2. Write a function:

int solution(vector<int> &A);

; that, given an array A of N integers, returns the smallest positive integer (greater than 0) that does not

occur in A. For example, given A = [1, 3, 6, 4, 1, 2], the function should return 5. Given A = [1, 2, 3], the

function should return 4. Given A = [−1, −3], the function should return 1. Write an efficient algorithm for

the following assumptions:

• N is an integer within the range [1..100,000] each element of array A is an integer within the

range [−1,000,000..1,000,000].

Solution:

int solution(vector<int>& A) {

if(A.size() == 0)

return 1;

std::set<int> numset;

for(int i=0; i<A.size(); i++)

{

numset.insert(A[i]);

}

if( \*(numset.begin()) > 1 )

return 1;

if( \*(--numset.end()) < 1 )

return 1;

if( \*(--numset.end()) == 1 )

return 2;

if(numset.size() == 1)

{

auto it1 = numset.begin();

if(\*it1 >1 || \*it1 < 0)

return 1;

else

return (\*it1)+1;

}

auto it1 = numset.begin();

auto it2 = ++(numset.begin());

while( it2!=numset.end())

{

if((\*it2-\*it1) == 1)

{

it1++;

it2++;

}else

{

if( \*it1 <= 0 && \*it2 <= 0)

{

it1++;

it2++;

}

if( \*it2 > 1) /\*This condition can be combined with previous if block\*/

{

if(\*it1 <= 0)

return 1;

else

return \*it1+1;

}

if(\*it2 == 1)

{

it1++;

it2++;

}

}

}

it2--;

if(\*it2 <= 0)

return 1;

else

return \*(it2)+1;

}