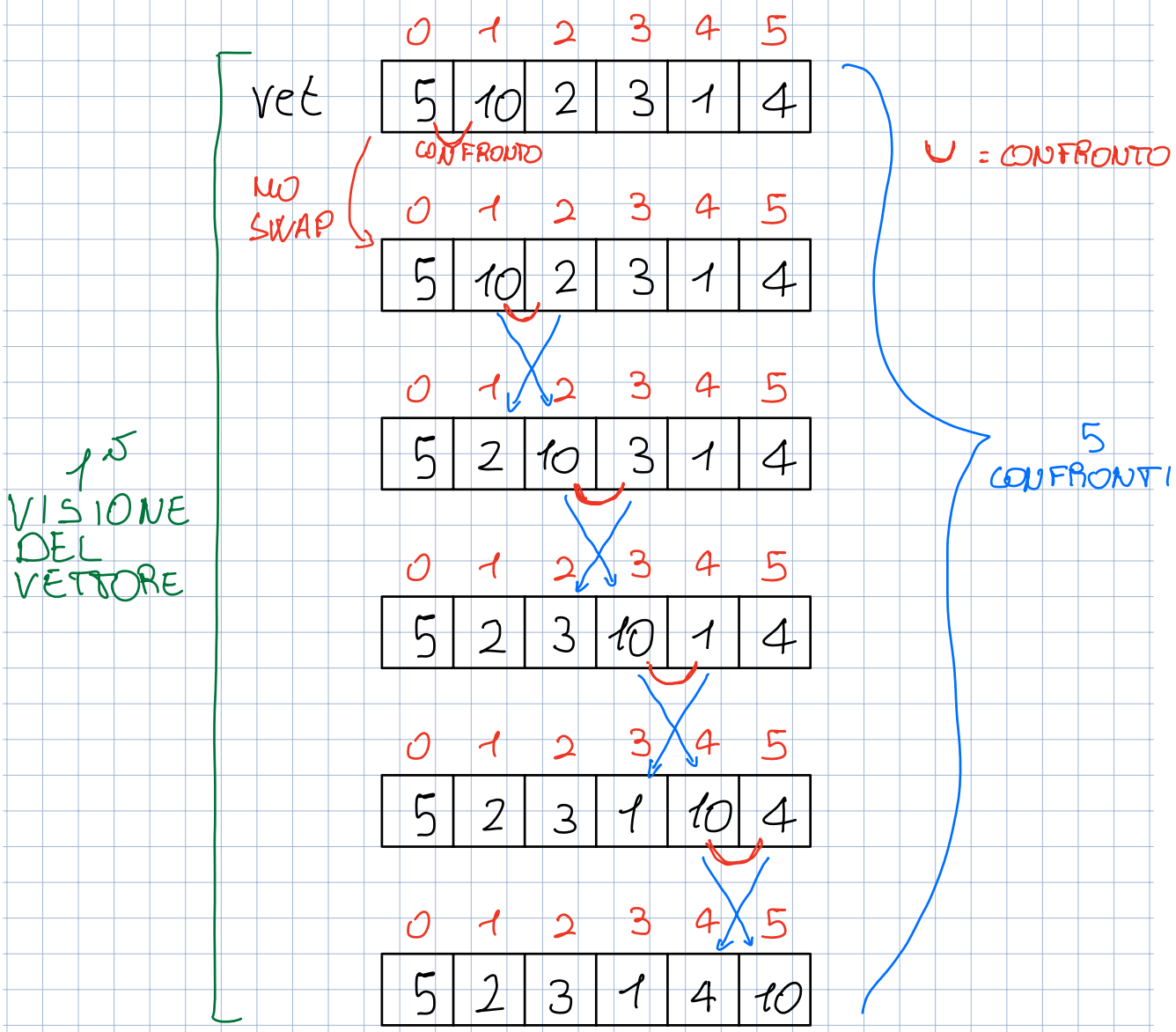
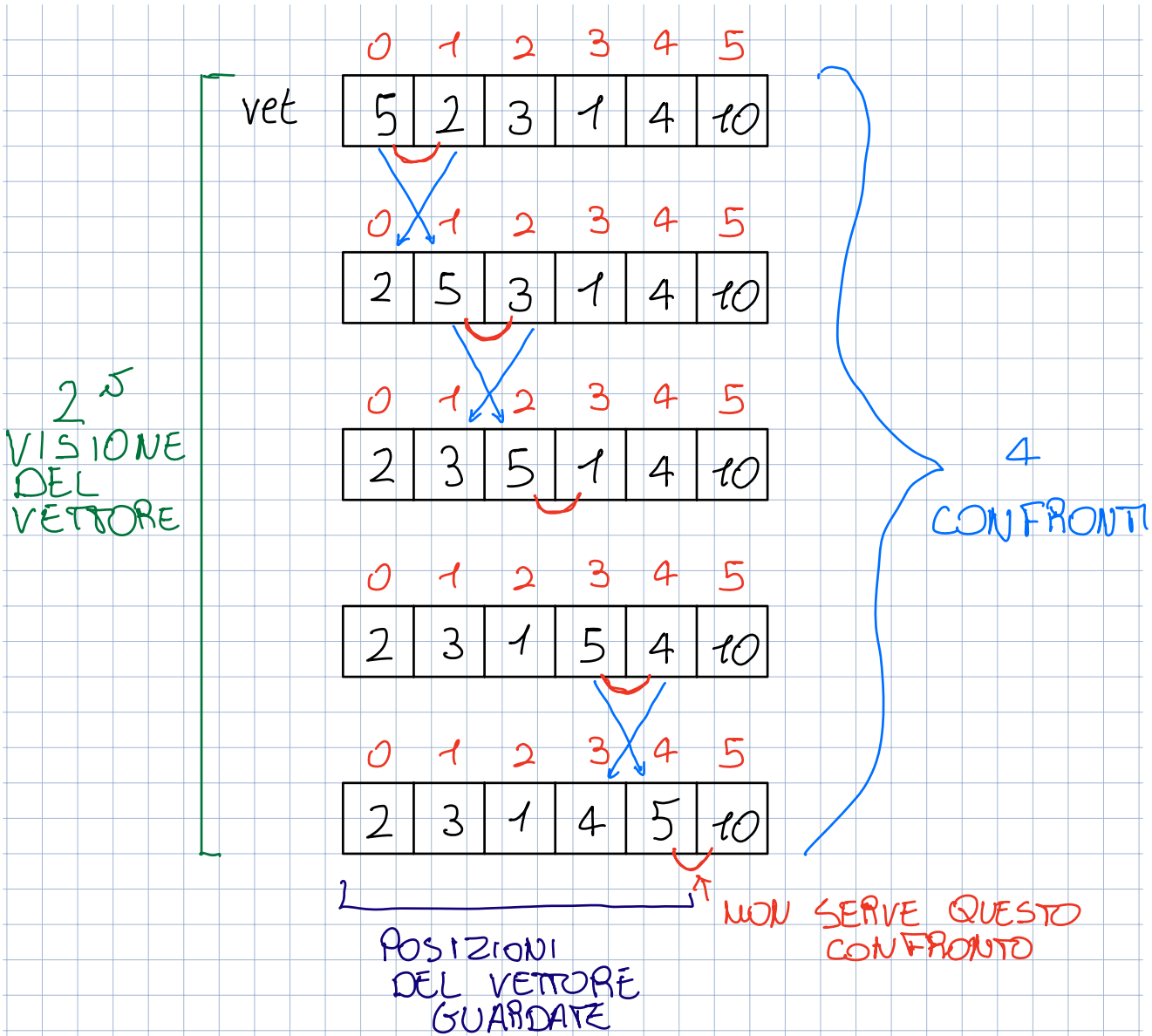
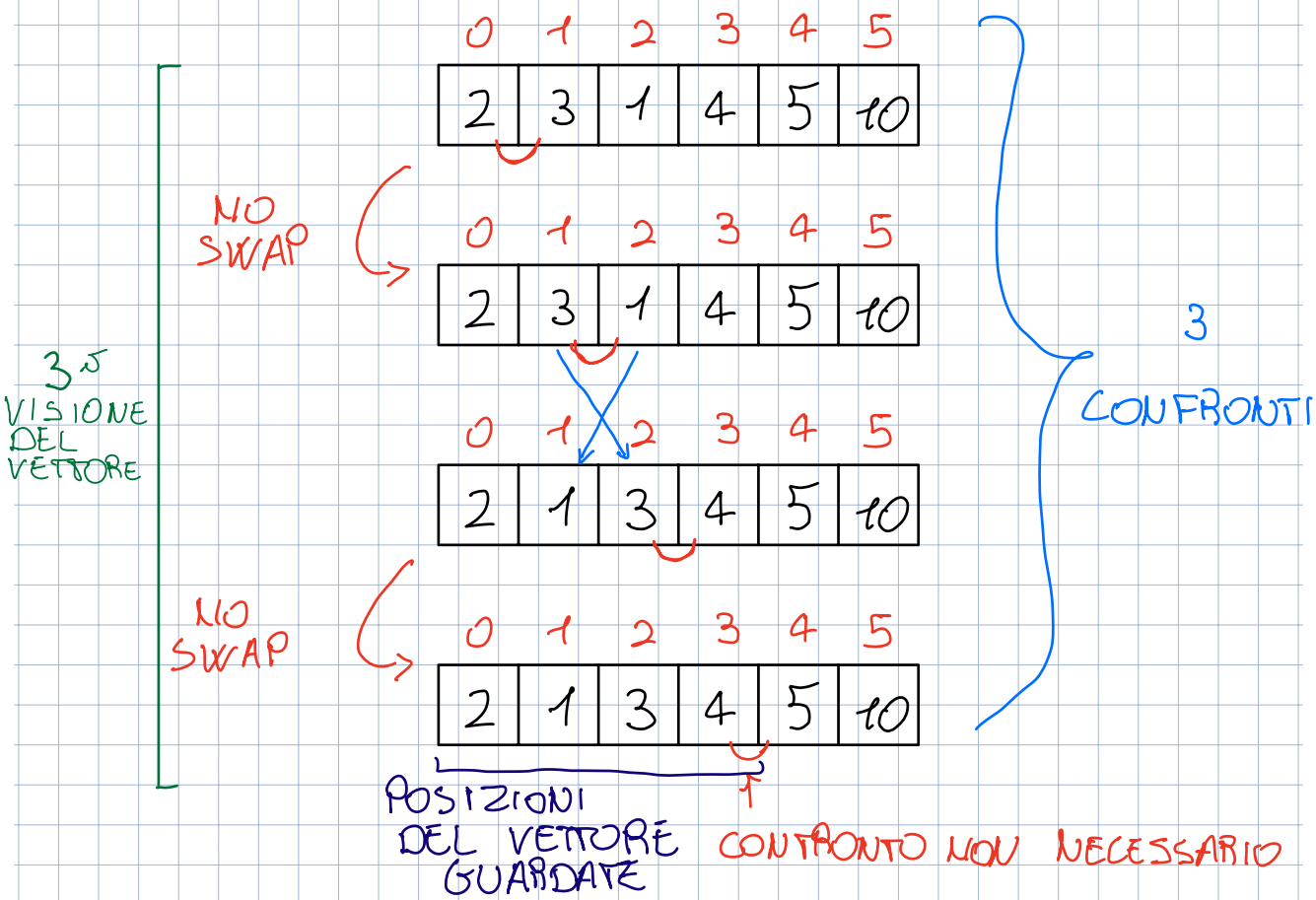
BUBBLE SORT

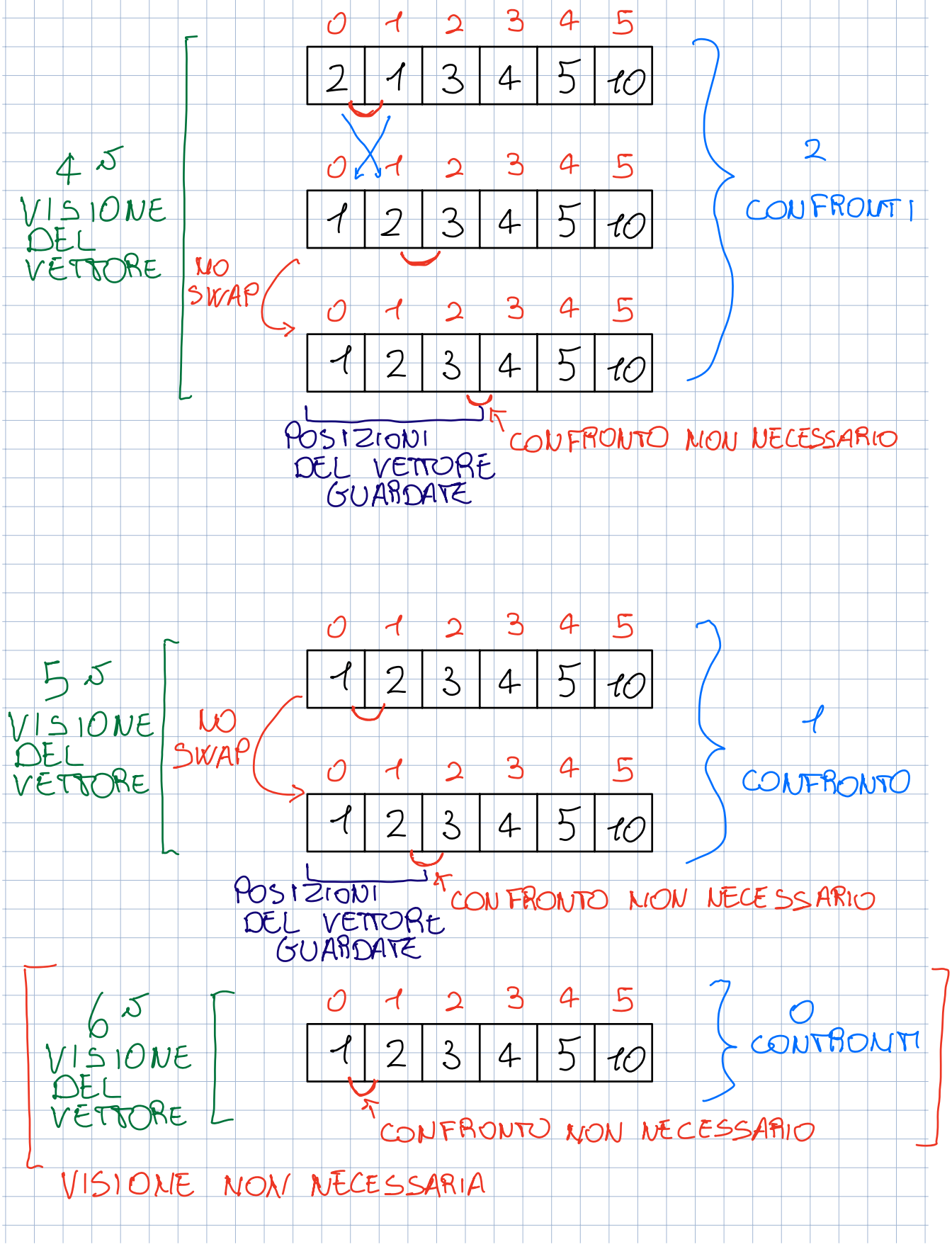
PRIMA DEL main() → #define DIM 6

NEL main() → int vet[DIM] = { 5, 10, 2, 3, 1, 4 };




vet
0 1 2 3 4 5
5 2 3 1 4 10
2 5 3 1 4 10
2 3 5 1 4 10
2 3 1 5 4 10
2 3 1 4 5 10
2 5
VISIONE DEL VETTORE
4
CONFRONTI
2
POSIZIONI DEL VETTORE GUARDATE
NON SERVE QUESTO CONFRONTO


0 1 2 3 4 5
2 3 1 4 5 10
NO SWAP
0 1 2 3 4 5
2 3 1 4 5 10
0 1 2 3 4 5
2 1 3 4 5 10
NO SWAP
0 1 2 3 4 5
2 1 3 4 5 10
3 VISIONE DEL VETTORE
3 CONFRONTI
POSIZIONI DEL VETTORE GUARDATE
CONFRONTO NON NECESSARIO


4ª VISIONE DEL VETTORE
0 1 2 3 4 5
2 1 3 4 5 10
0 1 2 3 4 5
1 2 3 4 5 10
NO SWAP
0 1 2 3 4 5
1 2 3 4 5 10
2 CONFRONTI
POSIZIONI DEL VETTORE GUARDATE
CONFRONTO NON NECESSARIO
5ª VISIONE DEL VETTORE
0 1 2 3 4 5
1 2 3 4 5 10
NO SWAP
0 1 2 3 4 5
1 2 3 4 5 10
1 CONFRONTO
POSIZIONI DEL VETTORE GUARDATE
CONFRONTO NON NECESSARIO
6ª VISIONE DEL VETTORE
0 1 2 3 4 5
1 2 3 4 5 10
0 CONFRONTI
CONFRONTO NON NECESSARIO
VISIONE NON NECESSARIA

FUNZIONE BUBBLE SORT

```
void bubbleSort(int vet[], int dim){
    int i, j;
    int tmp;

    for(i=0; i<(dim-1); i++){
        for(j=0; j<(dim-i-1); j++){
            if(vet[j] > vet[j+1]){
                tmp = vet[j];
                vet[j] = vet[j+1];
                vet[j+1] = tmp;
            }
        }
    }
}
```

PER L'ORDINE CRESCENTE