1. Hello, World

```
#include <stdio.h>

int main() {

  // printf() displays the string inside quotation

printf("Hello, World!");

return 0;

}
```

```
Hello, World!
```

2.C Program to Print an Integer

```
#include <stdio.h>

int main() {

int number;

printf("Enter an integer: ");

   // reads and stores input

scanf("%d", &number);

   // displays output

printf("You entered: %d", number);

return 0;

}
```

```
Enter an integer: 25

You entered: 25
```

3. C Program to Add Two Integers

```
#include <stdio.h>
int main() {


int number1, number2, sum;


printf("Enter two integers: ");
scanf("%d %d", &number1, &number2);


   // calculating sum
sum = number1 + number2;


printf("%d + %d = %d", number1, number2, sum);
return 0;
}
```

```
Enter two integers: 12
11
12 + 11 = 23
```

4. Program to Multiply Two Numbers

```
#include <stdio.h>
int main()
{
    double a, b, product;
    printf("Enter two numbers: ");
    scanf("%lf %lf", &a, &b);
    // Calculating product
    product = a * b;
    // %.2lf displays number up to 2 decimal point
    printf("Product = %.2lf", product);
    return 0;
}
```

OUTPUT

```
Enter two numbers: 2.4
1.12
Product = 2.69
```

5. Program to Print ASCII Value

```
#include <stdio.h>
int main() {
    char c;
    printf("Enter a character: ");
    scanf("%c", &c);

    // %d displays the integer value of a character
    // %c displays the actual character
    printf("ASCII value of %c = %d", c, c);

    return 0;
}
```

OUTPUT

```
Enter a character: G
ASCII value of G = 71
```

6..Program to Compute Quotient and Remainder

```
#include <stdio.h>
int main() {
    int dividend, divisor, quotient, remainder;
    printf("Enter dividend: ");
    scanf("%d", &dividend);
    printf("Enter divisor: ");
    scanf("%d", &divisor);

    // Computes quotient
    quotient = dividend / divisor;

    // Computes remainder
    remainder = dividend % divisor;

    printf("Quotient = %d\n", quotient);
    printf("Remainder = %d", remainder);
    return 0;
}
```

OUTPUT

```
Enter dividend: 25
Enter divisor: 4
Quotient = 6
Remainder = 1
```

7. Program to Find the Size of Variables

```
#include<stdio.h>

int main() {

   int intType;

   float floatType;

   double doubleType;

   char charType;


   // sizeof evaluates the size of a variable

   printf("Size of int: %zu bytes\n", sizeof(intType));

   printf("Size of float: %zu bytes\n", sizeof(floatType));

   printf("Size of double: %zu bytes\n", sizeof(doubleType));

   printf("Size of char: %zu byte\n", sizeof(charType));

   return 0;

}
```

OUTPUT

```
Size of int: 4 bytes
Size of float: 4 bytes
Size of double: 8 bytes
Size of char: 1 byte
```

8. Swap Numbers Using Temporary Variable

```
#include<stdio.h>

int main() {

  double first, second, temp;

  printf("Enter first number: ");

  scanf("%lf", &first);

  printf("Enter second number: ");

  scanf("%lf", &second);


  // value of first is assigned to temp

  temp = first;


  // value of second is assigned to first

  first = second;


  // value of temp (initial value of first) is assigned to second

  second = temp;


  // %.2lf displays number up to 2 decimal points

  printf("\nAfter swapping, first number = %.2lf\n", first);

  printf("After swapping, second number = %.2lf", second);

  return 0;

}
```

OUTPUT

```
Enter first number: 1.20
Enter second number: 2.45

After swapping, first number = 2.45
After swapping, second number = 1.20
```

9.Program to Check Even or Odd

```
#include <stdio.h>
int main() {
    int num;
    printf("Enter an integer: ");
    scanf("%d", &num);

    // true if num is perfectly divisible by 2
    if(num % 2 == 0)
        printf("%d is even.", num);
    else
        printf("%d is odd.", num);

    return 0;
}
```

OUTPUT

```
Enter an integer: -7
```

```
-7 is odd.
```

10.Program to Check Odd or Even Using the Ternary Operator

```
#include <stdio.h>
int main() {
    int num;
    printf("Enter an integer: ");
    scanf("%d", &num);

    (num % 2 == 0) ? printf("%d is even.", num) : printf("%d is
odd.", num);
    return 0;
}
```

OUTPUT

```
Enter an integer: 33
33 is odd.
```

11.Reverse an Integer

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

    while (n != 0) {

        remainder = n % 10;

        rev = rev * 10 + remainder;

        n /= 10;

    }

    printf("Reversed number = %d", rev);

    return 0;

}
```

OUTPUT

```
Enter an integer: 2345
Reversed number = 5432
```

12.Power of a Number Using the while Loop

```
#include <stdio.h>

int main() {

    int base, exp;

    long double result = 1.0;

    printf("Enter a base number: ");

    scanf("%d", &base);

    printf("Enter an exponent: ");

    scanf("%d", &exp);



    while (exp != 0) {

        result *= base;
        --exp;
    }
    printf("Answer = %.0Lf", result);
    return 0;
}
```

OUTPUT

```
Enter a base number: 3
Enter an exponent: 4
Answer = 81
```

13.Power Using pow() Function

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


int main() {
    double base, exp, result;
    printf("Enter a base number: ");
    scanf("%lf", &base);
    printf("Enter an exponent: ");
    scanf("%lf", &exp);


    // calculates the power
    result = pow(base, exp);


    printf("%.1lf^%.1lf = %.2lf", base, exp, result);
    return 0;
}
```

OUTPUT

```
Enter a base number: 2.3
Enter an exponent: 4.5
2.3^4.5 = 42.44
```

14.Program to Check Palindrome

```
#include <stdio.h>

int main() {

  int n, reversed = 0, remainder, original;

    printf("Enter an integer: ");

    scanf("%d", &n);

    original = n;


    // reversed integer is stored in reversed variable

    while (n != 0) {

        remainder = n % 10;

        reversed = reversed * 10 + remainder;

        n /= 10;

    }


    // palindrome if orignal and reversed are equal

    if (original == reversed)

        printf("%d is a palindrome.", original);

    else
```

```
        printf("%d is not a palindrome.", original);


    return 0;
}
```

OUTPUT

```
Enter an integer: 1001
1001 is a palindrome.
```

15.Program to Check Prime Number

```
#include <stdio.h>
int main() {
  int n, i, flag = 0;
  printf("Enter a positive integer: ");
  scanf("%d", &n);


  for (i = 2; i <= n / 2; ++i) {
    // condition for non-prime
    if (n % i == 0) {
      flag = 1;
      break;
    }
  }


  if (n == 1) {
    printf("1 is neither prime nor composite.");
  }
  else {
    if (flag == 0)
      printf("%d is a prime number.", n);
    else
      printf("%d is not a prime number.", n);
  }

  return 0;
}
```

OUTPUT

```
Enter a positive integer: 29
29 is a prime number.
```

16.Display Prime Numbers Between Two Intervals

```
#include <stdio.h>


int main() {
   int low, high, i, flag;
   printf("Enter two numbers(intervals): ");
   scanf("%d %d", &low, &high);
   printf("Prime numbers between %d and %d are: ", low, high);

   // iteration until low is not equal to high
   while (low < high) {
      flag = 0;
```

```
    // ignore numbers less than 2

    if (low <= 1) {

      ++low;

      continue;

    }


    // if low is a non-prime number, flag will be 1

    for (i = 2; i <= low / 2; ++i) {


      if (low % i == 0) {

        flag = 1;

        break;

      }

    }


    if (flag == 0)

      printf("%d ", low);


    // to check prime for the next number

    // increase low by 1

    ++low;

  }


  return 0;

}
```

OUTPUT

```
Enter two numbers(intervals): 20
50
Prime numbers between 20 and 50 are: 23 29 31 37 41 43 47
```

17.Display Prime Numbers when Larger Number is Entered first

```c
#include <stdio.h>

int main() {
  int low, high, i, flag, temp;
  printf("Enter two numbers(intervals): ");
  scanf("%d %d", &low, &high);

  // swap numbers if low is greater than high
  if (low > high) {
    temp = low;
    low = high;
    high = temp;
  }

  printf("Prime numbers between %d and %d are: ", low, high);
  while (low < high) {
    flag = 0;

    // ignore numbers less than 2
    if (low <= 1) {
      ++low;
      continue;
    }

    for (i = 2; i <= low / 2; ++i) {
      if (low % i == 0) {
        flag = 1;
        break;
      }
    }
    if (flag == 0)
      printf("%d ", low);
    ++low;
  }

  return 0;
}
```

18.Check Armstrong Number of three digits

```
#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;

       // removing last digit from the orignal number
       originalNum /= 10;
    }

    if (result == num)
        printf("%d is an Armstrong number.", num);
    else
        printf("%d is not an Armstrong number.", num);

    return 0;
}
```

OUTPUT

```
Enter a three-digit integer: 371
371 is an Armstrong number.
```

19.Check Armstrong Number of n digits

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

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

```
  float result = 0.0;

  printf("Enter an integer: ");
  scanf("%d", &num);

  originalNum = num;

  // store the number of digits of num in n
  for (original Num = num; original Num != 0; ++n) {
      original Num /= 10;
  }

  for (original Num = num; original Num != 0; originalNum /= 10) {
      remainder = original Num % 10;

    // store the sum of the power of individual digits in result
    result + = pow (remainder, n);
  }

  // if num is equal to result, the number is an Armstrong number
  if ((int)result == num)
   printf("%d is an Armstrong number.", num);
  else
   printf("%d is not an Armstrong number.", num);
  return 0;
}
```

OUTPUT

```
Enter an integer: 1634
1634 is an Armstrong number.
```

20.Armstrong Numbers Between Two Integers

```
#include <math.h>
#include <stdio.h>
int main() {
  int low, high, number, originalNumber, rem, count = 0;
  double result = 0.0;
  printf("Enter two numbers(intervals): ");
  scanf("%d %d", &low, &high);
  printf("Armstrong numbers between %d and %d are: ", low, high);

  // swap numbers if high < low
  if (high < low) {
    high += low;
    low = high - low;
    high -= low;
  }

  // iterate number from (low + 1) to (high - 1)
  // In each iteration, check if number is Armstrong
  for (number = low + 1; number < high; ++number) {
    original Number = number;

    // number of digits calculation
    while (original Number != 0) {
      original Number /= 10;
      ++count;
    }
    Original Number = number;

    // result contains sum of nth power of individual digits
    while (original Number != 0) {
      rem = original Number % 10;
      result += pow (rem, count);
      originalNumber /= 10;
    }

    // check if number is equal to the sum of nth power of individual digits
    if (( int )result == number) {
      printf ("%d ", number);
    }

    // resetting the values
    count = 0;
    result = 0;
  }

  return 0;
}
```

OUTPUT

```
Enter two numbers(intervals): 200
2000
Armstrong numbers between 200 and 2000 are: 370 371 407 1634
```

21.Factors of a Positive Integer

```
#include <stdio.h>
int main() {
    int num, i;
    printf("Enter a positive integer: ");
    scanf("%d", &num);
    printf("Factors of %d are: ", num);
    for (i = 1; i <= num; ++i) {
        if (num % i == 0) {
            printf("%d ", i);
        }
    }
    return 0;
}
```

OUTPUT

```
Enter a positive integer: 60
Factors of 60 are: 1 2 3 4 5 6 10 12 15 20 30 60
```

22.Simple Calculator using switch Statement

```
#include <stdio.h>
int main() {
  char op;
  double first, second;
  printf("Enter an operator (+, -, *, /): ");
  scanf("%c", &op);
  printf("Enter two operands: ");
  scanf("%lf %lf", &first, &second);

  switch (op) {
    case '+':
      printf("%.1lf + %.1lf = %.1lf", first, second, first + second);
      break;
    case '-':
      printf("%.1lf - %.1lf = %.1lf", first, second, first - second);
      break;
    case '*':
      printf("%.1lf * %.1lf = %.1lf", first, second, first * second);
      break;
    case '/':
      printf("%.1lf / %.1lf = %.1lf", first, second, first / second);
      break;
    // operator doesn't match any case constant
    default:
      printf("Error! operator is not correct");
  }

  return 0;
}
```

OUTPUT

```
Enter an operator (+, -, *,): *
Enter two operands: 1.5
4.5
1.5 * 4.5 = 6.8
```

23.Half Pyramid of *

```
*
* *
* * *
* * * *
* * * * *
```

C Program

```
#include <stdio.h>
int main() {
  int i, j, rows;
  printf("Enter the number of rows: ");
  scanf("%d", &rows);
  for (i = 1; i <= rows; ++i) {
    for (j = 1; j <= i; ++j) {
      printf("* ");
    }
    printf("\n");
  }
  return 0;
}
```

24.Example 2: Half Pyramid of Numbers

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

C Program

```
#include <stdio.h>
int main() {
  int i, j, rows;
  printf("Enter the number of rows: ");
  scanf("%d", &rows);
  for (i = 1; i <= rows; ++i) {
    for (j = 1; j <= i; ++j) {
      printf("%d ", j);
    }
    printf("\n");
  }
  return 0;
}
```

25. Half Pyramid of Alphabets

```
A
B B
C C C
D D D D
E E E E E
```

C Program

```
#include <stdio.h>

int main() {

  int i, j;

  char input, alphabet = 'A';

  printf("Enter an uppercase character you want to print in the last row: ");

  scanf("%c", &input);

  for (i = 1; i <= (input - 'A' + 1); ++i) {

    for (j = 1; j <= i; ++j) {

      printf("%c ", alphabet);

    }

    ++alphabet;

    printf("\n");

  }

  return 0;

}
```

26. Inverted half pyramid of *

```
* * * * *
* * * *
* * *
* *
*
```

C Program

```
#include <stdio.h>
int main() {
   int i, j, rows;
   printf("Enter the number of rows: ");
   scanf("%d", &rows);
   for (i = rows; i >= 1; --i) {
      for (j = 1; j <= i; ++j) {
         printf("* ");
      }
      printf("\n");
   }
   return 0;
}
```

27. Inverted half pyramid of numbers

```
1 2 3 4 5
1 2 3 4
1 2 3
1 2
1
```

C Program

```
#include <stdio.h>
int main() {
    int i, j, rows;
    printf("Enter the number of rows: ");
    scanf("%d", &rows);
    for (i = rows; i >= 1; --i) {
        for (j = 1; j <= i; ++j) {
            printf("%d ", j);
        }
        printf("\n");
    }
    return 0;
}
```

28. Full Pyramid of *

```
        *
      * * *
    * * * * *
  * * * * * * *
* * * * * * * * *
```

C Program

```
#include <stdio.h>
int main() {
    int i, space, rows, k = 0;
    printf("Enter the number of rows: ");
    scanf("%d", &rows);
    for (i = 1; i <= rows; ++i, k = 0) {
        for (space = 1; space <= rows - i; ++space) {
            printf("  ");
        }
        while (k != 2 * i - 1) {
            printf("* ");
            ++k;
    }
    printf("\n");
  }
  return 0;
}
```

29. Full Pyramid of Numbers

```
        1
      2 3 2
    3 4 5 4 3
  4 5 6 7 6 5 4
5 6 7 8 9 8 7 6 5
```

C Program

```
#include <stdio.h>
int main() {
  int i, space, rows, k = 0, count = 0, count1 = 0;
  printf("Enter the number of rows: ");
  scanf("%d", &rows);
  for (i = 1; i <= rows; ++i) {
    for (space = 1; space <= rows - i; ++space) {
      printf("  ");
      ++count;
    }
    while (k != 2 * i - 1) {
      if (count <= rows - 1) {
        printf("%d ", i + k);
        ++count;
      } else {
        ++count1;
        printf("%d ", (i + k - 2 * count1));
      }
      ++k;
    }
    count1 = count = k = 0;
    printf("\n");
  }
  return 0;
}
```

30. Inverted full pyramid of *

```
* * * * * * * * *
  * * * * * * *
    * * * * *
      * * *
        *
```

C Program

```
#include <stdio.h>
int main() {
    int rows, i, j, space;
    printf("Enter the number of rows: ");
    scanf("%d", &rows);
    for (i = rows; i >= 1; --i) {
      for (space = 0; space < rows - i; ++space)
        printf("  ");
      for (j = i; j <= 2 * i - 1; ++j)
        printf("* ");
      for (j = 0; j < i - 1; ++j)
        printf("* ");
      printf("\n");
    }
```

```
  return 0;
}
```

31. Pascal's Triangle

```
          1
        1   1
      1   2   1
    1   3   3    1
  1  4    6   4   1
1  5   10   10  5    1
```

C Program

```
#include <stdio.h>
int main() {
  int rows, coef = 1, space, i, j;
  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;
}
```

32. Floyd's Triangle.

```
1
2 3
4 5 6
7 8 9 10
```

C Program

```
#include <stdio.h>
int main() {
   int rows, i, j, number = 1;
   printf("Enter the number of rows: ");
   scanf("%d", &rows);
   for (i = 1; i <= rows; i++) {
      for (j = 1; j <= i; ++j) {
         printf("%d ", number);
         ++number;
      }
      printf("\n");
   }
   return 0;
}
```

33.Simple Calculator using switch Statement

```
#include <stdio.h>
int main() {
  char op;
  double first, second;
  printf("Enter an operator (+, -, *, /): ");
  scanf("%c", &op);
  printf("Enter two operands: ");
  scanf("%lf %lf", &first, &second);

  switch (op) {
    case '+':
      printf("%.1lf + %.1lf = %.1lf", first, second, first + second);
      break;
    case '-':
      printf("%.1lf - %.1lf = %.1lf", first, second, first - second);
      break;
    case '*':
      printf("%.1lf * %.1lf = %.1lf", first, second, first * second);
      break;
    case '/':
      printf("%.1lf / %.1lf = %.1lf", first, second, first / second);
      break;
    // operator doesn't match any case constant
    default:
      printf("Error! operator is not correct");
  }

  return 0;
}
```

OUTPUT

```
Enter an operator (+, -, *,): *
Enter two operands: 1.5
4.5
1.5 * 4.5 = 6.8
```

34.Program to Check Vowel or consonant

```c
#include <stdio.h>
int main() {
    char c;
    int lowercase_vowel, uppercase_vowel;
    printf("Enter an alphabet: ");
    scanf("%c", &c);

    // evaluates to 1 if variable c is a lowercase vowel
    lowercase_vowel = (c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u');

    // evaluates to 1 if variable c is a uppercase vowel
    uppercase_vowel = (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U');

    // evaluates to 1 (true) if c is a vowel
    if (lowercase_vowel || uppercase_vowel)
        printf("%c is a vowel.", c);
    else
        printf("%c is a consonant.", c);
    return 0;
}
```

OUTPUT

```
Enter an alphabet: G
G is a consonant.
```

35.Using if Statement

```
#include <stdio.h>
int main() {
   double n1, n2, n3;
   printf("Enter three different numbers: ");
   scanf("%lf %lf %lf", &n1, &n2, &n3);

   // if n1 is greater than both n2 and n3, n1 is the largest
   if (n1 >= n2 && n1 >= n3)
      printf("%.2f is the largest number.", n1);

   // if n2 is greater than both n1 and n3, n2 is the largest
   if (n2 >= n1 && n2 >= n3)
      printf("%.2f is the largest number.", n2);

   // if n3 is greater than both n1 and n2, n3 is the largest
   if (n3 >= n1 && n3 >= n2)
      printf("%.2f is the largest number.", n3);
   return 0;
}
```

36. Using if...else Ladder

```
#include <stdio.h>
int main() {
   double n1, n2, n3;
```

```
    printf("Enter three numbers: ");
    scanf("%lf %lf %lf", &n1, &n2, &n3);

    // if n1 is greater than both n2 and n3, n1 is the largest
    if (n1 >= n2 && n1 >= n3)
        printf("%.2lf is the largest number.", n1);

    // if n2 is greater than both n1 and n3, n2 is the largest
    else if (n2 >= n1 && n2 >= n3)
        printf("%.2lf is the largest number.", n2);

    // if both above conditions are false, n3 is the largest
    else
        printf("%.2lf is the largest number.", n3);

    return 0;
}
```

37.Using Nested if...else

```
#include <stdio.h>
int main() {
    double n1, n2, n3;
    printf("Enter three numbers: ");
    scanf("%lf %lf %lf", &n1, &n2, &n3);

    if (n1 >= n2) {
        if (n1 >= n3)
            printf("%.2lf is the largest number.", n1);
        else
```

```
            printf("%.2lf is the largest number.", n3);
    } else {
        if (n2 >= n3)
            printf("%.2lf is the largest number.", n2);
        else
            printf("%.2lf is the largest number.", n3);
    }

    return 0;
}
```

OUTPUT

```
Enter three numbers: -4.5
3.9
5.6
5.60 is the largest number.
```

38.Program to Find Roots of a Quadratic Equation

```
#include <math.h>
#include <stdio.h>
int main() {
    double a, b, c, discriminant, root1, root2, realPart, imagPart;
    printf("Enter coefficients a, b and c: ");
    scanf("%lf %lf %lf", &a, &b, &c);

    discriminant = b * b - 4 * a * c;

    // condition for real and different roots
    if (discriminant > 0) {
        root1 = (-b + sqrt(discriminant)) / (2 * a);
        root2 = (-b - sqrt(discriminant)) / (2 * a);

        printf("root1 = %.2lf and root2 = %.2lf", root1, root2);

    }


    // condition for real and equal roots

    else if (discriminant == 0) {

        root1 = root2 = -b / (2 * a);

        printf("root1 = root2 = %.2lf;", root1);

    }


    // if roots are not real

    else {

        realPart = -b / (2 * a);

        imagPart = sqrt(-discriminant) / (2 * a);

        printf("root1 = %.2lf+%.2lfi and root2 = %.2f-%.2fi", realPart, imagPart, realPart,
imagPart);

    }


    return 0;

}
```

OUTPUT

```
Enter coefficients a, b and c: 2.3
4
5.6
root1 = -0.87+1.30i and root2 = -0.87-1.30i
```

39.Program to Check Leap Year

```
#include <stdio.h>

int main() {

   int year;
```

```
    printf("Enter a year: ");
    scanf("%d", &year);

    // leap year if perfectly divisible by 400
    if (year % 400 == 0) {
      printf("%d is a leap year.", year);
    }
    // not a leap year if divisible by 100
    // but not divisible by 400
    else if (year % 100 == 0) {
      printf("%d is not a leap year.", year);
    }
    // leap year if not divisible by 100
    // but divisible by 4
    else if (year % 4 == 0) {
      printf("%d is a leap year.", year);
    }
    // all other years are not leap years
    else {
      printf("%d is not a leap year.", year);
    }

    return 0;
  }
```

Output 1

```
Enter a year: 1900
1900 is not a leap year.
```

Output 2

```
Enter a year: 2012
```

```
2012 is a leap year.
```

40.Check Positive or Negative Using if...else

```
#include <stdio.h>
int main() {
    double num;
    printf("Enter a number: ");
    scanf("%lf", &num);
    if (num <= 0.0) {
        if (num == 0.0)
            printf("You entered 0.");
        else
            printf("You entered a negative number.");
    } else
        printf("You entered a positive number.");
    return 0;
}
```

41.Check Positive or Negative Using Nested if...else

```
#include <stdio.h>
int main() {
    double num;
    printf("Enter a number: ");
    scanf("%lf", &num);
    if (num < 0.0)
        printf("You entered a negative number.");
    else if (num > 0.0)
        printf("You entered a positive number.");
    else
        printf("You entered 0.");
    return 0;
}
```

Output 1

```
Enter a number: 12.3
You entered a positive number.
```

Output 2

```
Enter a number: 0
You entered 0.
```

42.Program to Check Alphabet

```
#include <stdio.h>
int main() {
    char c;
    printf("Enter a character: ");
    scanf("%c", &c);

    if ((c >= 'a' && c <= 'z') || (c >= 'A' && c <= 'Z'))
        printf("%c is an alphabet.", c);
    else
        printf("%c is not an alphabet.", c);

    return 0;
}
```

Output

```
Enter a character: *
* is not an alphabet
```

43.Sum of Natural Numbers Using for Loop

```
#include <stdio.h>
int main() {
    int n, i, sum = 0;

    printf("Enter a positive integer: ");
    scanf("%d", &n);

    for (i = 1; i <= n; ++i) {
        sum += i;
    }

    printf("Sum = %d", sum);
    return 0;
}
```

44.Sum of Natural Numbers Using while Loop

```
#include <stdio.h>
int main() {
    int n, i, sum = 0;
    printf("Enter a positive integer: ");
    scanf("%d", &n);
    i = 1;

    while (i <= n) {
        sum += i;

        ++i;

    }


    printf("Sum = %d", sum);

    return 0;

}
```

Output

```
Enter a positive integer: 100
Sum = 5050
```

45.Read Input Until a Positive Integer is Entered

```
#include <stdio.h>

int main() {

    int n, i, sum = 0;


    do {

        printf("Enter a positive integer: ");

        scanf("%d", &n);

    } while (n <= 0);


    for (i = 1; i <= n; ++i) {

        sum += i;

    }


    printf("Sum = %d", sum);

    return 0;

}
```

46.Factorial of a Number

```
#include <stdio.h>
int main() {
    int n, i;
    unsigned long long fact = 1;
    printf("Enter an integer: ");
    scanf("%d", &n);

    // shows error if the user enters a negative integer
    if (n < 0)
        printf("Error! Factorial of a negative number doesn't exist.");
    else {
        for (i = 1; i <= n; ++i) {
            fact *= i;
        }
        printf("Factorial of %d = %llu", n, fact);
    }

    return 0;
}
```

Output

```
Enter an integer: 10
Factorial of 10 = 3628800
```

47.Multiplication Table Up to 10

```
#include <stdio.h>
int main() {
  int n, i;
  printf("Enter an integer: ");
  scanf("%d", &n);
  for (i = 1; i <= 10; ++i) {
    printf("%d * %d = %d \n", n, i, n * i);
  }
  return 0;
}
```

Output

```
Enter an integer: 9
9 * 1 = 9
9 * 2 = 18
9 * 3 = 27
9 * 4 = 36
9 * 5 = 45
9 * 6 = 54
9 * 7 = 63
9 * 8 = 72
9 * 9 = 81
9 * 10 = 90
```

48.Multiplication Table Up to a range

```
#include <stdio.h>
int main() {

  int n, i, range;
  printf("Enter an integer: ");
  scanf("%d", &n);

  // prompt user for positive range
  do {
    printf("Enter the range (positive integer): ");
    scanf("%d", &range);
  } while (range <= 0);
  for (i = 1; i <= range; ++i) {
    printf("%d * %d = %d \n", n, i, n * i);
  }

  return 0;
}
```

Output

```
Enter an integer: 12
Enter the range (positive integer): -8
Enter the range (positive integer): 8
12 * 1 = 12
12 * 2 = 24
12 * 3 = 36
12 * 4 = 48
12 * 5 = 60
12 * 6 = 72
12 * 7 = 84
12 * 8 = 96
```

49.Fibonacci Series up to n terms

```
#include <stdio.h>
int main() {

  int i, n;

  // initialize first and second terms
  int t1 = 0, t2 = 1;

  // initialize the next term (3rd term)
  int nextTerm = t1 + t2;

  // get no. of terms from user
  printf("Enter the number of terms: ");
  scanf("%d", &n);

  // print the first two terms t1 and t2
  printf("Fibonacci Series: %d, %d, ", t1, t2);

  // print 3rd to nth terms
  for (i = 3; i <= n; ++i) {
    printf("%d, ", nextTerm);
    t1 = t2;
    t2 = nextTerm;
    nextTerm = t1 + t2;
  }

  return 0;
}
```

Output

```
Enter the number of terms: 10
Fibonacci Series: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34,
```

50. GCD Using for loop and if Statement

```
#include <stdio.h>
int main()
{
    int n1, n2, i, gcd;

    printf("Enter two integers: ");
    scanf("%d %d", &n1, &n2);
    for(i=1; i <= n1 && i <= n2; ++i)
    {
        // Checks if i is factor of both integers
        if(n1%i==0 && n2%i==0)
            gcd = i;
    }

    printf("G.C.D of %d and %d is %d", n1, n2, gcd);

    return 0;
}
```

51.GCD Using while loop and if...else Statement

```
#include <stdio.h>
int main()
{
    int n1, n2;

    printf("Enter two positive integers: ");
    scanf("%d %d",&n1,&n2);

    while(n1!=n2)
    {
        if(n1 > n2)
            n1 -= n2;
        else
            n2 -= n1;
    }
    printf("GCD = %d",n1);

    return 0;
}
```

52. GCD for both positive and negative numbers

```
#include <stdio.h>
int main()
{
    int n1, n2;

    printf("Enter two integers: ");
    scanf("%d %d",&n1,&n2);
    // if user enters negative number, sign of the number is changed to
positive
    n1 = ( n1 > 0) ? n1 : -n1;
    n2 = ( n2 > 0) ? n2 : -n2;

    while(n1!=n2)
    {
        if(n1 > n2)
            n1 -= n2;
        else
            n2 -= n1;
    }
    printf("GCD = %d",n1);

    return 0;
}
```

Output

```
Enter two integers: 81
-153
GCD = 9
```

53. Program to Print English Alphabets

```
#include <stdio.h>

int main() {

    char c;

    for (c = 'A'; c <= 'Z'; ++c)

        printf("%c ", c);

    return 0;

}
```

Output

```
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

54. Print Lowercase/Uppercase alphabets

```
#include <stdio.h>

int main() {

   char c;

   printf("Enter u to display uppercase alphabets.\n");

   printf("Enter l to display lowercase alphabets. \n");

   scanf("%c", &c);


   if (c == 'U' || c == 'u') {

      for (c = 'A'; c <= 'Z'; ++c)

         printf("%c ", c);

   } else if (c == 'L' || c == 'l') {

      for (c = 'a'; c <= 'z'; ++c)

         printf("%c ", c);

   } else {

      printf("Error! You entered an invalid character.");

   }


   return 0;

}
```

Output

```
Enter u to display uppercase alphabets.
Enter l to display lowercase alphabets. l
a b c d e f g h i j k l m n o p q r s t u v w x y z
```

55. Program to Count the Number of Digits

```
#include <stdio.h>

int main() {

  long long n;

  int count = 0;
  printf("Enter an integer: ");
  scanf("%lld", &n);

  // iterate at least once, then until n becomes 0
  // remove last digit from n in each iteration
  // increase count by 1 in each iteration
  do {
    n /= 10;
    ++count;
  } while (n != 0);

  printf("Number of digits: %d", count);
}
```

Output

```
Enter an integer: 3452
Number of digits: 4
```

56. Power of a Number Using the while Loop

```
#include <stdio.h>
int main() {
    int base, exp;
    long double result = 1.0;
    printf("Enter a base number: ");
    scanf("%d", &base);
    printf("Enter an exponent: ");
    scanf("%d", &exp);

    while (exp != 0) {
        result *= base;
        --exp;
    }
    printf("Answer = %.0Lf", result);
    return 0;
}
```

Output

```
Enter a base number: 3
Enter an exponent: 4
Answer = 81
```

57. Power Using pow() Function

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

int main() {
    double base, exp, result;
    printf("Enter a base number: ");
    scanf("%lf", &base);
    printf("Enter an exponent: ");
    scanf("%lf", &exp);

    // calculates the power
    result = pow(base, exp);

    printf("%.1lf^%.1lf = %.2lf", base, exp, result);
    return 0;
}
```

58. Program to Check Palindrome

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

    // reversed integer is stored in reversed variable
    while (n != 0) {
        remainder = n % 10;
        reversed = reversed * 10 + remainder;
        n /= 10;
    }

    // palindrome if orignal and reversed are equal
    if (original == reversed)
        printf("%d is a palindrome.", original);
    else
        printf("%d is not a palindrome.", original);

    return 0;
}
```

59.Program to Check Prime Number

```
#include <stdio.h>
int main() {
  int n, i, flag = 0;
  printf("Enter a positive integer: ");
  scanf("%d", &n);
  for (i = 2; i <= n / 2; ++i) {
    // condition for non-prime
    if (n % i == 0) {
      flag = 1;
      break;
    }
  }

  if (n == 1) {
    printf("1 is neither prime nor composite.");
  }
  else {
    if (flag == 0)
      printf("%d is a prime number.", n);
    else
      printf("%d is not a prime number.", n);
  }

  return 0;
}
```

Output

```
Enter a positive integer: 29
29 is a prime number.
```

60. Display Prime Numbers Between Two Intervals

```
#include <stdio.h>

int main() {
```

```
int low, high, i, flag;
printf("Enter two numbers(intervals): ");
scanf("%d %d", &low, &high);
printf("Prime numbers between %d and %d are: ", low, high);

// iteration until low is not equal to high
while (low < high) {
  flag = 0;

  // ignore numbers less than 2
  if (low <= 1) {
    ++low;
    continue;
  }

  // if low is a non-prime number, flag will be 1
  for (i = 2; i <= low / 2; ++i) {

    if (low % i == 0) {
      flag = 1;
      break;
    }
  }

  if (flag == 0)
    printf("%d ", low);

  // to check prime for the next number
  // increase low by 1
```

```
    ++low;
  }

  return 0;
}
```

Output

```
Enter two numbers(intervals): 20
50
Prime numbers between 20 and 50 are: 23 29 31 37 41 43 47
```

61. Check Armstrong Number of three digits

```
#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;

      // removing last digit from the orignal number
      originalNum /= 10;
   }

   if (result == num)
```

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

    return 0;
}
```

Output

```
Enter a three-digit integer: 371
371 is an Armstrong number.
```

62. Check Armstrong Number of n digits

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

int main() {
   int num, originalNum, remainder, n = 0;
   float result = 0.0;

   printf("Enter an integer: ");
   scanf("%d", &num);

   originalNum = num;

   // store the number of digits of num in n
   for (originalNum = num; originalNum != 0; ++n) {
       originalNum /= 10;
   }

   for (originalNum = num; originalNum != 0; originalNum /= 10) {
      remainder = originalNum % 10;

      // store the sum of the power of individual digits in result
      result += pow(remainder, n);
   }

   // if num is equal to result, the number is an Armstrong number
   if ((int)result == num)
    printf("%d is an Armstrong number.", num);
   else
    printf("%d is not an Armstrong number.", num);
   return 0;
}
```

Output

```
Enter an integer: 1634
1634 is an Armstrong number.
```

63. Integer as a Sum of Two Prime Numbers

```
#include <stdio.h>
int checkPrime(int n);
int main() {
  int n, i, flag = 0;
  printf("Enter a positive integer: ");
  scanf("%d", &n);

  for (i = 2; i <= n / 2; ++i) {
    // condition for i to be a prime number
    if (checkPrime(i) == 1) {
      // condition for n-i to be a prime number
      if (checkPrime(n - i) == 1) {
        printf("%d = %d + %d\n", n, i, n - i);
        flag = 1;
      }
    }
  }

  if (flag == 0)
    printf("%d cannot be expressed as the sum of two prime numbers.", n);

  return 0;
}

// function to check prime number
int checkPrime(int n) {
  int i, isPrime = 1;

  // 0 and 1 are not prime numbers
  if (n == 0 || n == 1) {
    isPrime = 0;
  }
  else {
    for(i = 2; i <= n/2; ++i) {
      if(n % i == 0) {
        isPrime = 0;
        break;
      }
    }
  }
  return isPrime;
}
```

Output

```
Enter a positive integer: 34
34 = 3 + 31
34 = 5 + 29
34 = 11 + 23
34 = 17 + 17
```

64. Simple Calculator using switch Statement

```
#include <stdio.h>
int main() {
  char op;
  double first, second;
  printf("Enter an operator (+, -, *, /): ");
  scanf("%c", &op);
  printf("Enter two operands: ");
  scanf("%lf %lf", &first, &second);

  switch (op) {
    case '+':
      printf("%.1lf + %.1lf = %.1lf", first, second, first + second);
      break;
    case '-':
      printf("%.1lf - %.1lf = %.1lf", first, second, first - second);
      break;
    case '*':
      printf("%.1lf * %.1lf = %.1lf", first, second, first * second);
      break;
    case '/':
      printf("%.1lf / %.1lf = %.1lf", first, second, first / second);
      break;
    // operator doesn't match any case constant
    default:
      printf("Error! operator is not correct");
  }

  return 0;
}
```

Output

```
Enter an operator (+, -, *,): *
Enter two operands: 1.5
4.5
1.5 * 4.5 = 6.8
```

65. Check Prime and Armstrong

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

int checkPrimeNumber(int n);
int checkArmstrongNumber(int n);

int main() {
   int n, flag;
   printf("Enter a positive integer: ");
   scanf("%d", &n);

   // check prime number
```

```
  flag = checkPrimeNumber(n);
  if (flag == 1)
    printf("%d is a prime number.\n", n);
  else
    printf("%d is not a prime number.\n", n);

  // check Armstrong number
  flag = checkArmstrongNumber(n);
  if (flag == 1)
    printf("%d is an Armstrong number.", n);
  else
    printf("%d is not an Armstrong number.", n);
  return 0;
}

// function to check prime number
int checkPrimeNumber(int n) {
  int i, flag = 1, squareRoot;

  // computing the square root
  squareRoot = sqrt(n);
  for (i = 2; i <= squareRoot; ++i) {
    // condition for non-prime number
    if (n % i == 0) {
      flag = 0;
      break;
    }
  }
  return flag;
```

```
}


// function to check Armstrong number
int checkArmstrongNumber(int num) {
  int originalNum, remainder, n = 0, flag;
  double result = 0.0;

  // store the number of digits of num in n
  for (originalNum = num; originalNum != 0; ++n) {
    originalNum /= 10;
  }

  for (originalNum = num; originalNum != 0; originalNum /= 10) {
    remainder = originalNum % 10;

    // store the sum of the power of individual digits in result
    result += pow(remainder, n);
  }

  // condition for Armstrong number
  if (round(result) == num)
    flag = 1;
  else
    flag = 0;
  return flag;
}
```

Output

```
Enter a positive integer: 407
407 is not a prime number.
```

```
407 is an Armstrong number.
```

66. Program to convert binary to decimal

```
#include <math.h>

#include <stdio.h>

int convert(long long n);

int main() {

    long long n;

    printf("Enter a binary number: ");

    scanf("%lld", &n);

    printf("%lld in binary = %d in decimal", n, convert(n));

    return 0;

}


int convert(long long n) {

    int dec = 0, i = 0, rem;

    while (n != 0) {

        rem = n % 10;

        n /= 10;

        dec += rem * pow(2, i);

        ++i;

    }

    return dec;

}
```

Output

```
Enter a binary number: 110110111
110110111 in binary = 439
```

67. Program to convert decimal to binary

```
#include <math.h>
#include <stdio.h>
long long convert(int n);
int main() {
    int n;
    printf("Enter a decimal number: ");
    scanf("%d", &n);
    printf("%d in decimal = %lld in binary", n, convert(n));
    return 0;
}

long long convert(int n) {
    long long bin = 0;
    int rem, i = 1, step = 1;
    while (n != 0) {
        rem = n % 2;
        printf("Step %d: %d/2, Remainder = %d, Quotient = %d\n", step++, n, rem, n / 2);
        n /= 2;
        bin += rem * i;
        i *= 10;
    }
    return bin;
}
```

Output

```
Enter a decimal number: 19
Step 1: 19/2, Remainder = 1, Quotient = 9
Step 2: 9/2, Remainder = 1, Quotient = 4
Step 3: 4/2, Remainder = 0, Quotient = 2
Step 4: 2/2, Remainder = 0, Quotient = 1
Step 5: 1/2, Remainder = 1, Quotient = 0
19 in decimal = 10011 in binary
```

68. Reverse a sentence using recursion

```
#include <stdio.h>

void reverseSentence();

int main() {

    printf("Enter a sentence: ");

    reverseSentence();

    return 0;

}


void reverseSentence() {

    char c;

    scanf("%c", &c);

    if (c != '\n') {

        reverseSentence();

        printf("%c", c);

    }

}
```

Output

```
Enter a sentence: margorp emosewa
awesome program
```

69. Store Numbers and Calculate Average Using Arrays

```
#include <stdio.h>

int main() {

    int n, i;

    float num[100], sum = 0.0, avg;


    printf("Enter the numbers of elements: ");
    scanf("%d", &n);

    while (n > 100 || n < 1) {
        printf("Error! number should in range of (1 to 100).\n");
        printf("Enter the number again: ");
        scanf("%d", &n);
    }

    for (i = 0; i < n; ++i) {
        printf("%d. Enter number: ", i + 1);
        scanf("%f", &num[i]);
        sum += num[i];
    }

    avg = sum / n;
    printf("Average = %.2f", avg);
    return 0;
}
```

Output

```
Enter the numbers of elements: 6
1. Enter number: 45.3
2. Enter number: 67.5
3. Enter number: -45.6
4. Enter number: 20.34
5. Enter number: 33
6. Enter number: 45.6
Average = 27.69
```

70. Largest Element in an array

https://www.programiz.com/c-programming/examples