**1**

**Algorithm:**

1. **Start**
2. **Input**: Read the integer value of n (upper limit).
3. **Initialize**: Set sum = 0.
4. **Loop**: Iterate from i = 1 to n:
   * If i is an even number (i.e., i % 2 == 0), add i to sum.
5. **Output**: Display the value of sum.
6. **End**

**Pseudocode:**

Start

Input n

Set sum = 0

For i from 1 to n do

If i % 2 == 0 then

sum = sum + i

End If

End For

Output sum

End

**Flowchart:**

**Program:**

#include <stdio.h>

int main() {

int n, sum = 0;

// Input: Get the value of n

printf("Enter the value of n: ");

scanf("%d", &n);

// Loop through numbers from 1 to n

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

if(i % 2 == 0) { // Check if the number is even

sum += i; // Add even number to sum

}

}

// Output: Display the sum of even numbers

printf("The sum of even numbers from 1 to %d is: %d\n", n, sum);

return 0;

}

**2**

**Algorithm:**

1. **Start**
2. **Input**: Read the integer value n (number for which factorial is to be calculated).
3. **Define the recursive function** factorial(n):
   * If n == 0 or n == 1, return 1 (Base case).
   * Else, return n \* factorial(n-1) (Recursive case).
4. **Call the recursive function** to calculate factorial(n).
5. **Output**: Display the result of factorial(n).
6. **End**

**Pseudocode:**

Start

Input n

Function factorial(n):

If n == 0 or n == 1 then

Return 1

Else

Return n \* factorial(n - 1)

End Function

result = factorial(n)

Output result

End

**Flowchart:**

3

Algorithm:

**Start**

**Input**: Read the integer value n.

**Check if n is less than or equal to 1**:

If n <= 1, print "Not a prime number" and **End**.

**Loop**: Iterate from i = 2 to sqrt(n):

n % i == 0 (i.e., n is divisible by i), then print "Not a prime number" and **End**.

**If no divisor is found**: Print "Prime number".

**End**

Pseudocode:

Start

Input n

If n <= 1 then

Output "Not a prime number"

End

End If

For i = 2 to sqrt(n) do

If n % i == 0 then

Output "Not a prime number"

End

End If

End For

Output "Prime number"

End

Program:

#include <stdio.h>

#include <math.h>

// Function to check if the number is prime

int isPrime(int n) {

if (n <= 1) {

return 0; // Not a prime number

}

// Check divisibility from 2 to sqrt(n)

for (int i = 2; i <= sqrt(n); i++) {

if (n % i == 0) {

return 0; // Not a prime number

}

}

return 1; // Prime number

}

int main() {

int n;

// Input: Get the value of n

printf("Enter a number: ");

scanf("%d", &n);

// Check if the number is prime

if (isPrime(n)) {

printf("%d is a prime number.\n", n);

} else {

printf("%d is not a prime number.\n", n);

}

return 0;

}

4

Algorithm:

 **Start**

 **Input**: Read the integer value n (size of the array).

 **Input**: Read the n elements into an array.

 **Initialize**: Set the first element of the array as the initial largest number (i.e., largest = arr[0]).

 **Loop**: Iterate through the array from index 1 to n-1:

* If arr[i] > largest, then set largest = arr[i].

 **Output**: Print the largest number.

 **End**

### Pseudocode:

Start

Input n

Declare array arr[n]

For i = 0 to n-1 do

Input arr[i]

End For

largest = arr[0]

For i = 1 to n-1 do

If arr[i] > largest then

largest = arr[i]

End If

End For

Output largest

End

### Flowchart:

Program:

#include <stdio.h>

int main() {

int n;

// Input: Get the number of elements in the array

printf("Enter the number of elements: ");

scanf("%d", &n);

int arr[n];

// Input: Get the elements of the array

printf("Enter %d elements: \n", n);

for (int i = 0; i < n; i++) {

scanf("%d", &arr[i]);

}

// Initialize largest to the first element of the array

int largest = arr[0];

// Loop through the array to find the largest number

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

if (arr[i] > largest) {

largest = arr[i];

}

}

// Output: Display the largest number

printf("The largest number in the array is: %d\n", largest);

return 0;

}

5

Algorithm:

 **Start**

 **Input**: Read the values of P (Principal), R (Rate of interest), and T (Time in years).

 **Calculate**: Use the formula SI = (P × R × T) / 100 to compute the simple interest.

 **Output**: Display the calculated simple interest (SI).

 **End**

Pseudocode:

BEGIN

Declare variables P, R, T, SI

Print "Enter Principal Amount (P):"

Input P

Print "Enter Rate of Interest (R):"

Input R

Print "Enter Time in Years (T):"

Input T

SI ← (P \* R \* T) / 100

Print "The Simple Interest is: ", SI

END

Flowchart:

#include <stdio.h>

int main() {

float P, R, T, SI;

// Input values

printf("Enter Principal Amount (P): ");

scanf("%f", &P);

printf("Enter Rate of Interest (R): ");

scanf("%f", &R);

printf("Enter Time in Years (T): ");

scanf("%f", &T);

// Calculate Simple Interest

SI = (P \* R \* T) / 100;

// Display the result

printf("The Simple Interest is: %.2f\n", SI);

return 0;

}

6

Algorithm:

 **Start**

 Declare variables: n (number of terms), a (first term), b (second term), next (next term in the sequence), and i (loop counter).

 Input the value of n.

 If n is less than or equal to 0, print an error message and stop.

 If n is 1, print a and stop.

 If n is 2, print a and b and stop.

 Print a and b.

 Use a loop to calculate the next terms in the Fibonacci sequence:

* next = a + b
* Print next
* Update a = b and b = next

 Repeat until all n terms are printed.

 **Stop**

**Pseudocode:**

BEGIN

Declare n, a, b, next, i

a ← 0

b ← 1

Print "Enter the number of terms (n):"

Input n

IF n <= 0 THEN

Print "Invalid input! Enter a positive number."

STOP

ENDIF

IF n == 1 THEN

Print a

STOP

ENDIF

Print a, b

FOR i FROM 3 TO n DO

next ← a + b

Print next

a ← b

b ← next

END FOR

END

Flowchart:

Program:

#include <stdio.h>

int main() {

int n, a = 0, b = 1, next, i;

// Input the number of terms

printf("Enter the number of terms: ");

scanf("%d", &n);

if (n <= 0) {

printf("Invalid input! Enter a positive number.\n");

return 0;

}

if (n == 1) {

printf("Fibonacci Series: %d\n", a);

return 0;

}

printf("Fibonacci Series: %d, %d", a, b);

for (i = 3; i <= n; i++) {

next = a + b;

printf(", %d", next);

a = b;

b = next;

}

printf("\n");

return 0;

}

7

 **Start**

 Generate a random number between 1 and 100 and store it in randomNumber.

 Declare a variable guess to store the user's input and attempts to count the number of guesses.

 Initialize attempts to 0.

 Repeat the following steps until the user guesses the number:

* Increment attempts by 1.
* Ask the user to input a guess and store it in guess.
* If guess is equal to randomNumber, display "Congratulations!" and stop.
* If guess is less than randomNumber, display "Higher!".
* If guess is greater than randomNumber, display "Lower!".

 Display the total number of attempts.

 **Stop**

Pseudocode:

BEGIN

Import random library

Generate randomNumber between 1 and 100

Declare guess, attempts

attempts ← 0

REPEAT

Increment attempts by 1

Print "Enter your guess:"

Input guess

IF guess = randomNumber THEN

Print "Congratulations! You've guessed the correct number."

Print "Total attempts: ", attempts

EXIT

ELSE IF guess < randomNumber THEN

Print "Higher!"

ELSE

Print "Lower!"

ENDIF

UNTIL guess = randomNumber

END

Flowchart:

Program:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int main() {

int randomNumber, guess, attempts = 0;

// Initialize random seed

srand(time(0));

randomNumber = (rand() % 100) + 1; // Generate a random number between 1 and 100

printf("Welcome to the Number Guessing Game!\n");

printf("I have chosen a number between 1 and 100. Try to guess it!\n");

do {

printf("Enter your guess: ");

scanf("%d", &guess);

attempts++;

if (guess == randomNumber) {

printf("Congratulations! You guessed the correct number in %d attempts.\n", attempts);

} else if (guess < randomNumber) {

printf("Higher!\n");

} else {

printf("Lower!\n");

}

} while (guess != randomNumber);

return 0;

}

8

Algorithm:

 **Start**

 Declare variables: C (Celsius temperature) and F (Fahrenheit temperature).

 Input the value of C.

 Convert the Celsius temperature to Fahrenheit using the formula: F=(C×95)+32F = (C \times \frac{9}{5}) + 32F=(C×59​)+32

 Display the Fahrenheit temperature F.

 **Stop**

Pseudocode:

BEGIN

Declare variables C, F

Print "Enter temperature in Celsius:"

Input C

F ← (C × 9/5) + 32

Print "Temperature in Fahrenheit is: ", F

END

Flowchart:

Program:

#include <stdio.h>

int main() {

float celsius, fahrenheit;

// Input the Celsius temperature

printf("Enter temperature in Celsius: ");

scanf("%f", &celsius);

// Convert Celsius to Fahrenheit

fahrenheit = (celsius \* 9 / 5) + 32;

// Display the result

printf("Temperature in Fahrenheit:

return 0;

}

9

Algorithm:

 **Start**

 Declare a string variable str and an integer variable vowelCount initialized to 0.

 Input the string str from the user.

 Loop through each character of the string:

* Check if the character is a vowel (a, e, i, o, u or A, E, I, O, U).
* If it is a vowel, increment vowelCount by 1.

 After the loop ends, display the value of vowelCount.

 **Stop**

**Pseudocode:**

BEGIN

Declare string str

Declare integer vowelCount ← 0

Print "Enter a string:"

Input str

FOR each character in str DO

IF character is 'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U' THEN

Increment vowelCount by 1

ENDIF

END FOR

Print "Number of vowels:", vowelCount

END

Flowchart:

Program:

#include <stdio.h>

#include <string.h>

int main() {

char str[100];

int i, vowelCount = 0;

// Input the string

printf("Enter a string: ");

fgets(str, sizeof(str), stdin);

// Count vowels

for (i = 0; str[i] != '\0'; i++) {

char ch = str[i];

if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||

ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {

vowelCount++;

}

}

// Display the result

printf("Number of vowels: %d\n", vowelCount);

return 0;}

10

Algorithm:

 **Start**

 Input two integers a and b.

 While b is not equal to 0:

* Compute the remainder of a divided by b (remainder = a % b).
* Assign the value of b to a.
* Assign the value of remainder to b.

 When b becomes 0, the value of a is the GCD.

 Output the GCD.

 **Stop**

Pseudocode:

BEGIN

Declare a, b, remainder

Print "Enter two numbers:"

Input a, b

WHILE b ≠ 0 DO

remainder ← a % b

a ← b

b ← remainder

END WHILE

Print "GCD is:", a

END

Flowchart:

Program:

#include <stdio.h>

int main() {

int a, b, remainder;

// Input two numbers

printf("Enter two numbers: ");

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

// Ensure positive numbers

if (a < 0) a = -a;

if (b < 0) b = -b;

// Euclidean algorithm to find GCD

while (b != 0) {

remainder = a % b;

a = b;

b = remainder;

}

// Output the GCD

printf("GCD is: %d\n", a);

return 0;

}