

LAB ASSIGNMENT 13

1.

```
main.c
14 }
15
16 int main()
17 {
    List of happy numbers between 1 and 100:
    1 7 10 13 19 23 28 31 32 44 49 68 70 79 82 86 91 94 97 100
    ...Program finished with exit code 0
    Press ENTER to exit console.
```

2.

```
main.c
4 int i, k=0, n;
5
6 printf("Enter number of terms: ");
7 scanf("%d", &n);
    Enter number of terms: 10
    The Keith series is:
    1, 9, 7, 17, 33, 57, 107, 197, 361, 665
    ...Program finished with exit code 0
    Press ENTER to exit console.
```

3.

The screenshot shows a C program in a code editor. The code defines a function `primetrip` that takes an integer `num` and returns an integer. The program prompts the user to enter a starting and ending number, then lists all prime triplets within that range. The output shows triplets (1, 3, 7), (5, 7, 11), (7, 11, 13), (11, 13, 17), (13, 17, 19), (17, 19, 23), (37, 41, 43), (41, 43, 47), and (67, 71, 73). It concludes that there are 9 prime triplets in the set and finishes with exit code 0.

```
main.c
1 #include <stdio.h>
2
3 int primetrip(int num){
4     int flag = 0;
5     for (int i = 1; i <= num; i++)
6     {
7         if (i % 2 == 0)
8             continue;
9         for (int j = i + 2; j <= num; j++)
10         {
11             if (j % 2 == 0)
12                 continue;
13             for (int k = j + 2; k <= num; k++)
14             {
15                 if (k % 2 == 0)
16                     continue;
17                 if (i + j == k)
18                 {
19                     printf("%d, %d, %d\n", i, j, k);
20                     flag++;
21                 }
22             }
23         }
24     }
25     return flag;
26 }
```

Enter starting number: 0
Enter ending number: 70
1, 3, 7
5, 7, 11
7, 11, 13
11, 13, 17
13, 17, 19
17, 19, 23
37, 41, 43
41, 43, 47
67, 71, 73
9 is the total number of prime triplets set
...Program finished with exit code 0
Press ENTER to exit console.

4.

The screenshot shows a C program in a code editor. The code defines a `main` function that takes two strings as input and checks if they are anagrams by comparing their sorted versions. The output shows the strings "god" and "dog" and the message "Strings are anagrams". The program finishes with exit code 0.

```
main.c
2 #include <string.h>
3
4 int main () {
5     char s1[25], s2[25], tempstr[25];
6     int i, j, k;
7     printf("Enter string 1:\n");
8     scanf("%s", s1);
9     printf("Enter string 2:\n");
10    scanf("%s", s2);
11    for (i = 0; s1[i] != '\0'; i++)
12        tempstr[i] = s1[i];
13    for (j = 0; s2[j] != '\0'; j++)
14        tempstr[j + i] = s2[j];
15    tempstr[j + i] = '\0';
16    qsort(s1, i, sizeof(char), cmp);
17    qsort(tempstr, j + i, sizeof(char), cmp);
18    for (k = 0; k < i + j; k++)
19        if (s1[k] != tempstr[k])
20            return 1;
21    printf("Strings are anagrams\n");
22    return 0;
23 }
```

Enter string 1:god
Enter string 2:dog
Strings are anagrams
...Program finished with exit code 0
Press ENTER to exit console.

5.

The screenshot shows a code editor with the following C code in `main.c`:

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

Below the code is a terminal window with the following output:

```
input
Enter any string: fullerene
Minimum occurring character is 'f' = 1.
...Program finished with exit code 0
Press ENTER to exit console.
```

6.

The screenshot shows a code editor with the following C code in `main.c`:

```
1 #include <stdio.h>
2 #include <string.h>
3 void remove_duplicates(int len, char str[]) {
```

Below the code is a terminal window with the following output:

```
Enter string: abebbb
String after removing all duplicates is abe
...Program finished with exit code 0
Press ENTER to exit console.
```