CHANDIGARH UNIVERSITY

DEPARTMENT OF
COMPUTER SCIENCE & ENGINEERING
Discover. Learn. Empower.

Assignment 1

Student Name: Arush Tripathi UID: 22BCS11993
Branch: Computer Science and Engineering Section/Group:22BCSFL-603 A
Semester: 5th Date of Performance: 19/12/24

1. Sum of Natural Numbers up to N

Objective

Calculate the sum of all natural numbers from 1 to n, where n is a positive integer. Use the formula: Sum = n × (n + 1) / 2.

Input Format

One integer n, the upper limit for calculating the sum.

Constraints

$1 \leq n \leq 104$

Output Format

Print the sum of all natural numbers from 1 to n.

Test Cases

Input: 5 Output: 15

Input: 100 Output: 5050

Input: 1 Output: 1

Code:

```
#include <iostream>

using namespace std;
```

```
int sum_of_natural_numbers(int n) {
    return n * (n + 1) / 2;
}

int main() {
    cout << sum_of_natural_numbers(5) << endl;
    cout << sum_of_natural_numbers(100) << endl;
    return 0;
}
```

---

2. Check if a Number is Prime

Objective

Check if a given number n is a prime number.

Input Format

One integer n.

Constraints

$2 \leq n \leq 105$

Output Format

Print "Prime" if n is a prime number, otherwise print "Not Prime".

Test Cases

Input: 7 Output: Prime

Input: 9 Output: Not Prime

Input: 2 Output: Prime

Code:

```
#include <iostream>
using namespace std;

string is_prime(int n) {
    if (n <= 1) return "Not Prime";
    if (n <= 3) return "Prime";
    if (n % 2 == 0 || n % 3 == 0) return "Not Prime";

    for (int i = 5; i * i <= n; i += 6) {
        if (n % i == 0 || n % (i + 2) == 0) return "Not Prime";
    }
    return "Prime";
}

int main() {
    cout << is_prime(7) << endl;
    cout << is_prime(9) << endl;
    return 0;
}
```

---

3. Print Odd Numbers up to N

Objective

Print all odd numbers between 1 and n, inclusive.

Input Format

One integer n, the upper limit of the range.

Constraints

$1 \leq n \leq 104$

Output Format

A single line containing all odd numbers from 1 to n, separated by spaces.

Test Cases

Input: 10 Output: 1 3 5 7 9

Input: 7 Output: 1 3 5 7

Input: 1 Output: 1

Code:

```
#include <iostream>
using namespace std;

void print_odd_numbers(int n) {
    for (int i = 1; i <= n; i += 2) {
        cout << i << " ";
    }
    cout << endl;
}
```

```
int main() {
    print_odd_numbers(10);
    return 0;
}
```

---

4. Sum of Odd Numbers up to N

Objective

Calculate the sum of all odd numbers from 1 to n.

Input Format

One integer n.

Constraints

$1 \leq n \leq 104$

Output Format

Print the sum of all odd numbers from 1 to n.

Test Cases

Input: 5 Output: 9

Input: 10 Output: 25

Input: 1 Output: 1

Code:

```
#include <iostream>
using namespace std;
```

```
int sum_of_odd_numbers(int n) {
    int sum = 0;
    for (int i = 1; i <= n; i += 2) {
        sum += i;
    }
    return sum;
}

int main() {
    cout << sum_of_odd_numbers(5) << endl;
    return 0;
}
```

---

5. Print Multiplication Table of a Number

Objective

Print the multiplication table of a given number n.

Input Format

One integer n.

Constraints

1 ≤ n ≤ 100

Output Format

For each integer i from 1 to 10, print the product n × i in the format: n × i = product.

Test Cases

Input: 3 Output: 3 x 1 = 3 3 x 2 = 6 ... 3 x 10 = 30

Input: 7 Output: 7 x 1 = 7 ... 7 x 10 = 70

Input: 10 Output: 10 x 1 = 10 ... 10 x 10 = 100

Code:

```
#include <iostream>
using namespace std;

void multiplication_table(int n) {
    for (int i = 1; i <= 10; ++i) {
        cout << n << " x " << i << " = " << n * i << endl;
    }
}

int main() {
    multiplication_table(3);
    return 0;
}
```

---

6. Count Digits in a Number

Objective

Count the total number of digits in a given number n.

Input Format

One integer n.

Constraints

$1 \leq n \leq 109$

Output Format

Print the total number of digits in n.

Test Cases

Input: 12345 Output: 5

Input: 900000 Output: 6

Input: 1 Output: 1

Code:

```
#include <iostream>
using namespace std;

int count_digits(int n) {
    return to_string(n).length();
}

int main() {
    cout << count_digits(12345) << endl;
    return 0;
}
```

7. Reverse a Number

Objective

Reverse the digits of a given number n.

Input Format

One integer n.

Constraints

1 ≤ n ≤ 109

Output Format

Print the number with its digits in reverse order.

Test Cases

Input: 12345 Output: 54321

Input: 9876 Output: 6789

Input: 1000 Output: 1

Code:

```
#include <iostream>
using namespace std;

int reverse_number(int n) {
    int reversed = 0;
    while (n > 0) {
        reversed = reversed * 10 + n % 10;
        n /= 10;
    }
    return reversed;
}

int main() {
    cout << reverse_number(12345) << endl;
    return 0;
}
```

---

8. Find the Largest Digit in a Number

Objective

Find the largest digit in a given number n.

Input Format

One integer n.

Constraints

$1 \leq n \leq 109$

Output Format

Print the largest digit in n.

Test Cases

Input: 2734 Output: 7

Input: 9450 Output: 9

Input: 1111 Output: 1

```
#include <iostream>
using namespace std;
```

```
int largest_digit(int n) {
    int largest = 0;
    while (n > 0) {
        int digit = n % 10;
        if (digit > largest) {
            largest = digit;
        }
        n /= 10;
    }
    return largest;
}

int main() {
    cout << largest_digit(2734) << endl;
    return 0;
}
```

---

9. Check if a Number is a Palindrome

Objective

Check whether a given number is a palindrome or not.

Input Format

One integer n.

Constraints

1 ≤ n ≤ 109

Output Format

Print "Palindrome" if the number is a palindrome, otherwise print "Not Palindrome".

Test Cases

Input: 121 Output: Palindrome

Input: 12345 Output: Not Palindrome

Input: 12321 Output: Palindrome

```
#include <iostream>
using namespace std;

string is_palindrome(int n) {
    string str = to_string(n);
    string reversed_str = string(str.rbegin(), str.rend());
    return str == reversed_str ? "Palindrome" : "Not Palindrome";
}

int main() {
    cout << is_palindrome(121) << endl;
    cout << is_palindrome(12345) << endl;
    return 0;
}
```

---

10. Find the Sum of Digits of a Number

Objective

Calculate the sum of the digits of a given number n.

Input Format

One integer n.

Constraints

1 ≤ n ≤ 109

Output Format

Print the sum of the digits of n.

Test Cases

Input: 12345 Output: 15

Input: 4567 Output: 22

Input: 999 Output: 27

Code:

```
#include <iostream>
using namespace std;

int sum_of_digits(int n) {
    int sum = 0;
    while (n > 0) {
        sum += n % 10;
        n /= 10;
    }
    return sum;
}


int main() {
    cout << sum_of_digits(12345) << endl;
    return 0;
}
```

---

11. Function Overloading for Calculating Area

Objective

Write a program to calculate the area of different shapes using function overloading.

Input Format

The program should accept:

1. Radius of the circle for the first function.
2. Length and breadth of the rectangle for the second function.
3. Base and height of the triangle for the third function.

Constraints

1 ≤ radius, length, breadth, base, height ≤ 103 Use 3.14159 for the value of π.

Output Format

Print the computed area of each shape in a new line.

Test Cases

Input: Radius = 5 Length = 4, breadth = 6 Base = 3, height = 7

Output: 78.53975 24 10.5

Input: Radius = 10 Length = 15, breadth = 8 Base = 12, height = 9

Output: 314.159 120 54

Problem 11: Calculate Factorial of a Number

Objective

Calculate the factorial of a given number nn.

Input Format

One integer nn.

Constraints

1≤n≤121 \leq n \leq 12

Output Format

Print the factorial of the number nn.

Code

```
#include <iostream>
using namespace std;

int factorial(int n) {
    int fact = 1;
    for (int i = 1; i <= n; i++) {
        fact *= i;
    }
    return fact;
}

int main() {
    int n;
    cin >> n;
    cout << factorial(n) << endl;
    return 0;
}
```

Problem 12: Count Digits in a Number

Objective

Count the total number of digits in a given number nn.

Input Format

One integer nn.

Constraints

1≤n≤1091 \leq n \leq 10^9

Output Format

Print the total number of digits in nn.

Code

```
#include <iostream>
using namespace std;

int countDigits(int n) {
    int count = 0;
    while (n > 0) {
        n /= 10;
        count++;
    }
    return count;
}

int main() {
    int n;
    cin >> n;
    cout << countDigits(n) << endl;
    return 0;
}
```

Problem 13: Reverse a Number

Objective

Reverse the digits of a given number nn.

Input Format

One integer nn.

Constraints

1≤n≤1091 \leq n \leq 10^9

Output Format

Print the number with its digits in reverse order.

Code

```
#include <iostream>
using namespace std;

int reverseNumber(int n) {
    int reversed = 0;
    while (n > 0) {
        reversed = reversed * 10 + n % 10;
        n /= 10;
    }
    return reversed;
}

int main() {
    int n;
    cin >> n;
    cout << reverseNumber(n) << endl;
    return 0;
}
```

Problem 14: Find Largest Digit in a Number

Objective

Find the largest digit in a given number nn.

Input Format

One integer nn.

Constraints

1≤n≤1091 \leq n \leq 10^9

Output Format

Print the largest digit in nn.

Code

```
#include <iostream>
using namespace std;

int largestDigit(int n) {
    int largest = 0;
    while (n > 0) {
        int digit = n % 10;
        if (digit > largest) {
            largest = digit;
        }
        n /= 10;
    }
    return largest;
}

int main() {
    int n;
    cin >> n;
    cout << largestDigit(n) << endl;
    return 0;
}
```

Problem 15: Check if a Number is Palindrome

Objective

Check whether a given number is a palindrome or not.

Input Format

One integer nn.

Constraints

1≤n≤1091 \leq n \leq 10^9

Output Format

Print "Palindrome" if the number is a palindrome, otherwise print "Not Palindrome".

Code

```
#include <iostream>
using namespace std;

bool isPalindrome(int n) {
    int original = n, reversed = 0;
    while (n > 0) {
        reversed = reversed * 10 + n % 10;
        n /= 10;
    }
    return original == reversed;
}

int main() {
    int n;
    cin >> n;
    if (isPalindrome(n)) {
        cout << "Palindrome" << endl;
    } else {
        cout << "Not Palindrome" << endl;
    }
    return 0;
}
```

Problem 16: Sum of Digits of a Number

Objective

Calculate the sum of the digits of a given number nn.

Input Format

One integer nn.

Constraints

1≤n≤1091 \leq n \leq 10^9

Output Format

Print the sum of the digits of nn.

Code

```
#include <iostream>
using namespace std;

int sumOfDigits(int n) {
    int sum = 0;
    while (n > 0) {
        sum += n % 10;
        n /= 10;
    }
    return sum;
}

int main() {
    int n;
    cin >> n;
    cout << sumOfDigits(n) << endl;
    return 0;
}
```

Problem 17: Print Fibonacci Series up to N Terms

Objective

Print the Fibonacci series up to nn terms.

Input Format

One integer nn.

Constraints

1≤n≤201 \leq n \leq 20

Output Format

Print the Fibonacci series up to nn terms.

Code

```
#include <iostream>
```

```
using namespace std;

void printFibonacci(int n) {
    int a = 0, b = 1;
    for (int i = 1; i <= n; i++) {
        cout << a << " ";
        int next = a + b;
        a = b;
        b = next;
    }
    cout << endl;
}

int main() {
    int n;
    cin >> n;
    printFibonacci(n);
    return 0;
}
```

Problem 18: Check Armstrong Number

Objective

Check if a given number is an Armstrong number.

Input Format

One integer nn.

Constraints

1≤n≤1051 \leq n \leq 10^5

Output Format

Print "Armstrong" if the number is an Armstrong number, otherwise print "Not Armstrong".

Code

```
#include <iostream>

#include <cmath>

using namespace std;


bool isArmstrong(int n) {

    int original = n, sum = 0;

    int digits = log10(n) + 1;

    while (n > 0) {

        int digit = n % 10;

        sum += pow(digit, digits);

        n /= 10;

    }

    return sum == original;

}


int main() {

    int n;
    cin >> n;
    if (isArmstrong(n)) {
        cout << "Armstrong" << endl;
    } else {
        cout << "Not Armstrong" << endl;
    }
    return 0;
}
```

Problem 19: Find GCD of Two Numbers

Objective

Calculate the greatest common divisor (GCD) of two given numbers.

Input Format

Two integers aa and bb.

Constraints

1≤a,b≤1051 \leq a, b \leq 10^5

Output Format

Print the GCD of aa and bb.

Code

```
#include <iostream>
using namespace std;

int gcd(int a, int b) {
    while (b != 0) {
        int temp = b;
        b = a % b;
        a = temp;
    }
    return a;
}

int main() {
    int a, b;
    cin >> a >> b;
    cout << gcd(a, b) << endl;
    return 0;
}
```