Assignment 1

Q1. WAP to check whether a given is Armstrong or not.

```
#include <stdio.h>

int main() {
   int num, originalNum, remainder, result = 0;

   printf("Enter a three-digit integer: ");
   scanf("%d", &num);

   originalNum = num;

   while (originalNum != 0) {
      // remainder contains the last digit
      remainder = originalNum % 10;
      // result = remainder * remainder * remainder;
      result += remainder * remainder * remainder;
      // removing last digit from the original number
      originalNum /= 10;
   }

   if (result == num)
      printf("%d is an Armstrong number.", num);
   else
```

```
    printf("%d is not an Armstrong number.", num);

  return 0;
}
```

Q2. WAP to read two integers and print their HCF (Highest Common Factor).

```
#include <stdio.h>

int main() {
  int num1, num2, i, hcf;

  printf("Enter two integers: ");
  scanf("%d %d", &num1, &num2);

  for (i = 1; i <= num1 && i <= num2; ++i) {
    // Checks if i is factor of both integers
    if (num1 % i == 0 && num2 % i == 0) {
      hcf = i;
    }
  }

  printf("HCF of %d and %d is %d", num1, num2, hcf);

  return 0;
}
```

Q3. WAP to subtract two integers without using Minus (-) operator. (Hint Bitwise operator).

```
#include <stdio.h>

int main() {
   int a, b;

   a = 5;
   b = 3;

   printf("Bitwise NOT: %d %d\n", ~a, ~b);

   return 0;
}
```

Q4. WAP to accept two integer numbers and swap them using 4 different methods in C
language.

```
#include <stdio.h>

int main() {
   int a, b;

   printf("Enter the two numbers you want to swap: ");
```

```
    scanf("%d %d", &a, &b);

    printf("a is %d and b is %d\n", a, b);

    a = a +b;
    b = a - b;
    a = a-b;

    printf("After swapping a is %d and b is %d", a, b);

    return 0;
}
#include <stdio.h>

int main() {
    int a, b, c;

    printf("Enter the two numbers you want to swap: ");
    scanf("%d %d", &a, &b);

    printf("a is %d and b is %d\n", a, b);

    c = a;
    a = b;
    b = c;
```

```
    printf("After swapping a is %d and b is %d", a, b);


    return 0;
}
```

Q5. WAP to check whether number is Perfect Number or not.
(To check perfect number, we have to find all divisors of that number and find their sum, if
sum of divisors is equal to number it means number is Perfect Number.

```
#include <stdio.h>

int main() {
    int num, i, sum = 0;

    printf("Enter a number: ");
    scanf("%d", &num);

    for (i = 1; i < num; i++) {
        if (num % i == 0) {
            sum += i;
        }
    }

    if (sum == num) {
        printf("%d is a perfect number", num);
```

```
  } else {
    printf("%d is not a perfect number", num);
  }

  return 0;
}
```

Q6. WAP to accept a coordinate point in an XY coordinate system and determine in which
quadrant the coordinate point lies

Test Data: 7 9
Expected Output: The coordinate point (7,9) lies in the First quadrant.

```
#include <stdio.h>

int main() {
  int x, y;

  printf("Enter the points in a co-ordinate system: ");
  scanf("%d %d", &x, &y);

  if (x > 0 && y > 0) {
    printf("Point lies in first quadrant");
  } else if (x < 0 && y > 0) {
    printf("Point lies in second quadrant");
  } else if (x < 0 && y < 0) {
```

```
    printf("Point lies in third quadrant");
  } else {
    printf("Point lies in fourth quadrant");
  }

  return 0;
}
```

Q7. WAP for Binary to Decimal conversion & Decimal to Binary for a given number as per
user's choice

Q8. WAP to print below mentioned pattern.

1
01
101
0101
10101

```
#include <stdio.h>

int main() {
  for (int i = 1; i <= 5; i++) {
    for (int j = 1; j <= i; j++) {
      if ((i + j) % 2 == 0) {
        printf("1");
      } else {
```

```
            printf("0");
          }
        }
        printf("\n");
    }

    return 0;
}
```

Q9. WAP to print following Pyramid:

```
0 0
01 01
010 010
0101 0101
0101001010
```

Q10. WAP to print Pascal's Triangle.

```
#include <stdio.h>

int main() {
    int rows, space, i, j, coef = 1;

    printf("Enter the number of rows: ");
    scanf("%d", &rows);

    for (i = 0; i < rows; i++) {
```

```
      for (space = 1; space <= rows - i; space++) {
         printf(" ");
      }
      for (j = 0; j <= i; j++) {
         if (j == 0 || i == 0) {
            coef = 1;
         } else {
            coef = coef * (i - j + 1) / j;
         }
         printf("%4d", coef);
      }
      printf("\n");
   }

   return 0;
}
```