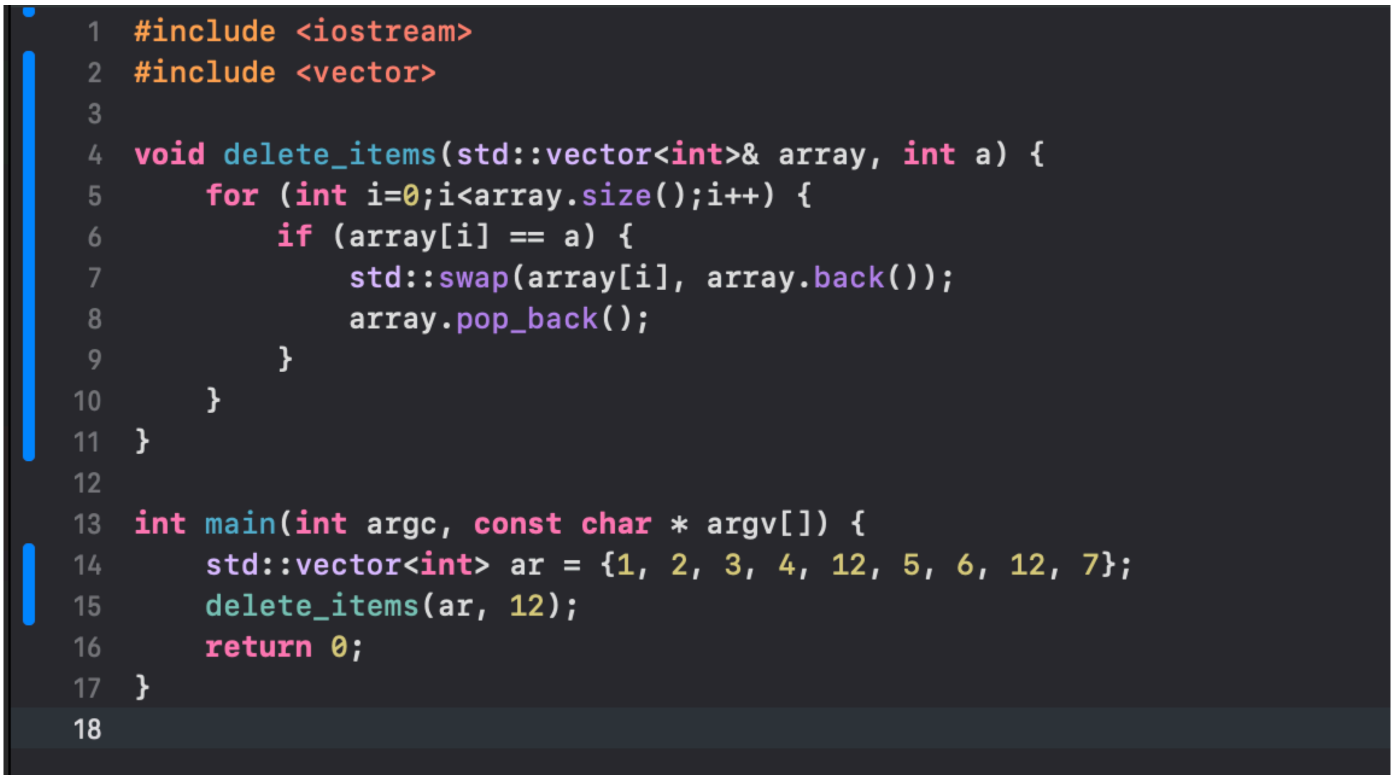
P1



```cpp
#include <iostream>
#include <vector>

void delete_items(std::vector<int>& array, int a) {
    for (int i=0;i<array.size();i++) {
        if (array[i] == a) {
            std::swap(array[i], array.back());
            array.pop_back();
        }
    }
}

int main(int argc, const char * argv[]) {
    std::vector<int> ar = {1, 2, 3, 4, 12, 5, 6, 12, 7};
    delete_items(ar, 12);
    return 0;
}

```

P2

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
void delete_items(std::vector<int>& array, int a) {
    for (int i=0;i<array.size();i++) {
        if (array[i] == a) {
            std::swap(array[i], array.back());
            array.pop_back();
        }
    }
}

void modify_top(std::vector<int>& arr, int num) {
    if (num > arr[0]) {
        arr[2] = arr[1];
        arr[1] = arr[0];
        arr[0] = num;
    }
    else if (num > arr[1]) {
        arr[2] = arr[1];
        arr[1] = num;
    }
    else {
        arr[2] = num;
    }
}

void modify_bottom(std::vector<int>& arr, int num) {
    if (num < arr[0]) {
        arr[2] = arr[1];
        arr[1] = arr[0];
        arr[0] = num;
    }
    else if (num < arr[1]) {
        arr[2] = arr[1];
        arr[1] = num;
    }
    else {
        arr[2] = num;
    }
}


void top_bottom_3(const std::vector<int>& array, std::vector<int>& top3,
std::vector<int>& bottom3) {
    for (int i=0;i<std::min(3,(int)array.size());i++) {
        top3.push_back(array[i]);
        bottom3.push_back(array[i]);
    }
    for (int i=3;i<array.size();i++) {
        if (array[i] > top3.back()) {
            modify_top(top3, array[i]);
        }
        if (array[i] < bottom3[0]) {
            modify_bottom(bottom3, array[i]);
        }
    }
}
int main(int argc, const char * argv[]) {
    std::vector<int> ar = {1, 2, 3, 4, 12, 5, 6, 88, -1, 77, 12, 7};
    std::vector<int> res_top3;
    std::vector<int> res_bottom3;

    top_bottom_3(ar, res_top3, res_bottom3);
    return 0;
}
```

P3

```
#include <iostream>


// graph data typmp
const int maxn = 1mp3+5;
const int INF = 0x3f3f3f3f;
int mp[maxn][maxn];

void search_shortest(const T& graph, int src, int dest, std::vector<int>& result) {
    for(k=1;k<=n;k++) {
        for(i=1;i<=n;i++) {
            for(j=1;j<=n;j++) {
                if(mp[i][j] > mp[i][k] + mp[k][j] )
                    mp[i][j] = mp[i][k] + mp[k][j];
            }
        }
    }
    res.push_back(mp[src][dest]);
}


int main() {
    // todo initial graph data
    int n, m; // n vertex,  m edges
    std::cin >> n >> m;
    for(i=1;i<=n;i++) {
        for(j=1;j<=n;j++) {
            if(i == j) {
                mp[i][j] = 0;
            }
            else {
                mp[i][j] = INF;
            }
        }
    }
    // input edge
    for(i=1;i<=m;i++)
    {
        std::cin >> t1 >> t2 >> t3;
        mp[t1][t2]=t3;
    }
    return 0;
}
```