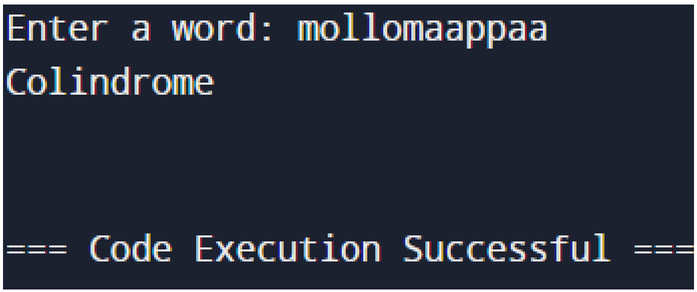
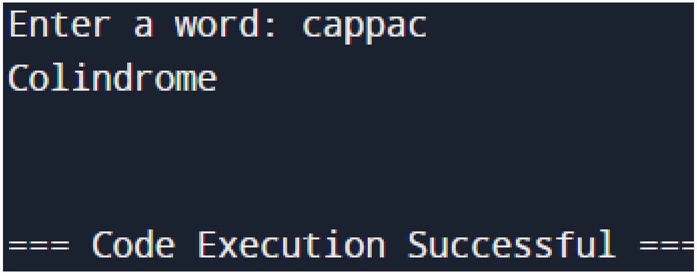
Programming Task

Q1) Write a Program in C or Java to check if a number is a Colindrome or not. Colidrome is a word that has 3 alphabets followed by the reverse of the 3 alphabets and so on

```
#include <stdio.h>

#include <string.h>


int main() {

  char word[10];

  printf("Enter a word: ");

  scanf("%s", word);


  int length = strlen(word);

  if (length % 6 != 0) {

    printf("Not a Colindrome\n");

    return 0;

  }

  for (int i = 0; i < length; i += 3) {

    if (word[i + 2] == word[i + 3] && word[i + 1] == word[i + 4] && word[i] == word[i + 5]) {

      printf("Colindrome\n");

      return 0;

    }

  }

  printf("Not a Colindrome\n");

  return 0;

}
```



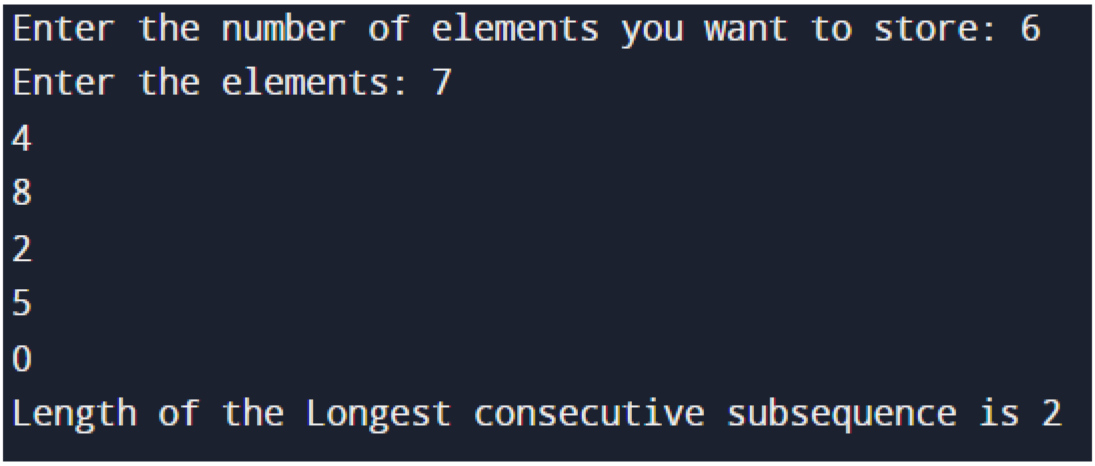
```
Enter a word: mollomaappaa
Colindrome


=== Code Execution Successful ===
```



```
Enter a word: cappac
Colindrome


=== Code Execution Successful ===
```

Q2) Longest Consecutive Subsequence

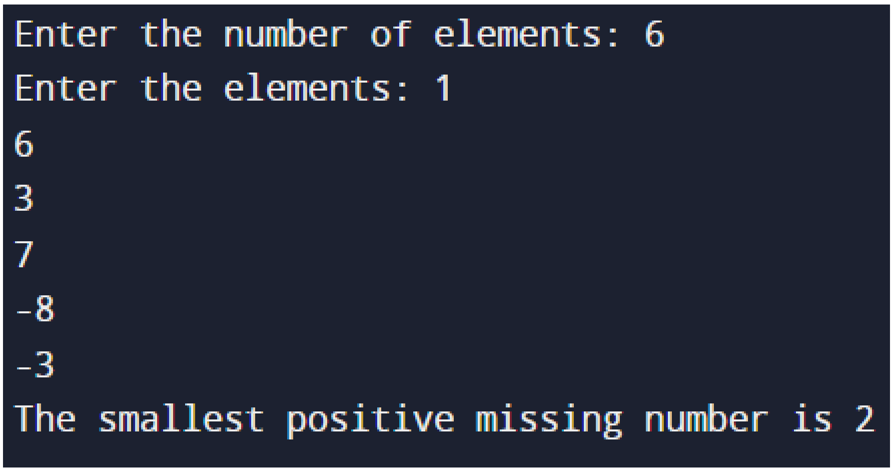
```
#include <stdio.h>
#include <stdlib.h>

int max(int a, int b) {
    return (a > b) ? a : b;
}
int LCS(int nums[], int n) {
    if (n == 0) return 0;
    int *hashSet = (int *)calloc(n, sizeof(int));
    for (int i = 0; i < n; i++) {
        hashSet[i] = 0;
    }
    for (int i = 0; i < n; i++) {
        hashSet[nums[i]] = 1;
    }
    int maxLen = 0;
    for (int i = 0; i < n; i++) {
        if (hashSet[nums[i] - 1] == 0) {
            int j = nums[i];
            while (hashSet[j] == 1) {
                j++;
            }
            maxLen = max(maxLen, j - nums[i]);
        }
    }
    free(hashSet);
    return maxLen;
}
int main() {
    int n;
    printf("Enter the number of elements you want to store: ");
    scanf("%d", &n);
    int *nums = (int *)malloc(n * sizeof(int));
    printf("Enter the elements: ");
    for (int i = 0; i < n; i++) {
        scanf("%d", &nums[i]);
    }
    printf("Length of the Longest consecutive subsequence is %d\n", LCS(nums, n));
    free(nums);
    return 0;
}
```



```
Enter the number of elements you want to store: 6
Enter the elements: 7
4
8
2
5
0
Length of the Longest consecutive subsequence is 2
```

Q3) Write a C program to find the smallest positive integer that is missing from an unsorted array of integers.

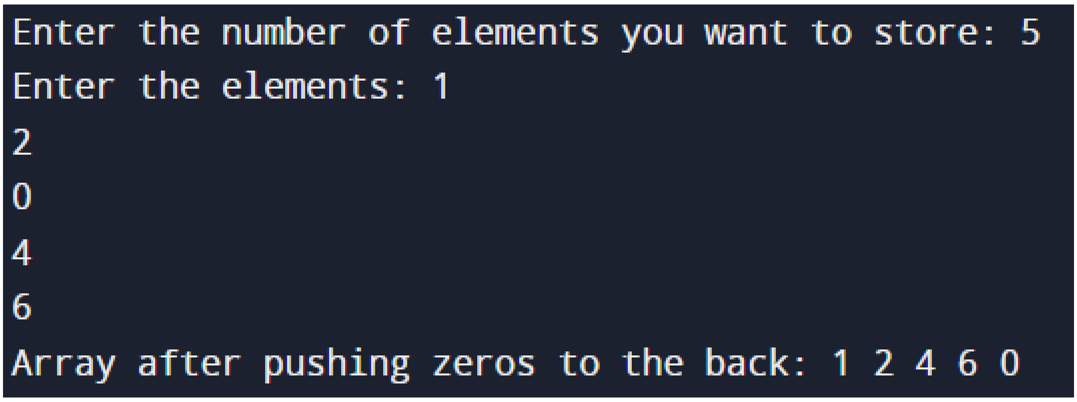
```
#include <stdio.h>


void swap(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}
int findSmallestMissingPositive(int arr[], int size) {
    for (int i = 0; i < size; i++) {
        while (arr[i] > 0 && arr[i] <= size && arr[arr[i] - 1] != arr[i]) {
            swap(&arr[i], &arr[arr[i] - 1]);
        }
    }
    for (int i = 0; i < size; i++) {
        if (arr[i] != i + 1) {
            return i + 1;
        }
    }
    return size + 1;
}
int main() {
    int size;
    printf("Enter the number of elements: ");
    scanf("%d", &size);
    int arr[size];
    printf("Enter the elements: ");
    for (int i = 0; i < size; i++) {
        scanf("%d", &arr[i]);
    }
    int missing = findSmallestMissingPositive(arr, size);
    printf("The smallest positive missing number is %d\n", missing);
    return 0;
}
```



```
Enter the number of elements: 6
Enter the elements: 1
6
3
7
-8
-3
The smallest positive missing number is 2
```

Q4) Write a C program to move all zeroes in an array to the end while maintaining the relative order of the non-zero elements.

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


void pushZero(int num[], int n) {
    int c = 0;
    for (int i = 0; i < n; i++) {
        if (num[i] != 0)
            num[c++] = num[i];
    }
    while (c < n) {
        num[c++] = 0;
    }
}
int main() {
    int n;
    printf("Enter the number of elements you want to store: ");
    scanf("%d", &n);
    int *num = (int *)malloc(n * sizeof(int));
    printf("Enter the elements: ");
    for (int i = 0; i < n; i++) {
        scanf("%d", &num[i]);
    }
    pushZero(num, n);
    printf("Array after pushing zeros to the back: ");
    for (int i = 0; i < n; i++) {
        printf("%d ", num[i]);
    }
    printf("\n");
    free(num);
    return 0;
}
```



```
Enter the number of elements you want to store: 5
Enter the elements: 1
2
0
4
6
Array after pushing zeros to the back: 1 2 4 6 0
```