1	3	7

Output Format

First line should print sum of A 0 0, A 0 2, A 1 1, A 2 0, A 2 2

Second line should print sum of A 0 1, A 1 0, A 1 2, A 2 1

SAMPLE INPUT

1 2 3 4 5 6 7 8 9

SAMPLE OUTPUT

25

20

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>

int main() {
    int A[3][3];
    int i, j;

    // Read 9 inputs in row-major order
    for (i = 0; i < 3; i++)
        for (j = 0; j < 3; j++)
            scanf("%d", &A[i][j]);

    int sum1 = 0, sum2 = 0;

    // Sum of A[0][0], A[0][2], A[1][1], A[2][0], A[2][2]
    sum1 = A[0][0] + A[0][2] + A[1][1] + A[2][0] + A[2][2];

    // Sum of A[0][1], A[1][0], A[1][2], A[2][1]
    sum2 = A[0][1] + A[1][0] + A[1][2] + A[2][1];

    printf("%d\n", sum1);
    printf("%d\n", sum2);

    return 0;
}
```

	Input	Expected	Got	
✓	1 2 3 4 5 6 7 8 9	25 20	25 20	✓
✓	21 422 423 443 586 645 657 846 904	2591 2356	2591 2356	✓
✓	201 212 673 685 16 604 762 147 762	2414 1648	2414 1648	✓
✓	121 75 63 85 1 4 464 125 791	1440 289	1440 289	✓

Passed all tests! ✓

Question **2**

Correct

Microsoft has come to hire interns from your college. N students got shortlisted out of which few were males and a few females. All the students have been assigned talent levels. Smaller the talent level, lesser is your chance to be selected. Microsoft wants to create the result list where it wants the candidates sorted according to their talent levels, but there is a catch. This time Microsoft wants to hire female candidates first and then male candidates.

The task is to create a list where first all-female candidates are sorted in a descending order and then male candidates are sorted in a descending order.

Input Format

The first line contains an integer N denoting the number of students. Next, N lines contain two space-separated integers, ai and bi.

The first integer, ai will be either 1(for a male candidate) or 0(for female candidate).

The second integer, bi will be the candidate's talent level.

Constraints

$1 <= N <= 10^5$

$0 <= ai <= 1$

$1 <= bi <= 10^9$

Output Format

Output space-separated integers, which first contains the talent levels of all female candidates sorted in descending order and then the talent levels of male candidates in descending order.

SAMPLE INPUT

5

0 3

1 6

0 2

0 7

1 15

SAMPLE OUTPUT

7 3 2 15 6

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <stdlib.h>

// Comparison function for descending order
int compare_desc(const void *a, const void *b) {
    long long x = *(long long *)a;
    long long y = *(long long *)b;
    if (x < y) return 1;
    else if (x > y) return -1;
    else return 0;
```

```c
}

int main() {
    int N;
    scanf("%d", &N);

    long long *female = (long long *)malloc(N * sizeof(long long));
    long long *male   = (long long *)malloc(N * sizeof(long long));

    int fcount = 0, mcount = 0;

    for (int i = 0; i < N; i++) {
        int gender;
        long long talent;
        scanf("%d %lld", &gender, &talent);

        if (gender == 0) {
            female[fcount++] = talent;
        } else {
            male[mcount++] = talent;
        }
    }

    // Sort both groups in descending order
    qsort(female, fcount, sizeof(long long), compare_desc);
    qsort(male, mcount, sizeof(long long), compare_desc);

    // Print female first
    for (int i = 0; i < fcount; i++) {
        printf("%lld ", female[i]);
    }

    // Then print male
    for (int i = 0; i < mcount; i++) {
        printf("%lld ", male[i]);
    }

    free(female);
    free(male);

    return 0;
}
```

	Input	Expected
✓	5 0 3 1 6 0 2 0 7 1 15	7 3 2 15 6
✓	6 0 1 0 26 0 39 0 37 0 7 0 13	39 37 26 13 7 1
✓	12 1 12 1 14 1 18 1 1 1 2 1 3 1 5 1 8 1 9 1 10 0 29 0 31	31 29 18 14 12 10 9 8 5 3 2 1
✓	12 0 12	12 12 12 12 12 12 12 12 12 12 12 12

	Input	Expected
	1 12 0 12 1 12 0 12 0 12 1 12 0 12 1 12 1 12 0 12 1 12	
✓	29 1 695 1 321 1 291 0 260 1 757 0 515 1 731 0 193 0 591 0 881 0 706 1 231 1 675 1 577 1 528 0 825 1 937 0 25 1 151 0 865 1 857 0 513 1 152 0 381 0 306 1 75 1 397 0 921 0 839	921 881 865 839 825 706 591 515 513 381 306 260 193 25 937 857 757 731 695 675 577 528 397 321 291 231 152 151 75
✓	79 1 355 1 345 0 704 1 601 0 289 0 862 0 841 0 24 0 345 1 171 0 968 1 657 0 386 1 437 1 750 0 251 0 947 1 969 0 37 0 389 0 821 1 573 1 751 1 801 0 369 1 839 0 311 1 521 1 441	968 947 931 906 894 865 862 841 821 791 772 721 721 704 682 673 653 621 601 593 569 526 501 477 463 401 389 386 369 347 345

	Input	Expected
	1 127 1 369 0 463 1 189 0 682 0 111 0 347 0 81 0 653 0 75 1 113 0 151 0 501 0 721 0 601 1 901 0 477 1 967 0 17 0 621 0 673 1 751 0 593 0 125 1 625 0 526 1 681 1 641 0 894 1 731 1 505 1 675 1 172 0 569 0 906 0 103 1 1 1 613 0 401 0 111 0 931 0 791 0 865 1 396 0 126 0 249 0 772 0 311 0 721 1 387	

Passed all tests!

Question **3**

Correct

Shyam Lal, a wealthy landlord from the state of Rajasthan, being an old fellow and tired of doing hard work, decided to sell all his farmland and to live rest of his life with that money. No other farmer is rich enough to buy all his land so he decided to partition the land into rectangular plots of different sizes with different cost per unit area. So, he sold these plots to the farmers but made a mistake. Being illiterate, he made partitions that could be overlapping. When the farmers came to know about it, they ran to him for compensation of extra money they paid to him. So, he decided to return all the money to the farmers of that land which was overlapping with other farmer's land to settle down the conflict. All the portion of conflicted land will be taken back by the landlord.

To decide the total compensation, he has to calculate the total amount of money to return back to farmers with the same cost they had purchased from him. Suppose, Shyam Lal has a total land area of ***1000 x 1000*** equal square blocks where each block is equivalent to a unit square area which can be represented on the co-ordinate axis. Now find the total amount of money, he has to return to the farmers. Help Shyam Lal to accomplish this task.

Input Format:

The first line of the input contains an integer ***N***, denoting the total number of land pieces he had distributed. Next ***N*** line contains the ***5*** space separated integers ***(X1, Y1)***, ***(X2, Y2)*** to represent a rectangular piece of land, and cost per unit area ***C***.

(X1, Y1) and ***(X2, Y2)*** are the locations of first and last square block on the diagonal of the rectangular region.

Output Format:

Print the total amount he has to return to farmers to solve the conflict.

Constraints:

1 ≤ N ≤ 100

1 ≤ X1 ≤ X2 ≤ 1000

1 ≤ Y1 ≤ Y2 ≤ 1000

1 ≤ C ≤ 1000

SAMPLE INPUT

3

1 4 4 6 1

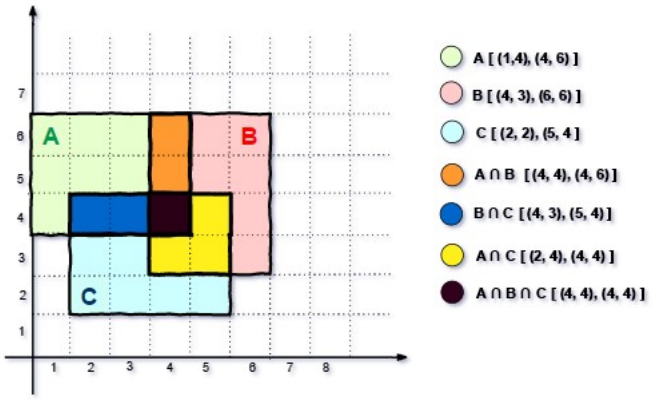
4 3 6 6 2

2 2 5 4 3

SAMPLE OUTPUT

35

Explanation



Simple Illustration of Distribution of Land

For given sample input (see given graph for reference), compensation money for different farmers is as follows:

Farmer with land area A: $C_1 = 5 * 1 = 5$

Farmer with land area B: $C_2 = 6 * 2 = 12$

Farmer with land area C: $C_3 = 6 * 3 = 18$

Total Compensation Money = $C_1 + C_2 + C_3 = 5 + 12 + 18 = 35$

Answer: (penalty regime: 0 %)

```c
#include <stdio.h>
#include <string.h>

int main() {
    int N;
    scanf("%d", &N);

    int X1[105], Y1_[105], X2[105], Y2[105], C[105];

    for (int i = 0; i < N; i++) {
        scanf("%d %d %d %d %d", &X1[i], &Y1_[i], &X2[i], &Y2[i], &C[i]);
    }

    // Max land size = 1000 x 1000
    static int coverCount[1001][1001];
    memset(coverCount, 0, sizeof(coverCount));

    // Step 1: Count coverage of each cell
    for (int k = 0; k < N; k++) {
        for (int x = X1[k]; x <= X2[k]; x++) {
            for (int y = Y1_[k]; y <= Y2[k]; y++) {
                coverCount[x][y]++;
            }
        }
    }

    long long totalCompensation = 0;

    // Step 2: For each farmer, count how many of its cells are overlapped
    for (int k = 0; k < N; k++) {

        long long overlapCells = 0;

        for (int x = X1[k]; x <= X2[k]; x++) {
            for (int y = Y1_[k]; y <= Y2[k]; y++) {

                if (coverCount[x][y] > 1) {
                    // This cell overlaps with at least one other farmer
                    overlapCells++;
                }
            }
        }

        totalCompensation += overlapCells * C[k];
    }
```

```
46
47      printf("%lld", totalCompensation);
48
49      return 0;
50  }
```

	Input	Expected	Got	
✓	3 1 4 4 6 1 4 3 6 6 2 2 2 5 4 3	35	35	✓
✓	1 48 12 49 27 8	0	0	✓
✓	3 88 34 99 76 44 82 65 94 100 81 58 16 65 66 7	10500	10500	✓
✓	63 3 84 69 93 37 28 59 42 93 17 31 33 67 45 83 64 23 71 76 62 18 15 92 92 67 32 35 74 90 47 78 59 99 76 66 19 32 94 89 53 77 49 83 52 63 79 15 85 22 91 71 84 78 85 11 95 64 97 65 75 87 57 92 67 65 20 91 76 98 61 60 57 87 69 98 94 17 97 61 37 43 54 85 64 24 62 67 88 76 97 9 62 84 87 41 6 38 13 78 26 98 42 99 82 74 11 91 31 93 65 77 62 81 68 97 20 17 26 83 4 43 20 73 85 32 32 89 86 98 21 15 86 41 98 36 93 85 95 96 16 55 15 62 88 16 3 15 24 79 85 92 61 98 100 54 55 47 75 90 82 95 58 100 65 3 66 10 90 55 22 52 28 66 42 62 52 17 70 28 10 45 57 63 92 23 37 21 47 92 14 80 68 95 77 95 12 62 76 74 87 58 48 65 71 11 92 34 95 80 65 66 47 87 54 24 1 1 79 68 80 78 93 92 94 41 35 90 37 94 33 79 40 91 63 93 53 97 61 99 9 15 53 30 58 68	2659214	2659214	✓

	Input	Expected	Got	
	71 15 74 29 26 67 75 100 90 95 42 19 62 65 66 17 87 84 96 84 21 16 73 24 16 5 63 12 77 23 68 70 83 74 66 96 45 98 89 21 38 31 93 63 78 39 20 84 47 46 70 85 71 91 14 12 46 65 89 49 69 71 98 89 54 11 39 95 97 36			
✓	100 265 84 385 531 154 467 71 634 683 884 3 853 947 934 658 856 698 933 762 374 53 427 524 670 884 793 151 840 584 241 571 828 712 853 910 637 623 860 977 81 189 796 755 838 582 105 843 451 947 52 239 77 623 358 633 114 721 626 834 785 178 34 995 988 665 155 43 508 263 63 994 186 995 842 314 812 326 953 900 131 95 110 787 673 41 636 444 958 547 974 659 497 969 672 993 140 546 995 981 112 467 498 496 556 802 618 625 869 892 746 399 703 690 745 362 875 930 892 1000 31 323 538 484 868 484 784 383 969 903 655 444 373 921 881 716 711 699 960 720 623 400 400 697 896 799 556 116 649 417 770 969 199 992 810 923 936 814 991 890 788 239 416 572 421 153 122 723 954 953 470 675 1 802 675 59 680 688 875 913 876 761 320 879 719 415 583 889 853 905 825 34 672 510 762 640 817 399 988 738 798 282 602 641 914 621 570 496 805 988 596 253 712 385 779 549 558 39 566 41 26 58 194 881 212 78 607 408 749 588 205 835 729 870 927 138 258 674 922 931 423 723 610 735 621 374 574 717 1000 870 751 873 145 896 755 725 341 585 879 601 92 683 856 897 893 327 707 426 758 928 300 401 409 910 422 504 335 878 337 977 786 543 886 616 913 422	2823896365	2823896365	✓

	Input	Expected	Got	
	352 331 420 484 70 119 66 369 185 43 499 906 953 986 307 774 270 915 359 149 846 335 886 972 256 305 857 545 910 733 624 871 957 929 28 158 749 459 890 519 453 864 730 900 655 203 816 725 847 547 373 145 468 995 658 33 434 759 486 144 438 652 535 658 371 667 221 880 660 161 94 34 457 98 516 980 524 985 830 201 937 219 977 920 462 819 761 838 930 139 170 908 589 927 180 168 247 377 283 70 846 933 893 946 306 813 684 850 713 473 925 883 980 953 782 567 234 881 984 395 213 443 771 655 940 888 964 933 981 478 465 431 675 952 597 906 227 935 970 399 289 988 541 993 970 182 13 388 891 311 417 642 604 945 5 492 806 544 913 130 139 596 637 897 304 863 721 925 768 55 966 792 973 930 254 924 80 972 302 84 109 496 167 591 387 126 420 704 967 358 636 51 685 371 243 659 859 870 949 459 165 427 496 704 755 35 442 225 698 55 744 389 762 869 261			

Passed all tests!