Easy 1

Given a string s consisting of words and spaces, return the length of the last word in the string.
A word is a maximal substring consisting of non-space characters only.

Solution:

```
#include <stdio.h>
#include <string.h>

int lengthOfLastWord(const char *s) {
    int count = 0;
    int end = strlen(s) - 1;
    while (end >= 0 && s[end] == ' ') {
        end--;
    }
    for (int i = end; i >= 0; i--) {
        if (s[i] == ' ') {
            break;
        }
        count++;
    }
    return count;

}

int main() {
    char input[1000];


    printf("Enter a string: ");
    fgets(input, sizeof(input), stdin);


    input[strcspn(input, "\n")] = '\0';

    int result = lengthOfLastWord(input);


    printf("Length of the last word: %d\n", result);

    return 0;
}
```

Medium 2

Given an integer array of size n, find all elements that appear more than ⌊ n/3 ⌋ times.

Solution:

```
#include <stdio.h>
void findMajority(int arr[], int n) {
   int flag = 0;
   for (int i = 0; i < n; i++) {
      int count = 0;
      for (int j = i; j < n; j++) {
         if (arr[i] == arr[j]) {
            count++;
         }
      }
      if (count > (n / 3)) {
         printf("%d ", arr[i]);
         flag = 1;
      }
   }
   if (!flag)
      printf("No Majority Element\n");
}

int main() {
   int n;


   printf("Enter the size of the array: ");
   scanf("%d", &n);

   int arr[n];


   printf("Enter the elements of the array:\n");
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }


   findMajority(arr, n);

   return 0;
}
```

Hard 3

Given an integer n, count the total number of digit 1 appearing in all non-negative integers less than or equal to n.

Solution:

```c
#include <stdio.h>

int countDigitOne(int n) {
    int count = 0;
    long long factor = 1;

    while (n / factor > 0) {
        int currentDigit = (n / factor) % 10;
        int higherDigits = n / (factor * 10);
        int lowerDigits = n % factor;

        if (currentDigit == 0) {
            count += higherDigits * factor;
        } else if (currentDigit == 1) {
            count += higherDigits * factor + (lowerDigits + 1);
        } else {
            count += (higherDigits + 1) * factor;
        }

        factor *= 10;
    }

    return count;
}

int main() {
    int n;


    printf("Enter the value of n: ");
    scanf("%d", &n);


    int result = countDigitOne(n);
    printf("Total number of digit 1 from 1 to %d: %d\n", n, result);

    return 0;
}
```