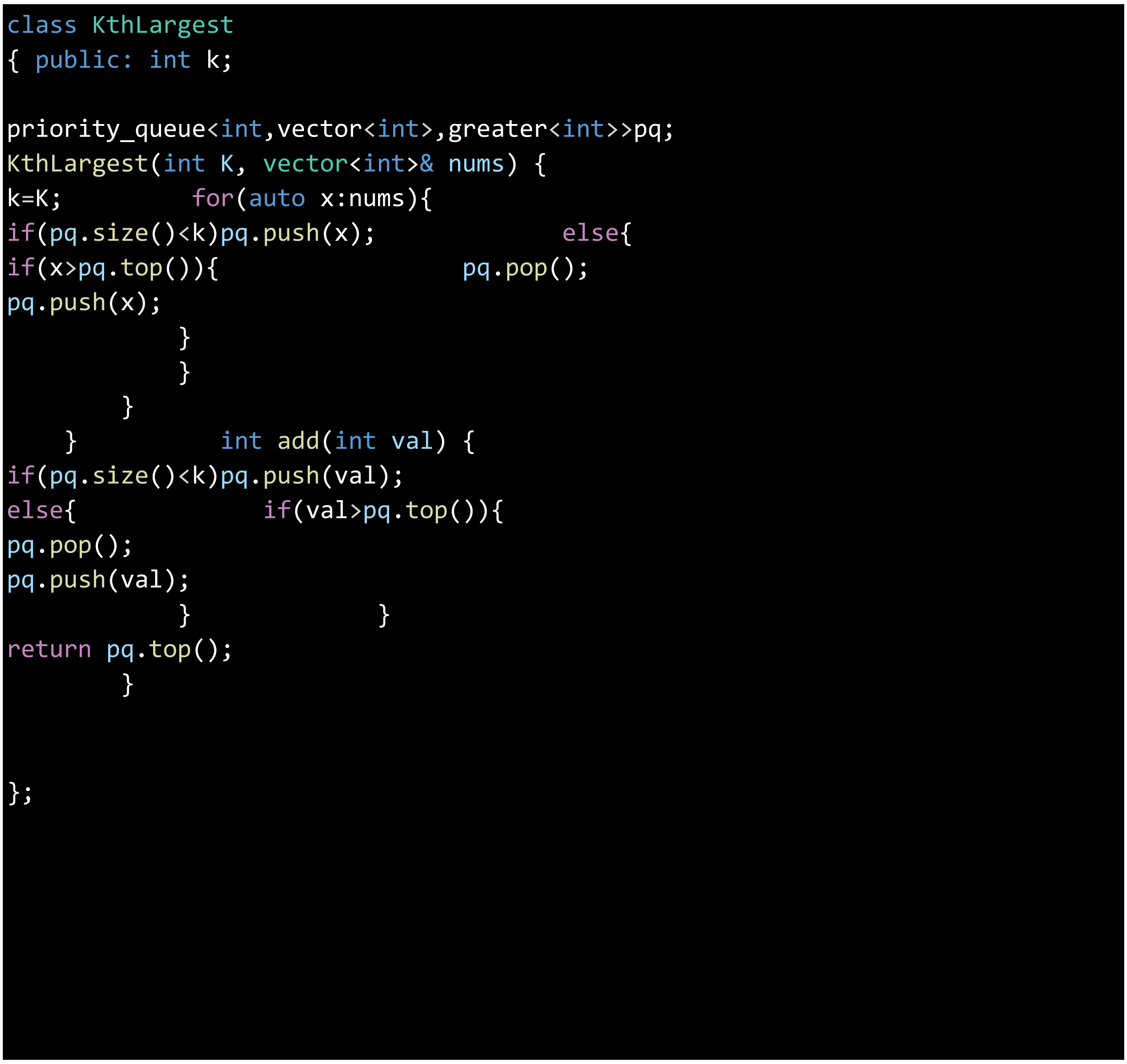
Assignment 1 priority_queue

Q.1 Kth largest element Code:-



```
class KthLargest
{ public: int k;

priority_queue<int,vector<int>,greater<int>>pq;
KthLargest(int K, vector<int>& nums) {
k=K;         for(auto x:nums){
if(pq.size()<k)pq.push(x);             else{
if(x>pq.top()){                 pq.pop();
pq.push(x);
            }
            }
        }
    }          int add(int val) {
if(pq.size()<k)pq.push(val);
else{             if(val>pq.top()){
pq.pop();
pq.push(val);
            }             }
return pq.top();
        }



};
```

Q.2 K closest points to origin LEETCODE:-973

Code:-

```
        vector<vector<int>>ans;
priority_queue<pair<int,pair<int,int>>>pq;          for(int
i=0;i<points.size();i++){              int
dis=points[i][0]*points[i][0]+points[i][1]*points[i][1];
if(pq.size()<k)pq.push({dis,{points[i][0],points[i][1]}});
else{            if(dis<pq.top().first){                 pq.pop();
pq.push({dis,{points[i][0],points[i][1]}});
            }
            }          }             while(!pq.empty()){
ans.push_back({pq.top().second.first,pq.top().second.second});
pq.pop();

}
        return ans;
    }
};
```

Q.3 Merge k sorted lists LEETCODE:-23

```
class Solution {
    ListNode* merge(ListNode* a, ListNode* b) {
        ListNode*c=new ListNode(100);
        ListNode*temp=c;
        while(a!=NULL&&b!=NULL){
            if(a->val > b->val){
                temp->next=b;
                b=b->next;
            temp=temp->next;
            }
            else{
                temp->next=a;
                a=a->next;
            temp=temp->next;
            }

        }
        if(a==NULL)temp->next=b;
        else temp->next=a;
        return c->next;
    }
public:
    ListNode* mergeKLists(vector<ListNode*>& arr) {
        if(arr.size()==0)return NULL;
        while(arr.size()>1){
            ListNode*a=arr[0];
            arr.erase(arr.begin());
            ListNode*b=arr[0];
            arr.erase(arr.begin());
            ListNode*c=merge(a,b);
            arr.push_back(c);
        }
        return arr[0];
    }
};
```