LAB ASSIGNMENT- 5

Implement C programs for the following problem statements:

a) Palindromized Number: A string is said to be a palindrome if the order of arrangement of alphabet symbols read from both right to left ad left to right are one and the same. Similarly, let us claim that any number is said to be a palindromized number if the order of arrangement of digits from left to right and right to left are same.

Given a number ’x’ check whether it is a palidromized number or not.

Ex 2: x = 12321 output : YES

Ex 3: x = 12211 output : NO

Ex 4: x = 8 output : YES

```
#include <stdio.h>
int main() {
 int n, temprev= 0, r, original;
  printf("Enter an integer: ");
  scanf("%d", &n);
 original=n;

  // reversed integer is stored in another variable
  while (n != 0) {
    r = n % 10;
    temprev = temprev * 10 + r;
    n /= 10;
  }
  //checks if palindrome or not
   if (original == temprev)
       printf("Reverse of %d is %d , so it is a palindrome.",original, 
temprev);
   else
       printf("Reverse of %d is %d , so it is not a palindrome.",original, 
temprev);
   return 0;
}
```

Inference- n is the variable which initially stores the entered number, temprev is the variable storing reversed number, and original is the variable storing the entered number. The program checks if temprev is same as original to declare the entered number as palindrome. If not, then the entered number is declared as not palindrome.

b) Anagrams: Two strings are said to be Anagrams if the possess same number of individual alphabet symbols and they differ by order of arrangements. For example, the strings ‘dog’ and ‘god’ are Anagrams whereas ‘good’ and ‘dog’ are not Anagrams. Given two strings S1 and S2, check whether they are Anagram strings are not.

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

int main () {
  char s1[25], s2[25], tempstr;
  int i, j, n, m;


printf("Enter string 1:");
 scanf("%s", s1);


 printf("Enter string 2:");
 scanf("%s", s2);


 n  = strlen(s1);
 m = strlen(s2);


  // If both strings are of different length, then they are not
anagrams
  if( n != m) {
    printf("Strings are not anagrams \n");
    return 0;
  }


  for (i = 0; i < n-1; i++) {
    for (j = i+1; j < n; j++) {
      if (s1[i] > s1[j]) {
        tempstr  = s1[i];
        s1[i] = s1[j];
        s1[j] = tempstr;
```

```
          }
          if (s2[i] > s2[j]) {
            tempstr  = s2[i];
            s2[i] = s2[j];
            s2[j] = tempstr;
          }
        }
      }

      // Compare both strings character by character
      for(i = 0; i<n; i++) {
        if(s1[i] != s2[i]) {
          printf("Strings are not anagrams \n");
          return 0;
        }
      }
      printf("Strings are anagrams \n");
      return 0;
    }
```

Inference- two strings are taken as input and then they are checked for string length. If both have same length then the program further checks by arranging the characters of the string in ascending order and checking each character of the string. This is how the program checks if the strings are anagrams or not.