

1. Make a single linked list of integers. There should be at least 15 nodes,. The list should not be sorted. Traverse the list.
Now sort the list using **Bubble sort/selection sort**. One of these 2 - do not use any other sorting algorithm. The list should be sorted such that your program unlinks the nodes and relinks them so that they are sorted. (DO NOT SWAP THE VALUES IN THE NODES).
Traverse the list again.

Code file is submitted separately for the question.

Driver code:

- 1) Input the data of nodes in the LLelements array.
- 2) insertnode function creates a node for every element in the array.
- 3) BubbleSort functions sorts the created Linked List by swapping nodes.
- 4) printLinkedList function prints the Linked List. Here the function prints the final sorted Linked List.

```
int main() {  
    Node* head = NULL;  
    vector<int> LLelements {20,10,30,4,122,12332,41,543,12,44,35,987,23,212,332,112112,1};  
    for(int i=0;i<LLelements.size();i++){  
        insertnode(head,LLelements[i]);  
    }  
    BubbleSort(head);  
    cout << "Final Sorted Linked List" << endl;  
    printLinkedList(head);  
}
```

INPUT: {20,10,30,4,122,12332,41,543,12,44,35,987,23,212,332,112112,1};

Screenshots of every Iteration: Refer to next page

LinkedList after 1th iteration

10 20 4 30 122 41 543 12 44 35 987 23 212 332 12332 1 112112

LinkedList after 2th iteration

10 4 20 30 41 122 12 44 35 543 23 212 332 987 1 12332 112112

LinkedList after 3th iteration

4 10 20 30 41 12 44 35 122 23 212 332 543 1 987 12332 112112

LinkedList after 4th iteration

4 10 20 30 12 41 35 44 23 122 212 332 1 543 987 12332 112112

LinkedList after 5th iteration

4 10 20 12 30 35 41 23 44 122 212 1 332 543 987 12332 112112

LinkedList after 6th iteration

4 10 12 20 30 35 23 41 44 122 1 212 332 543 987 12332 112112

LinkedList after 7th iteration

4 10 12 20 30 23 35 41 44 1 122 212 332 543 987 12332 112112

LinkedList after 8th iteration

4 10 12 20 23 30 35 41 1 44 122 212 332 543 987 12332 112112

LinkedList after 9th iteration

4 10 12 20 23 30 35 1 41 44 122 212 332 543 987 12332 112112

LinkedList after 10th iteration

4 10 12 20 23 30 1 35 41 44 122 212 332 543 987 12332 112112

LinkedList after 11th iteration

4 10 12 20 23 1 30 35 41 44 122 212 332 543 987 12332 112112

LinkedList after 12th iteration

4 10 12 20 1 23 30 35 41 44 122 212 332 543 987 12332 112112

LinkedList after 13th iteration

4 10 12 1 20 23 30 35 41 44 122 212 332 543 987 12332 112112

LinkedList after 14th iteration

4 10 1 12 20 23 30 35 41 44 122 212 332 543 987 12332 112112

LinkedList after 15th iteration

4 1 10 12 20 23 30 35 41 44 122 212 332 543 987 12332 112112

LinkedList after 16th iteration

1 4 10 12 20 23 30 35 41 44 122 212 332 543 987 12332 112112

LinkedList after 17th iteration

1 4 10 12 20 23 30 35 41 44 122 212 332 543 987 12332 112112

Final Sorted Linked List

1 4 10 12 20 23 30 35 41 44 122 212 332 543 987 12332 112112

☛ ☐

2. We covered binary search algorithm for an array. Write program similar to binary search, but now divide the list (array) into 3 parts each time. So this would be tertiary search algorithm.

Your **program must be recursive**,

run it 2 times .. once try to search a number in the list (array)

Second time search a number not in the list (array)

Submit the code and screen shot of executions

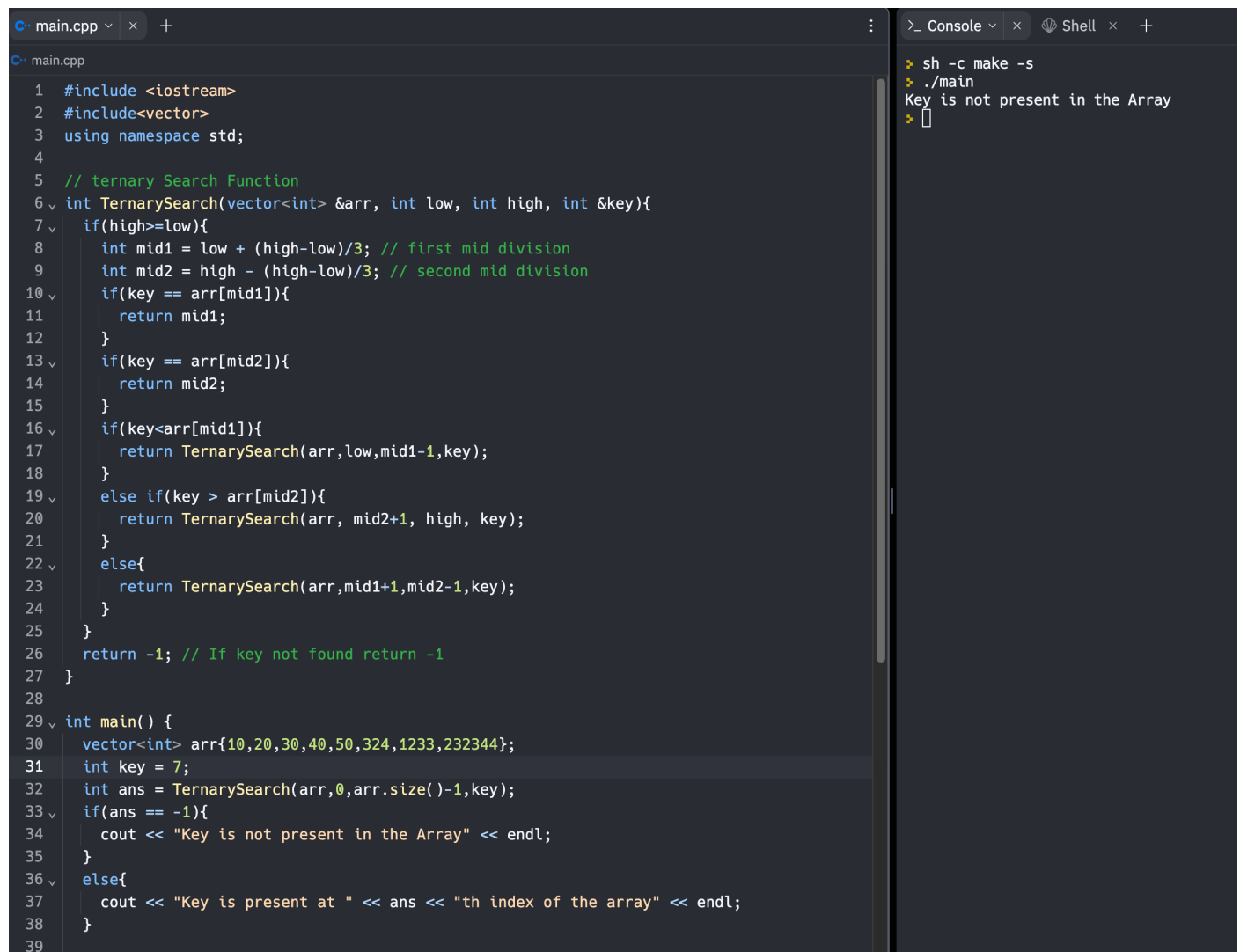
Code file is submitted separately for the question.

Case 1: Key is not present in the Array

Input = {10,20,30,40,50,324,1233,232344}

Key = 7

Output :



```
main.cpp x +
main.cpp
1 #include <iostream>
2 #include<vector>
3 using namespace std;
4
5 // ternary Search Function
6 int TernarySearch(vector<int> &arr, int low, int high, int &key){
7     if(high<=low){
8         int mid1 = low + (high-low)/3; // first mid division
9         int mid2 = high - (high-low)/3; // second mid division
10        if(key == arr[mid1]){
11            return mid1;
12        }
13        if(key == arr[mid2]){
14            return mid2;
15        }
16        if(key<arr[mid1]){
17            return TernarySearch(arr,low,mid1-1,key);
18        }
19        else if(key > arr[mid2]){
20            return TernarySearch(arr, mid2+1, high, key);
21        }
22        else{
23            return TernarySearch(arr,mid1+1,mid2-1,key);
24        }
25    }
26    return -1; // If key not found return -1
27 }
28
29 int main() {
30     vector<int> arr{10,20,30,40,50,324,1233,232344};
31     int key = 7;
32     int ans = TernarySearch(arr,0,arr.size()-1,key);
33     if(ans == -1){
34         cout << "Key is not present in the Array" << endl;
35     }
36     else{
37         cout << "Key is present at " << ans << "th index of the array" << endl;
38     }
39 }
```

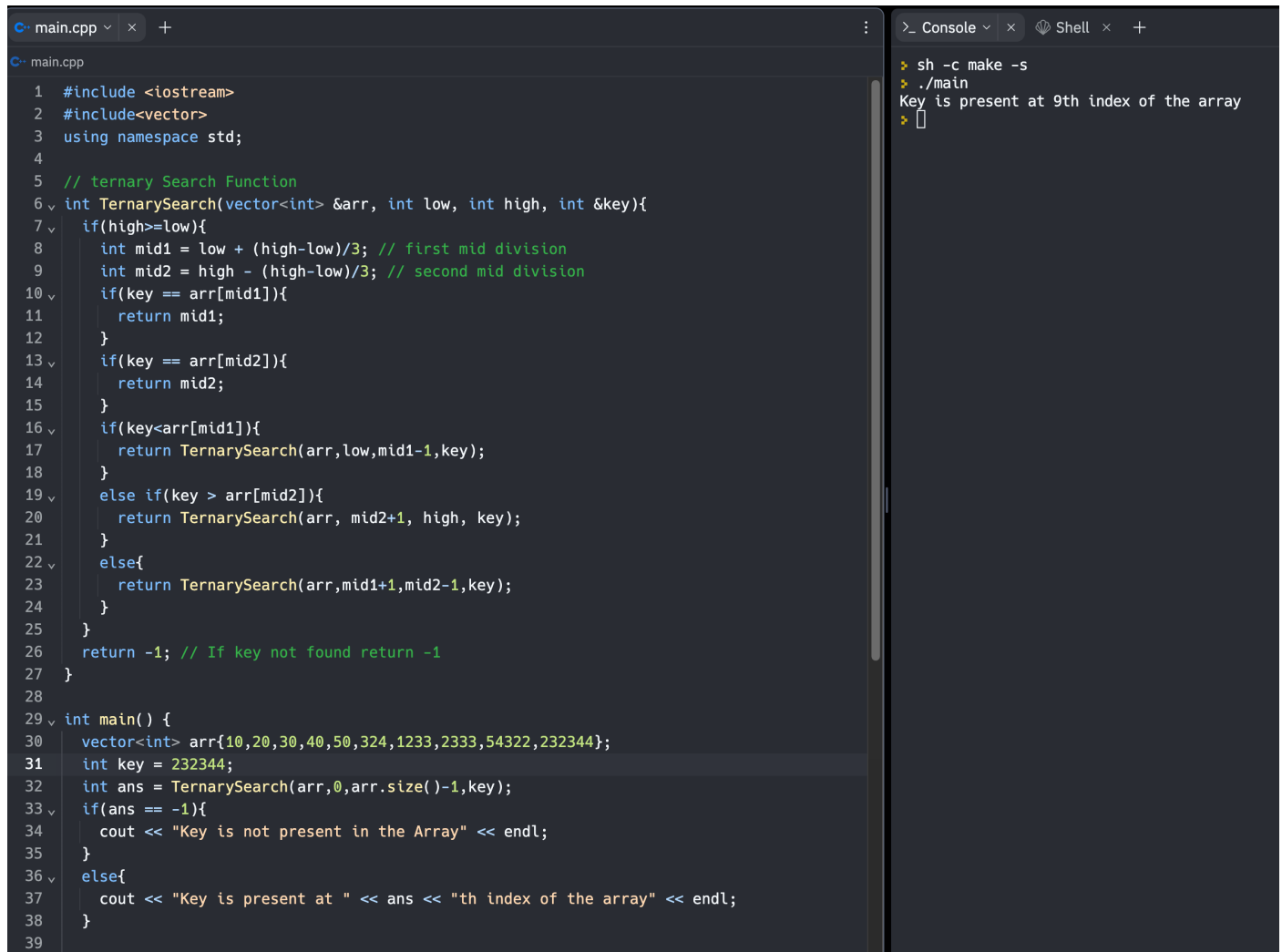
```
_ Console x Shell x +
sh -c make -s
./main
Key is not present in the Array
```

Case 2: Key is present in the Array

Input = {10,20,30,40,50,324,1233,2333,54322,232344};

Key = 232344

Output:



```
1 #include <iostream>
2 #include<vector>
3 using namespace std;
4
5 // ternary Search Function
6 int TernarySearch(vector<int> &arr, int low, int high, int &key){
7     if(high>=low){
8         int mid1 = low + (high-low)/3; // first mid division
9         int mid2 = high - (high-low)/3; // second mid division
10        if(key == arr[mid1]){
11            return mid1;
12        }
13        if(key == arr[mid2]){
14            return mid2;
15        }
16        if(key<arr[mid1]){
17            return TernarySearch(arr,low,mid1-1,key);
18        }
19        else if(key > arr[mid2]){
20            return TernarySearch(arr, mid2+1, high, key);
21        }
22        else{
23            return TernarySearch(arr,mid1+1,mid2-1,key);
24        }
25    }
26    return -1; // If key not found return -1
27 }
28
29 int main() {
30     vector<int> arr{10,20,30,40,50,324,1233,2333,54322,232344};
31     int key = 232344;
32     int ans = TernarySearch(arr,0,arr.size()-1,key);
33     if(ans == -1){
34         cout << "Key is not present in the Array" << endl;
35     }
36     else{
37         cout << "Key is present at " << ans << "th index of the array" << endl;
38     }
39 }
```

```
>_ Console x Shell x +
sh -c make -s
./main
Key is present at 9th index of the array
```