

# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## Worksheet 4

**Student Name:** Palak

**UID:** 22CS12960

**Branch:** CSE

**Section/Group:** 605-B

**Semester:** 5

**Date of Performance:** 05/03/25

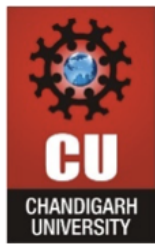
**Subject Name:** AP

**Subject Code:** 22CSP-351

1. A string  $s$  is nice if, for every letter of the alphabet that  $s$  contains, it appears both in uppercase and lowercase. For example, "abABB" is nice because 'A' and 'a' appear, and 'B' and 'b' appear. However, "abA" is not because 'b' appears, but 'B' does not.

Given a string  $s$ , return the longest substring of  $s$  that is nice. If there are multiple, return the substring of the earliest occurrence. If there are none, return an empty string.

```
class Solution {  
  
public:  
  
    bool isNice(const string& str) {  
  
        unordered_set<char> charSet(str.begin(), str.end());  
  
        for (char ch : str) {  
  
            if (charSet.count(tolower(ch)) == 0 || charSet.count(toupper(ch)) == 0) {  
  
                return false;  
  
            }  
  
        }  
  
        return true;  
  
    }  
  
    string longestNiceSubstring(string s) {
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

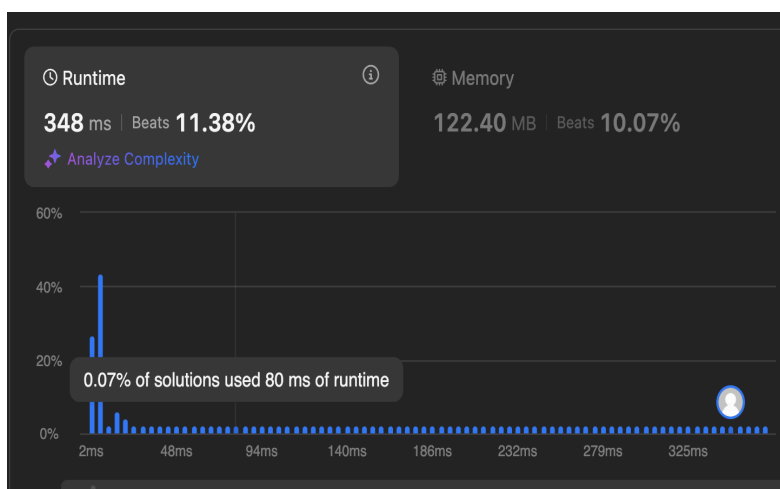
Discover. Learn. Empower.

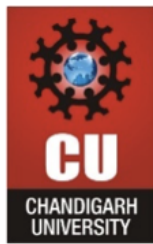
```
int maxLength = 0;

string result = "";

for (int i = 0; i < s.length(); ++i) {
    for (int j = i + 1; j <= s.length(); ++j) {
        string substring = s.substr(i, j - i);
        if (isNice(substring) && substring.length() > maxLength) {
            maxLength = substring.length();
            result = substring;
        }
    }
}

return result;
};
```





# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

## 2. Reverse bits of a given 32 bits unsigned integer.

Note:

Note that in some languages, such as Java, there is no unsigned integer type. In this case, both input and output will be given as a signed integer type. They should not affect your implementation, as the integer's internal binary representation is the same, whether it is signed or unsigned.

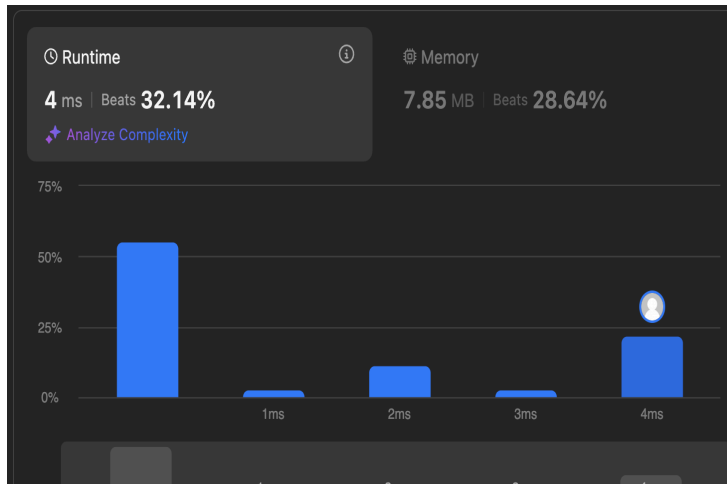
In Java, the compiler represents the signed integers using 2's complement notation. Therefore, in Example 2 above, the input represents the signed integer -3 and the output represents the signed integer -1073741825.

```
class Solution {  
  
    public:  
  
        uint32_t reverseBits(uint32_t n) {  
  
            uint32_t result = 0;  
  
            for (int i = 0; i < 32; ++i) {  
  
                result = (result << 1) | (n & 1);  
  
                n >>= 1;  
  
            }  
  
            return result;  
  
        }  
  
};
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

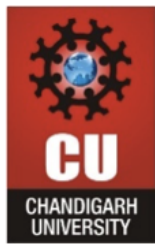
Discover. Learn. Empower.



3.

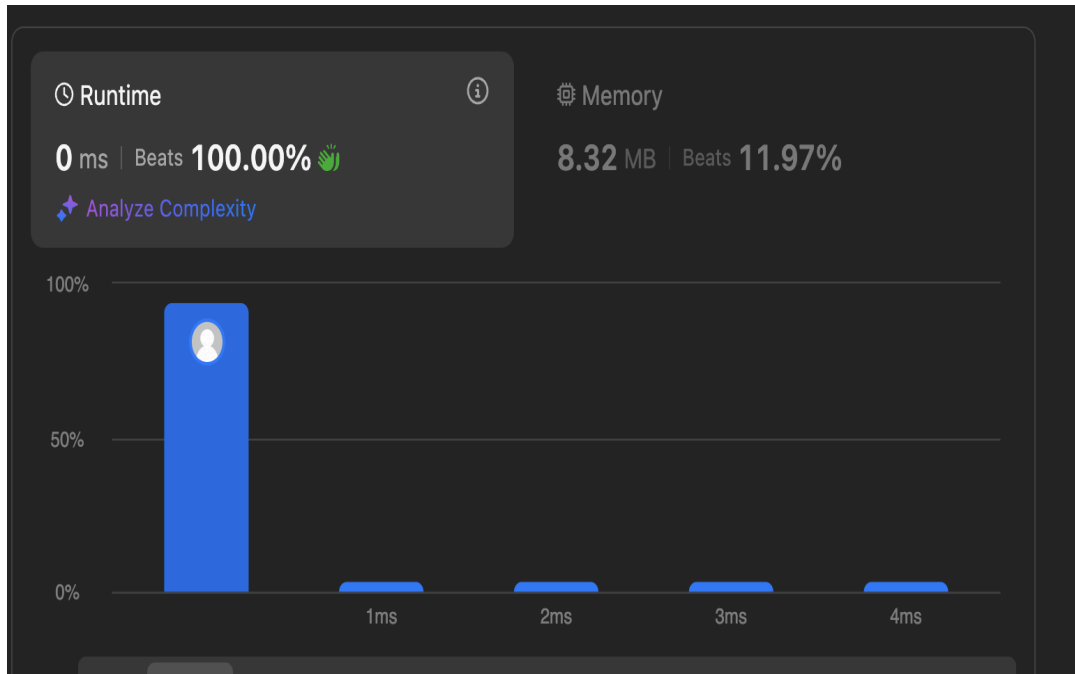
Given a positive integer  $n$ , write a function that returns the number of set bits in its binary representation (also known as the Hamming weight)..

```
class Solution {  
  
public:  
  
    int hammingWeight(uint32_t n) {  
  
        int count = 0;  
  
        while (n != 0) {  
  
            count += (n & 1);  
  
            n >>= 1;        }  
  
        return count;  
  
    }  
  
};
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.



4. Given an integer array `nums`, find the subarray with the largest sum, and return its sum.

```
class Solution {
```

```
public:
```

```
    int maxSubArray(vector<int>& nums) {
```

```
        int maxSum = nums[0];
```

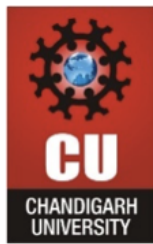
```
        int currentSum = nums[0];
```

```
        for (int i = 1; i < nums.size(); ++i) {
```

```
            currentSum = max(nums[i], currentSum + nums[i]);
```

```
            maxSum = max(maxSum, currentSum);
```

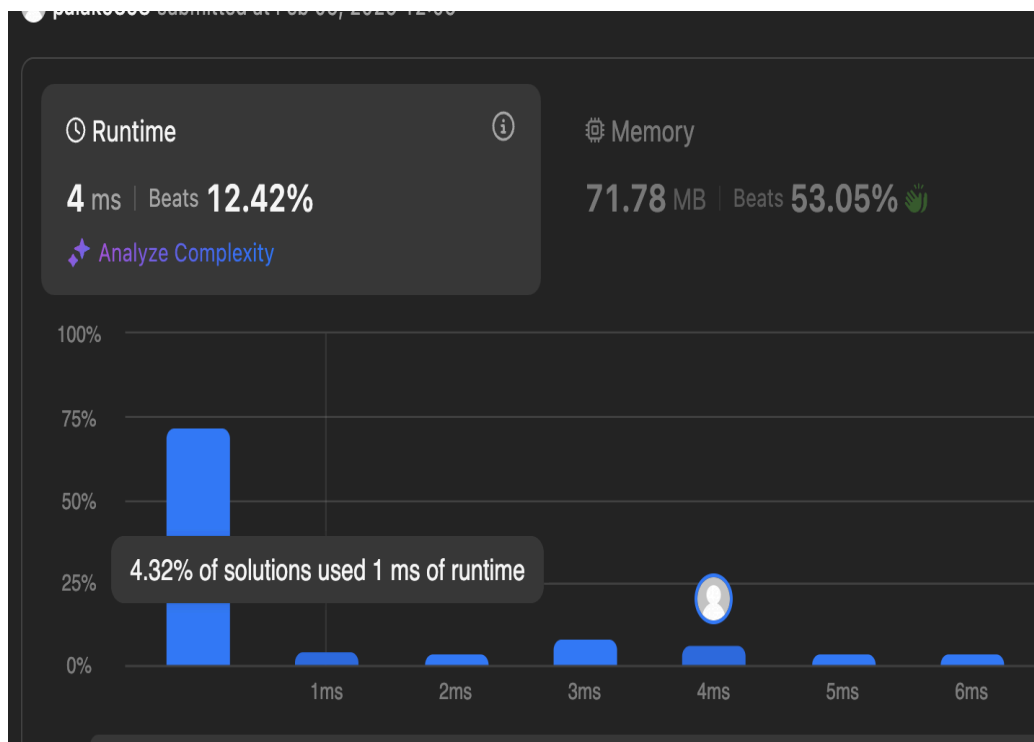
```
        }
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
return maxSum;  
  
}  
  
};
```



5. Write an efficient algorithm that searches for a value target in an  $m \times n$  integer matrix matrix. This matrix has the following properties:

Integers in each row are sorted in ascending from left to right.

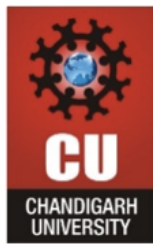
Integers in each column are sorted in ascending from top to bottom.

```
class Solution {
```

```
public:
```

```
bool searchMatrix(vector<vector<int>>& matrix, int target) {
```

```
if (matrix.empty() || matrix[0].empty()) return false;
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
int rows = matrix.size();

int cols = matrix[0].size();

int row = 0, col = cols - 1; // Start from the top-right corner

while (row < rows && col >= 0) {

    if (matrix[row][col] == target) {

        return true; // Target found

    } else if (matrix[row][col] > target) {

        col--; // Move left

    } else {

        row++; // Move down

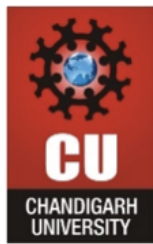
    }

}

return false; // Target not found

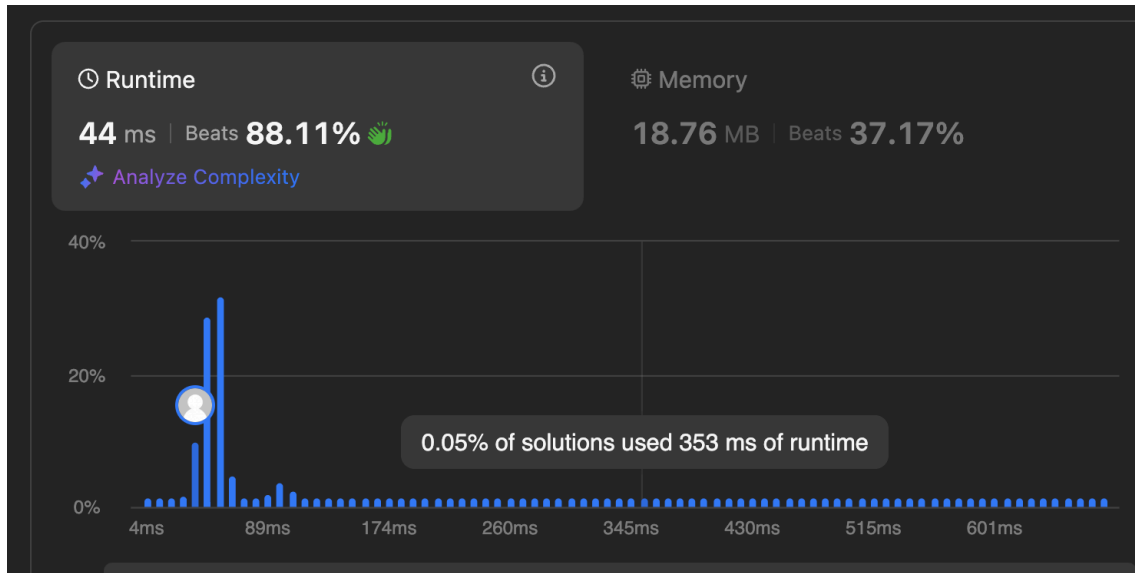
}
```

```
};
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.



6. Your task is to calculate  $ab \bmod 1337$  where  $a$  is a positive integer and  $b$  is an extremely large positive integer given in the form of an array.

```
const int MOD = 1337;
```

```
class Solution {
```

```
public:
```

```
    int modPow(int a, int b, int mod) {
```

```
        int result = 1;
```

```
        a %= mod;
```

```
        while (b > 0) {
```

```
            if (b % 2 == 1) {
```

```
                result = (result * a) % mod;
```

```
            }
```

```
            a = (a * a) % mod;
```

```
            b /= 2;
```

```
        }
```





# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
        return result;
    }

    int superPow(int a, vector<int>& b) {
        if (b.empty()) return 1;

        int lastDigit = b.back();
        b.pop_back();

        int part1 = modPow(superPow(a, b), 10, MOD);
        int part2 = modPow(a, lastDigit, MOD);

        return (part1 * part2) % MOD;
    }
};
```



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

