LAB ASSIGNMENT 13

1. Write a C program for printing happy numbers up to 50.

```
#include <stdio.h>

//isHappyNumber() will determine whether a number is happy or not
int isHappyNumber(int num){
   int rem = 0, sum = 0;

   //Calculates the sum of squares of digits
   while(num > 0){
      rem = num%10;
      sum = sum + (rem*rem);
      num = num/10;
   }
   return sum;
}

int main()
{
   //Displays all happy numbers between 1 and 100
   printf("List of happy numbers between 1 and 100: \n");
   for(int i = 1; i <= 100; i++){
      int result = i;
```

```
        //Happy number always ends with 1 and
        //unhappy number ends in a cycle of repeating numbers which contains 4
        while(result != 1 && result != 4){
            result = isHappyNumber(result);
        }

        if(result == 1)
            printf("%d ", i);
    }

    return 0;
}
```

2. Write a C program for generating Fibonacci-like sequence using Keith Number format.

Input : x = 197

Output : Yes

197 has 3 digits, so n = 3

The number is Keith because it appears in the special sequence that has first three terms as 1, 9, 7 and remaining terms evaluated using sum of previous 3 terms.

1, 9, 7, 17, 33, 57, 107, 197,

```
#include <stdio.h>

int main() {
int i, k=3, n;

printf("Enter number of terms: ");
scanf("%d", &n);

int keith[25];
keith[0]=1;
keith[1]=9;
keith[2]=7;

printf("The Keith series is:\n");
printf("%d, %d, %d",keith[0],keith[1],keith[2]);

for (i=3; i<n; i++)
{
keith[i]= keith[i-1] + keith[i-2]+ keith[i-3];
printf(", %d", keith[i]);
}

return 0;
}
```

3.Write a C program to print prime triplet using functions.

Note: prime triplet in the form of (p, p + 2, p + 6) or (p, p + 4, p + 6).

Output: (5, 7, 11), (7, 11, 13), (11, 13, 17), (13, 17, 19), (17, 19, 23), (37, 41, 43), (41,43, 47), (67, 71, 73)

```
#include <stdio.h>

int primetrip(int num){
   int flag=0, i;

   for(i=2;i<num;i++){
      if(num%i==0){
         flag=1;
         break;
      }
   }

   if(flag==0)
   return 1;

   else
   return 0;
}

int main()
```

```
{
   int n, N, i, c=0;

   printf("Enter starting number: ");
   scanf("%d", &n);

   printf("Enter ending number: ");
   scanf("%d", &N);

   if(n==1){
      n=n+1;
   }

   for(i=n;i<=N;i++){
      if(primetrip(i)==1 && primetrip(i+2)==1 && primetrip(i+6)==1){
         c++;
         printf("%d, %d, %d\n",i,i+2,i+6);
      }

      else if(primetrip(i)==1 && primetrip(i+4)==1 &&
primetrip(i+6)==1){
         c++;
         printf("%d, %d, %d\n",i,i+4,i+6);
      }
```

```
    }

    printf("%d is the total number of prime triplets set",c);

    return 0;
}
```

4.Write a C program to Check whether two strings are anagram of each other.

Ex: LISTEN- SILENT

```
#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;
}
```

5.Write a C program to find minimum occurring character in a string.

```
#include <stdio.h>
#define MAX_SIZE 100  // Maximum string size
#define MAX_CHARS 255 // Maximum characters allowed

int main()
{
    char str[MAX_SIZE];
    int freq[MAX_CHARS]; //Stores frequency of each character
    int i = 0, min;
    int ascii;
    printf("Enter any string: ");
    scanf("%s",str);

    /* Initialize frequency of all characters to 0 */
    for(i=0; i<MAX_CHARS; i++)
    {
        freq[i] = 0;
    }

    /* Finds frequency of each characters */
    i=0;
    while(str[i] != '\0')
    {
        ascii = (int)str[i];
        freq[ascii] += 1;

        i++;
    }

    /* Finds minimum frequency */
    min = 0;
    for(i=0; i<MAX_CHARS; i++)
    {
        if(freq[i] != 0)
        {
          if(freq[min] == 0 || freq[i] < freq[min])
            min = i;
        }
    }



    printf("Minimum occurring character is '%c' = %d.", min,
  freq[min]);


    return 0;
  }
```

6.Write a C program to delete all duplicate elements from an array using functions.

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

void dupremove(int len, char str[]){
    for(int i=0; i<len; i++)
  {
     char ch = str[i];

    //Check if current character matches any other character in
the subsequence
    for(int j=i+1; j<len; ){

      if(str[i] == str[j]){
        //If yes, then shift the right characters to the left
        for(int k=j; k<len; k++){
          str[k] = str[k+1];
        }
        len--;

      } else {
          //only increment if the duplicate is not found
          //because after the shift, index j can again have duplicate
          j++;
      }
    }
  }
  printf("%s",str);
}

int main()
{
  char string[30];
  int length= strlen(string);

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

  printf("String after removing all duplicates is ");
  dupremove(length, string);

  return 0;
}
```