#1 Multiplication table from 1 to 10

int Multiplication(int num1, int num2)

{

    return num1 \* num2;

}

void PrintTableHeader(int numOfMultipliers)

{

    cout << "\n\n\t\t\tMultiplication Table From1 to 10\n\n";

    for (int i = 1; i <= numOfMultipliers; i++)

    {

        cout << "\t" << i;

    }

    cout << "\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

}

string ColumnSeperator(int i)

{

    if (i == 10)

    {

        return "  |";

    }

    else

    {

        return "   |";

    }

}

void MultiplicationTable(int numOfMultipliers)

{

    PrintTableHeader(numOfMultipliers);

    for (int i = 1; i <= numOfMultipliers; i++)

    {

        cout << " " << i << ColumnSeperator(i) << "\t";

        for (int j = 1; j <= numOfMultipliers; j++)

        {

            cout << Multiplication(i, j) << "\t";

        }

        cout << endl;

    }

}

int main()

{

    MultiplicationTable(10);

}

#2 Print All prime Numbers From 1 to N

enum enPrimeNotPrime

{

    Prime = 1,

    NotPrime = 2

};

int ReadPositiveNumbers(string message)

{

    int num;

    do

    {

        cout << message << endl;

        cin >> num;

    } while (num <= 0);

    return num;

}

enPrimeNotPrime checkPrime(int number)

{

    int middleOfNum = round(number / 2);

    for (int counter = 2; counter <= middleOfNum; counter++)

    {

        if (number % counter == 0)

        {

            return enPrimeNotPrime ::NotPrime;

        }

    }

    return enPrimeNotPrime::Prime;

}

void PrintPrimeNumbersFrom1ToN(int N)

{

    cout << "\nOutput is :\n";

    for (int i = 1; i <= N; i++)

    {

        if (checkPrime(i) == enPrimeNotPrime::Prime)

        {

            cout << i << "\n";

        }

    }

}

int main()

{

    PrintPrimeNumbersFrom1ToN(ReadPositiveNumbers("please enter a positive number"));

}

#3 Perfect Number

Perfect number =sum (all divisors)

int ReadPositiveNumbers(string message)

{

    int num;

    do

    {

        cout << message << endl;

        cin >> num;

    } while (num <= 0);

    return num;

}

bool isPerfectnumber(int number)

{

    int middle = number / 2;

    int sumOfDivisors = 0;

    for (int i = 1; i <= middle; i++)

    {

        if (number % i == 0)

            sumOfDivisors += i;

    }

    return sumOfDivisors == number;

}

void printResult(int number)

{

    bool isPerfect = isPerfectnumber(number);

    if (isPerfect)

        cout << number << " is Perfect number ";

    else

        cout << number << " is not perfect number ";

}

int main()

{

    int number = ReadPositiveNumbers("Enter a positive number ");

    printResult(number);

}

#4 perfect number from 1 to n

int ReadPositiveNumbers(string message)

{

    int num;

    do

    {

        cout << message << endl;

        cin >> num;

    } while (num <= 0);

    return num;

}

bool isPerfectnumber(int number)

{

    int middle = number / 2;

    int sumOfDivisors = 0;

    for (int i = 1; i <= middle; i++)

    {

        if (number % i == 0)

            sumOfDivisors += i;

    }

    return sumOfDivisors == number;

}

void printPerfectNumbersFrom1toN(int N)

{

    cout << "Perfect numbers From 1 to " << N << " are :\n";

    for (int i = 1; i <= N; i++)

    {

        if (isPerfectnumber(i))

            cout << i << endl;

    }

}

int main()

{

    int N = ReadPositiveNumbers("Enter a positive number ");

    printPerfectNumbersFrom1toN(N);

}

#5 Print Digits in a Reversed Order

int ReadPositiveNumbers(string message)

{

    int num;

    do

    {

        cout << message << endl;

        cin >> num;

    } while (num <= 0);

    return num;

}

void printReversedDigits(int number)

{

    int reminder = 0;

    do

    {

        reminder = number % 10;

        cout << reminder << endl;

        number = number / 10;

    } while (number > 0);

}

int main()

{

    printReversedDigits(ReadPositiveNumbers("Enter a positive number :"));

}

#6 Sum of Digits

int ReadPositiveNumbers(string message)

{

    int num;

    do

    {

        cout << message << endl;

        cin >> num;

    } while (num <= 0);

    return num;

}

int SumOfDigits(int number)

{

    int reminder = 0, sum = 0;

    do

    {

        reminder = number % 10;

        sum += reminder;

        number = number / 10;

    } while (number > 0);

    return sum;

}

int main()

{

    int sumOfDigits = SumOfDigits(ReadPositiveNumbers("Enter a positive number :"));

    cout << "Sum OF Digits is : " << sumOfDigits;

}