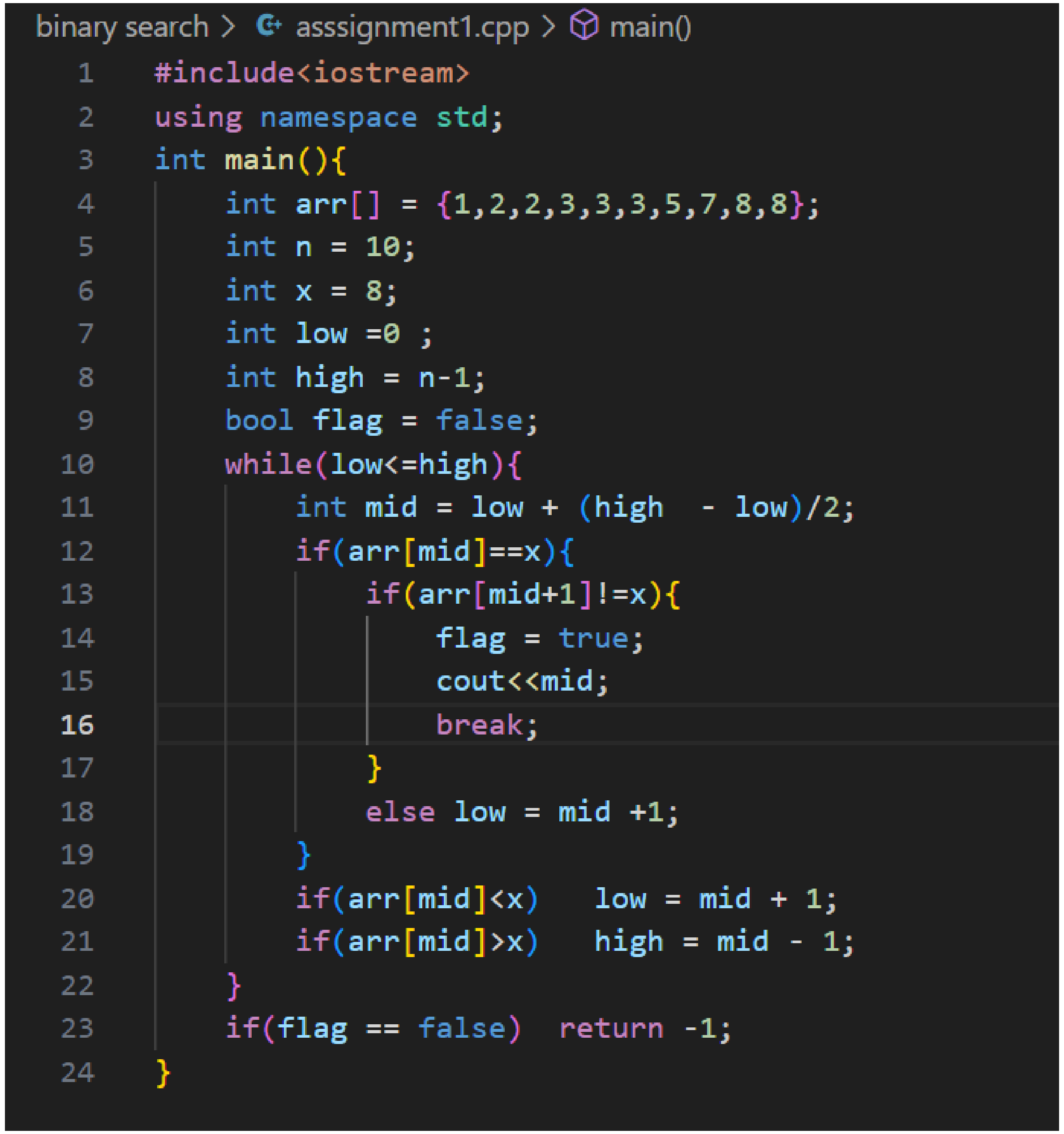
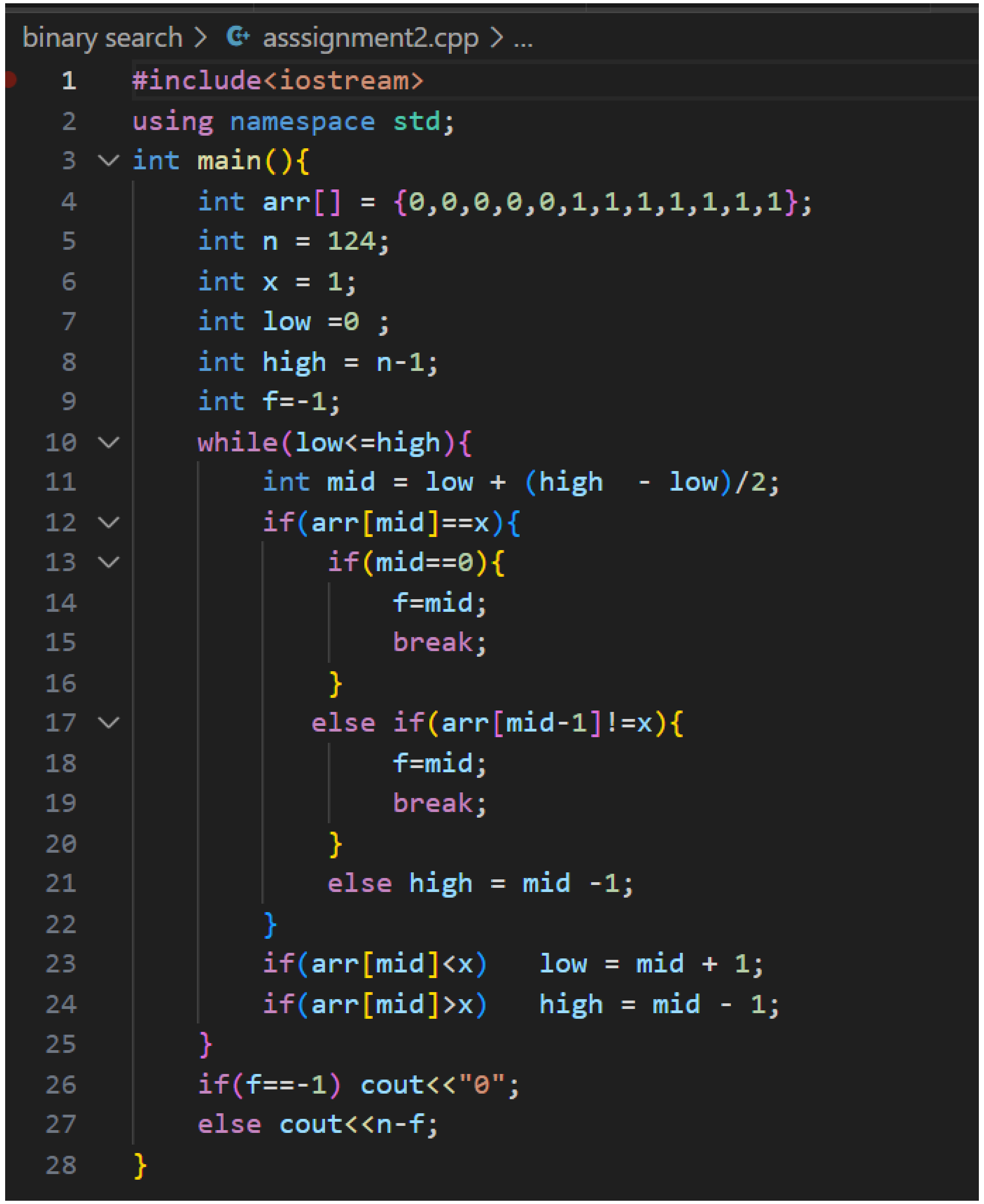
1. **Given a matrix having 0-1 only where each row is sorted in increasing order, find the row with the maximum number of 1's.**


binary search > asssignment1.cpp > main()

```cpp
#include<iostream>
using namespace std;
int main(){
    int arr[] = {1,2,2,3,3,3,5,7,8,8};
    int n = 10;
    int x = 8;
    int low =0 ;
    int high = n-1;
    bool flag = false;
    while(low<=high){
        int mid = low + (high  - low)/2;
        if(arr[mid]==x){
            if(arr[mid+1]!=x){
                flag = true;
                cout<<mid;
                break;
            }
            else low = mid +1;
        }
        if(arr[mid]<x)   low = mid + 1;
        if(arr[mid]>x)   high = mid - 1;
    }
    if(flag == false)  return -1;
}
```

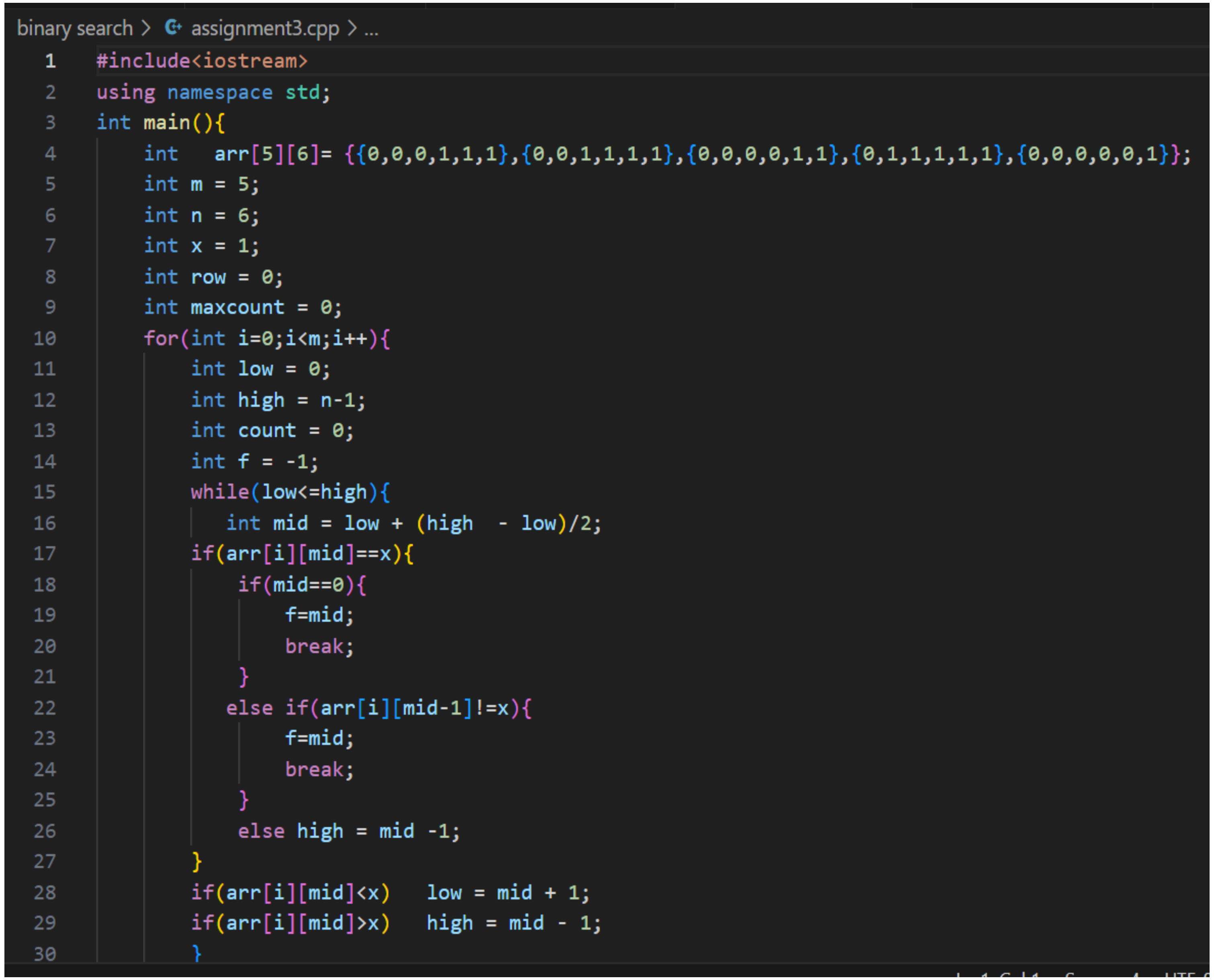
2. Given a sorted binary array, efficiently count the total number of 1’s in it.



binary search > asssignment2.cpp > ...

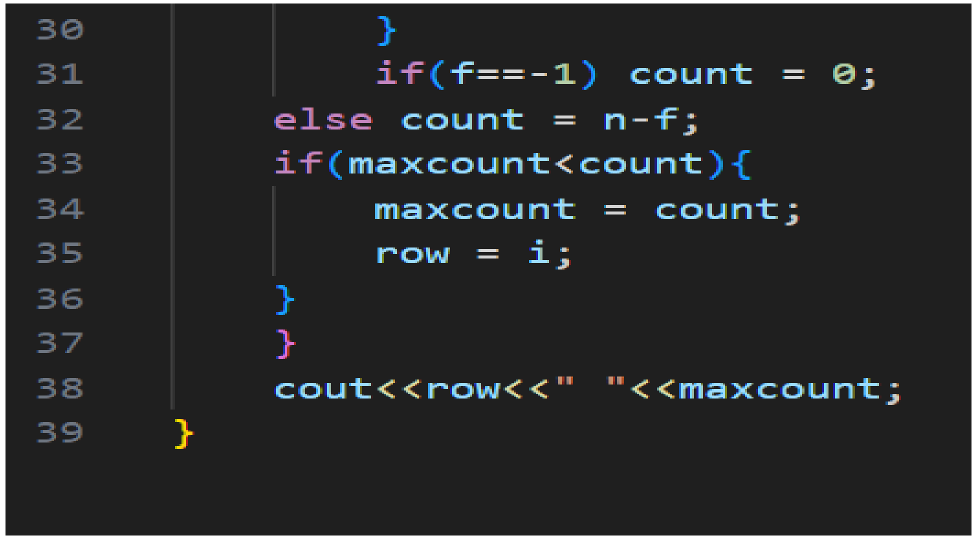
```cpp
#include<iostream>
using namespace std;
int main(){
    int arr[] = {0,0,0,0,0,1,1,1,1,1,1,1};
    int n = 124;
    int x = 1;
    int low =0 ;
    int high = n-1;
    int f=-1;
    while(low<=high){
        int mid = low + (high  - low)/2;
        if(arr[mid]==x){
            if(mid==0){
                f=mid;
                break;
            }
           else if(arr[mid-1]!=x){
                f=mid;
                break;
            }
            else high = mid -1;
        }
        if(arr[mid]<x)   low = mid + 1;
        if(arr[mid]>x)   high = mid - 1;
    }
    if(f==-1) cout<<"0";
    else cout<<n-f;
}
```

3. Given a matrix having 0-1 only where each row is sorted in increasing order, find the row with the maximum number of 1’s.



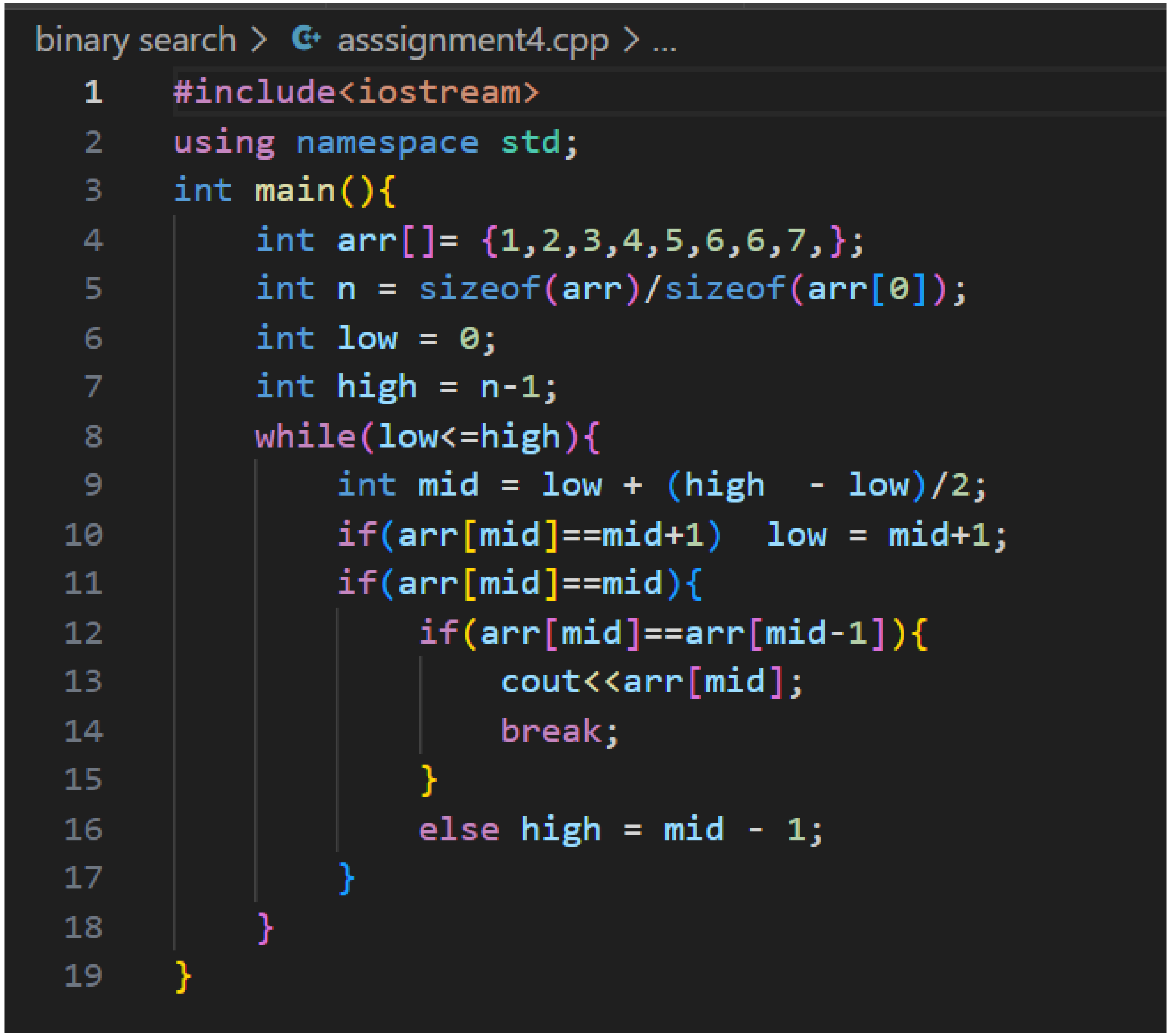
binary search > assignment3.cpp > ...

```cpp
#include<iostream>
using namespace std;
int main(){
    int   arr[5][6]= {{0,0,0,1,1,1},{0,0,1,1,1,1},{0,0,0,0,1,1},{0,1,1,1,1,1},{0,0,0,0,0,1}};
    int m = 5;
    int n = 6;
    int x = 1;
    int row = 0;
    int maxcount = 0;
    for(int i=0;i<m;i++){
        int low = 0;
        int high = n-1;
        int count = 0;
        int f = -1;
        while(low<=high){
           int mid = low + (high  - low)/2;
        if(arr[i][mid]==x){
            if(mid==0){
                f=mid;
                break;
            }
           else if(arr[i][mid-1]!=x){
                f=mid;
                break;
            }
            else high = mid -1;
        }
        if(arr[i][mid]<x)   low = mid + 1;
        if(arr[i][mid]>x)   high = mid - 1;
        }
```



```cpp
        }
        if(f==-1) count = 0;
    else count = n-f;
    if(maxcount<count){
        maxcount = count;
        row = i;
    }
    }
    cout<<row<<" "<<maxcount;
}
```

4. Given an array of integers nums containing n + 1 integers where each integer is in the range [1, n] inclusive in sorted order.
There is only one repeated number in nums, return this repeated number.



binary search > asssignment4.cpp > ...

```cpp
#include<iostream>
using namespace std;
int main(){
    int arr[]= {1,2,3,4,5,6,6,7,};
    int n = sizeof(arr)/sizeof(arr[0]);
    int low = 0;
    int high = n-1;
    while(low<=high){
        int mid = low + (high  - low)/2;
        if(arr[mid]==mid+1)  low = mid+1;
        if(arr[mid]==mid){
            if(arr[mid]==arr[mid-1]){
                cout<<arr[mid];
                break;
            }
            else high = mid - 1;
        }
    }
}
```