Iterative Control Statements (Part - 2)

1. Write a program to find the Nth term of the Fibonnaci series.

```c
#include<stdio.h>
int main()
{
    int n, i, a = -1, b = 1, s = 0;
    printf("Enter a number : ");
    scanf("%d", &n);
    for (i = 1; i <= n; i++)
    {
        s = a + b;
        a = b;
        b = s;
    }
      printf("%dth term of the Fibonnaci series is %d", n, s);
    return 0;
}
```

```
Output:
Enter a number : 13
13th term of the Fibonnaci series is 144
```

2. Write a program to print first N terms of Fibonacci series

```c
#include<stdio.h>
int main()
{
    int n, i, a = -1, b = 1, s = 0;
    printf("Enter a number : ");
    scanf("%d", &n);
    for (i = 1; i <= n; i++)
    {
        s = a + b;
        printf("%d ", s);
        a = b;
        b = s;
    }
    return 0;
}
```

```
Output:
Enter a number : 11
0 1 1 2 3 5 8 13 21 34 55
```

3. Write a program to check whether a given number is there in the Fibonacci series or not.

```c
#include<stdio.h>
int main()
{
    int n, i, a = -1, b = 1, s = 0;
    printf("Enter a number : ");
    scanf("%d", &n);
    for (i = 1; i <= n + 2; i++)
    {
        s = a + b;
        if (s == n)
        {
            break;
        }
        a = b;
        b = s;
    }
```

```
    s == n ? printf("Given number is in the Fibonnaci series") :
printf("Given number is not in the Fibonnaci series");
    return 0;
}
================================================================================
Output:
Enter a number : 56
Given number is not in the Fibonnaci series
```

4. Write a program to calculate HCF of two numbers

```
#include <stdio.h>
int main()
{
    int a, b, i;
    printf("Enter two numbers : ");
    scanf("%d%d", &a, &b);
    for (i = a < b ? a : b; i >= 1; i--)
    {
        if (a % i == 0 && b % i == 0)
        {
            printf("HCF of %d and %d is %d", a, b, i);
            break;
        }
    }
    return 0;
}
================================================================================
Enter two numbers : 15 25
HCF of 15 and 25 is 5
```

5. Write a program to check whether two given numbers are co-prime numbers or not

```
#include <stdio.h>
int main()
{
    int a, b, i;
    printf("Enter two numbers : ");
    scanf("%d%d", &a, &b);
    for (i = a < b ? a : b; i >= 1; i--)
    {
        if (a % i == 0 && b % i == 0)
        {
            break;
        }
    }
    i == 1 ? printf("%d and %d are co-prime", a, b) : printf("%d and %d are
not co-prime", a, b);
    return 0;
}
================================================================================
Output:
Enter two numbers : 87 88
87 and 88 are co-prime
```

6. Write a program to print all Prime numbers under 100

```
#include <stdio.h>
int main()
{
    int i, j;
    for (i = 2; i <= 100; i++)
    {
        for (j = 2; j < i; j++)
        {
```

```
            if (i % j == 0)
            {
                break;
            }
        } // end inner loop
        if (i == j)
        {
            printf("%d ", i);
        }
    } // end outer loop
    return 0;
}
```

```
Output:
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97
```

7. Write a program to print all Prime numbers between two given numbers

```
#include <stdio.h>
int main()
{
    int a, b, i, j;
    printf("Enter two numbers : ");
    scanf("%d%d", &a, &b);
    for (i = a + 1; i <= b - 1; i++)
    {
        for (j = 2; j < i; j++)
        {
            if (i % j == 0)
            {
                break;
            }
        } // end inner loop
        if (i == j)
        {
            printf("%d ", i);
        }
    } // end outer loop
    return 0;
}
```

```
Output:
Enter two numbers : 20 30
23 29
```

8. Write a program to find next Prime number of a given number

```
#include <stdio.h>
int main()
{
    int a, i, j;
    printf("Enter a numbers : ");
    scanf("%d", &a);
    for (i = a+1; 1 ; i++)
    {
        for (j = 2; j < i; j++)
        {
            if (i % j == 0)
            {
                break;
            }
        } // end inner loop
        if (i == j)
        {
            printf("Next prime number is %d ", i);
```

```
            break;
        }
    } // end outer loop
    return 0;
}
================================================================================
Output:
Enter a numbers : 19
Next prime number is 23
```

9. Write a program to check whether a given number is an Armstrong number or not

```
#include <stdio.h>
#include <math.h>
int main()
{
    int n, i, r, s = 0, temp, j;
    printf("Enter a number : ");
    scanf("%d", &n);
    temp = n;

    // below for loop is for counting digit
    for (i = 1; n; i++)
    {
        r = n % 10;
        n = n / 10;
    }

    i = i - 1; // digit stored in i variable
    n = temp;  // copy value of temp variable to n variable

    // below loop is for calculation
    for (j = 1; n; j++)
    {
        r = n % 10;
        n = n / 10;
        s = s + pow(r, i);
    }
    temp == s ? printf("Armstrong number") : printf("Not armstrong number");
    return 0;
}
================================================================================
Output:
Enter a number : 1634
Armstrong number
```

10. Write a program to print all Armstrong numbers under 1000

```
#include <stdio.h>
#include <math.h>
int main()
{
    int n = 0, i, r, s, temp, j;
    printf("Armstrong numbers under 1000\n");
    while (n <= 1000)
    {
        temp = n;
        s = 0;
        // below for loop is for counting digit
        for (i = 1; n; i++)
        {
            r = n % 10;
            n = n / 10;
        }          // end first for loop
        i = i - 1; // digit stored in i variable
```

```
        n = temp;  // copy value of temp variable to n variable

       // below loop is for calculation
        for (j = 1; n; j++)
        {
            r = n % 10;
            n = n / 10;
            s +=  pow(r, i); // s = s + pow(r, i)
        } // end second for loop
        if (temp == s)
            printf("%d ", s);
        n = temp;
        n++;
    } // end while loop
    return 0;
}
```

```
================================================================================
Output:
Armstrong numbers under 1000
0 1 2 3 4 5 6 7 8 9 153 370 371 407
```