DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
Discover. Learn. Empower.

# Experiment-4

**Student Name: Anshuman Singh** **UID: 20BCS2665**
**Branch: CSE** **Section/Group: 902/A**
**Semester: 6th** **Date of Performance: 02-03-2023**
**Subject Name: Competitive Coding II** **Subject Code: 20CSP-351**

**Aim: To demonstrate the concept of Hashing.**

**Problem1: Missing Number**

Given an array nums containing n distinct numbers in the range [0, n], return the only number in the range that is missing from the array.

**Example 1:**
Input: nums = [3,0,1]
Output: 2
Explanation: n = 3 since there are 3 numbers, so all numbers are in the range [0,3]. 2 is the missing number in the range since it does not appear in nums.

**Example 2:**
Input: nums = [0,1]
Output: 2
Explanation: n = 2 since there are 2 numbers, so all numbers are in the range [0,2]. 2 is the missing number in the range since it does not appear in nums.

**Example 3:**
Input: nums = [9,6,4,2,3,5,7,0,1]
Output: 8
**Explanation:** n = 9 since there are 9 numbers, so all numbers are in the range [0,9]. 8 is the missing number in the range since it does not appear in nums.
Constraints: n ==
nums.length 1
<= n <= 104
0 <= nums[i] <= n
All the numbers of nums are unique.

Follow up: Could you implement a solution using only O(1) extra space complexity and O(n) runtime complexity?

## Code:-

```
import java.util.HashMap;
class Solution {
public int missingNumber(int[] nums) {
HashMap <Integer,Integer> numMap=new HashMap<Integer,Integer>();
for(int i=0;i<nums.length;i++){
numMap.put(nums[i],i);
}
for(int i=0;i<=nums.length;i++){
if(numMap.containsKey(i)){continue;}
else return i;
}
return 0;
}
}
```

## Output:-

**Probem2:** Longest Substring Without Repeating Characters

Given a string s, find the length of the longest substring without repeating characters.

**Example 1:**
Input: s = "abcabcbb"
Output: 3
Explanation: The answer is "abc", with the length of 3.

**Example 2:**
Input: s = "bbbbb"
Output: 1
Explanation: The answer is "b", with the length of 1.

**Example 3:**
Input: s = "pwwkew"
Output: 3
Explanation: The answer is "wke", with the length of 3.
Notice that the answer must be a substring, "pwke" is a subsequence and not a substring.
Constraints:
0 <= s.length <= 5 * 104 s consists of English letters,
digits, symbols and spaces.

## Code:-

```
class Solution {
public int lengthOfLongestSubstring(String s) {
int left = 0; int right = 0; int result = 0;
Set<Character> chars = new HashSet<>();
while(right< s.length()){
char c = s.charAt(right);
if(!chars.contains(c)){
chars.add(c); right++;
result = Math.max(result, right - left);
}
else{

chars.remove(s.charAt(left)); left++;
}
```

```
}
return result;
} }
```

**Output:-**