KPIT Technologies

WEEKLY REPORT

Week 1 Report

|  |  |  |  |
| --- | --- | --- | --- |
| Student Name: Dayas A Dixen | Week 1 | Branch EEE | USN  :1NH20EE029 |

t(Date:24-05-2024)

1. Write a C program to print "Hello, World!"?

#include <stdio.h>

int main() {

printf("Hello, World!\n");

return 0;

}

1. Write a C program to add two numbers.

#include <stdio.h>

int main() {

int a, b, sum;

printf("Enter two integers: ");

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

sum = a + b;

printf("Sum: %d\n", sum);

return 0;

}

1. Write a C program to swap two numbers without using a temporary variable.

#include <stdio.h>

int main() {

int a, b;

printf("Enter two integers: ");

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

a = a + b;

b = a - b;

a = a - b;

printf("After swapping: a = %d, b = %d\n", a, b);

return 0;

}

1. Write a C program to check if a number is even or odd.

#include <stdio.h>

int main() {

int num;

printf("Enter an integer: ");

scanf("%d", &num);

if (num % 2 == 0) {

printf("%d is even.\n", num);

} else {

printf("%d is odd.\n", num);

}

return 0;

}

1. Write a C program to find the largest of three numbers.

#include <stdio.h>

int main() {

int a, b, c;

printf("Enter three integers: ");

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

if (a >= b && a >= c) {

printf("%d is the largest.\n", a);

} else if (b >= a && b >= c) {

printf("%d is the largest.\n", b);

} else {

printf("%d is the largest.\n", c);

}

return 0;

}

1. Write a C program to find the sum of digits of a number.

#include <stdio.h>

int main() {

int num, sum = 0;

printf("Enter an integer: ");

scanf("%d", &num);

while (num != 0) {

sum += num % 10;

num /= 10;

}

printf("Sum of digits: %d\n", sum);

return 0;

}

1. Write a C program to reverse a number.

#include <stdio.h>

int main() {

int num, reverse = 0;

printf("Enter an integer: ");

scanf("%d", &num);

while (num != 0) {

reverse = reverse \* 10 + num % 10;

num /= 10;

}

printf("Reversed number: %d\n", reverse);

return 0;

}

1. Write a C program to check if a number is prime.

#include <stdio.h>

int main() {

int num, i, flag = 0;

printf("Enter an integer: ");

scanf("%d", &num);

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

if (num % i == 0) {

flag = 1;

break;

}

}

if (num == 1) {

printf("1 is neither prime nor composite.\n");

} else {

if (flag == 0) {

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

} else {

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

}

}

return 0;

}

1. Write a C program to print the Fibonacci series up to n terms.

#include <stdio.h>

int main() {

int n, t1 = 0, t2 = 1, nextTerm;

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

scanf("%d", &n);

printf("Fibonacci Series: ");

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

printf("%d, ", t1);

nextTerm = t1 + t2;

t1 = t2;

t2 = nextTerm;

}

return 0;

}

1. Write a C program to find the factorial of a number.

#include <stdio.h>

int main() {

int n;

unsigned long long factorial = 1;

printf("Enter an integer: ");

scanf("%d", &n);

if (n < 0) {

printf("Error! Factorial of a negative number doesn't exist.\n");

} else {

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

factorial \*= i;

}

printf("Factorial of %d = %llu\n", n, factorial);

}

return 0;

}

1. Write a C program to check if a number is an Armstrong number.

#include <stdio.h>

#include <math.h>

int main() {

int num, originalNum, remainder, n = 0;

float result = 0.0;

printf("Enter an integer: ");

scanf("%d", &num);

originalNum = num;

while (originalNum != 0) {

originalNum /= 10;

++n;

}

originalNum = num;

while (originalNum != 0) {

remainder = originalNum % 10;

result += pow(remainder, n);

originalNum /= 10;

}

if ((int)result == num) {

printf("%d is an Armstrong number.\n", num);

} else {

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

}

return 0;

}

1. Write a C program to check if a number is a palindrome.

#include <stdio.h>

int main() {

int num, reversedNum = 0, remainder, originalNum;

printf("Enter an integer: ");

scanf("%d", &num);

originalNum = num;

while (num != 0) {

remainder = num % 10;

reversedNum = reversedNum \* 10 + remainder;

num /= 10;

}

if (originalNum == reversedNum) {

printf("%d is a palindrome.\n", originalNum);

} else {

printf("%d is not a palindrome.\n", originalNum);

}

return 0;

}

1. Write a C program to calculate the power of a number using a loop.

#include <stdio.h>

int main() {

int base, exp;

long long result = 1;

printf("Enter base and exponent: ");

scanf("%d %d", &base, &exp);

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

result \*= base;

}

printf("%d^%d = %lld\n", base, exp, result);

return 0;

}

1. Write a C program to create a simple calculator (add, subtract, multiply, divide).

#include <stdio.h>

int main() {

char operator;

double first, second;

printf("Enter an operator (+, -, \*, /): ");

scanf("%c", &operator);

printf("Enter two operands: ");

scanf("%lf %lf", &first, &second);

switch (operator) {

case '+':

printf("%.1lf + %.1lf = %.1lf\n", first, second, first + second);

break;

case '-':

printf("%.1lf - %.1lf = %.1lf\n", first, second, first - second);

break;

case '\*':

printf("%.1lf \* %.1lf = %.1lf\n", first, second, first \* second);

break;

case '/':

if (second != 0.0)

printf("%.1lf / %.1lf = %.1lf\n", first, second, first / second);

else

printf("Error! Division by zero.\n");

break;

default:

printf("Error! Operator is not correct.\n");

break;

}

return 0;

}

1. Write a C program to find the Greatest Common Divisor (GCD) of two numbers.

#include <stdio.h>

int main() {

int n1, n2;

printf("Enter two integers: ");

scanf("%d %d", &n1, &n2);

while (n1 != n2) {

if (n1 > n2)

n1 -= n2;

else

n2 -= n1;

}

printf("GCD = %d\n", n1);

return 0;

}

1. Write a C program to find the Least Common Multiple (LCM) of two numbers.

#include <stdio.h>

int main() {

int n1, n2, max;

printf("Enter two integers: ");

scanf("%d %d", &n1, &n2);

max = (n1 > n2) ? n1 : n2;

while (1) {

if (max % n1 == 0 && max % n2 == 0) {

printf("LCM = %d\n", max);

break;

}

++max;

}

return 0;

}

1. Write a C program to display the ASCII value of a character.

#include <stdio.h>

int main() {

char c;

printf("Enter a character: ");

scanf("%c", &c);

printf("ASCII value of %c = %d\n", c, c);

return 0;

}

1. Write a C program to print all the alphabets.

#include <stdio.h>

int main() {

char c;

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

printf("%c ", c);

}

printf("\n");

return 0;

}

1. Write a C program to count the number of digits in an integer.

#include <stdio.h>

int main() {

long long num;

int count = 0;

printf("Enter an integer: ");

scanf("%lld", &num);

while (num != 0) {

num /= 10;

++count;

}

printf("Number of digits: %d\n", count);

return 0;

}

1. Write a C program to reverse a string.

#include <stdio.h>

#include <string.h>

int main() {

char str[100], temp;

int i, j;

printf("Enter a string: ");

gets(str);

j = strlen(str) - 1;

for (i = 0; i < j; i++, j--) {

temp = str[i];

str[i] = str[j];

str[j] = temp;

}

printf("Reversed string: %s\n", str);

return 0;

}

1. Write a C program to count the number of vowels and consonants in a string.

#include <stdio.h>

#include <ctype.h>

int main() {

char str[100];

int vowels = 0, consonants = 0;

printf("Enter a string: ");

gets(str);

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

if (toupper(str[i]) == 'A' || toupper(str[i]) == 'E' || toupper(str[i]) == 'I' || toupper(str[i]) == 'O' || toupper(str[i]) == 'U') {

++vowels;

} else if ((str[i] >= 'a' && str[i] <= 'z') || (str[i] >= 'A' && str[i] <= 'Z')) {

++consonants;

}

}

printf("Vowels: %d\n", vowels);

printf("Consonants: %d\n", consonants);

return 0;

}

1. Write a C program to check if a year is a leap year.

#include <stdio.h>

int main() {

int year;

printf("Enter a year: ");

scanf("%d", &year);

if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {

printf("%d is a leap year.\n", year);

} else {

printf("%d is not a leap year.\n", year);

}

return 0;

}

1. Write a C program to check if a string is a palindrome.

#include <stdio.h>

#include <string.h>

int main() {

char str[100], reversedStr[100];

printf("Enter a string: ");

gets(str);

strcpy(reversedStr, str);

strrev(reversedStr);

if (strcmp(str, reversedStr) == 0) {

printf("%s is a palindrome.\n", str);

} else {

printf("%s is not a palindrome.\n", str);

}

return 0;

}

1. Write a C program to find the length of a string.

#include <stdio.h>

#include <string.h>

int main() {

char str[100];

printf("Enter a string: ");

gets(str);

printf("Length of the string: %d\n", strlen(str));

return 0;

}

1. Write a C program to concatenate two strings

#include <stdio.h>

#include <string.h>

int main() {

char str1[100], str2[100];

printf("Enter first string: ");

gets(str1);

printf("Enter second string: ");

gets(str2);

strcat(str1, str2);

printf("Concatenated string: %s\n", str1);

return 0;

}

1. Write a C program to copy one string to another.

#include <stdio.h>

#include <string.h>

int main() {

char str1[100], str2[100];

printf("Enter a string: ");

gets(str1);

strcpy(str2, str1);

printf("Copied string: %s\n", str2);

return 0;

}

1. Write a C program to find the maximum and minimum elements in an array.

#include <stdio.h>

int main() {

int n, i;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

int max = arr[0], min = arr[0];

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

if (arr[i] > max) {

max = arr[i];

}

if (arr[i] < min) {

min = arr[i];

}

}

printf("Maximum element: %d\n", max);

printf("Minimum element: %d\n", min);

return 0;

}

1. Write a C program to find the sum of all elements in an array.

#include <stdio.h>

int main() {

int n, sum = 0;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

sum += arr[i];

}

printf("Sum of elements: %d\n", sum);

return 0;

}

1. Write a C program to reverse an array.

#include <stdio.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

printf("Reversed array: ");

for (int i = n - 1; i >= 0; --i) {

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

1. Write a C program to sort an array using bubble sort.

#include <stdio.h>

void bubbleSort(int arr[], int n) {

int i, j, temp;

for (i = 0; i < n - 1; ++i) {

for (j = 0; j < n - i - 1; ++j) {

if (arr[j] > arr[j + 1]) {

temp = arr[j];

arr[j] = arr[j + 1];

arr[j + 1] = temp;

}

}

}

}

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

bubbleSort(arr, n);

printf("Sorted array: ");

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

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

1. Write a C program to perform a linear search on an array.

#include <stdio.h>

int main() {

int n, key, found = 0;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

printf("Enter the element to search: ");

scanf("%d", &key);

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

if (arr[i] == key) {

printf("%d found at position %d\n", key, i + 1);

found = 1;

break;

}

}

if (!found) {

printf("%d not found in the array.\n", key);

}

return 0;

}

1. Write a C program to perform a binary search on a sorted array.

#include <stdio.h>

int binarySearch(int arr[], int low, int high, int key) {

while (low <= high) {

int mid = low + (high - low) / 2;

if (arr[mid] == key) {

return mid;

}

if (arr[mid] < key) {

low = mid + 1;

} else {

high = mid - 1;

}

}

return -1;

}

int main() {

int n, key;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements in sorted order: ");

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

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

}

printf("Enter the element to search: ");

scanf("%d", &key);

int result = binarySearch(arr, 0, n - 1, key);

if (result != -1) {

printf("%d found at position %d\n", key, result + 1);

} else {

printf("%d not found in the array.\n", key);

}

return 0;

}

1. Write a C program to add two matrices.

#include <stdio.h>

int main() {

int r, c;

printf("Enter the number of rows and columns: ");

scanf("%d %d", &r, &c);

int a[r][c], b[r][c], sum[r][c];

printf("Enter elements of first matrix:\n");

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

for (int j = 0; j < c; ++j) {

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

}

}

printf("Enter elements of second matrix:\n");

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

for (int j = 0; j < c; ++j) {

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

}

}

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

for (int j = 0; j < c; ++j) {

sum[i][j] = a[i][j] + b[i][j];

}

}

printf("Sum of matrices:\n");

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

for (int j = 0; j < c; ++j) {

printf("%d ", sum[i][j]);

}

printf("\n");

}

return 0;

}

1. Write a C program to multiply two matrices.

#include <stdio.h>

int main() {

int r1, c1, r2, c2;

printf("Enter rows and columns for first matrix: ");

scanf("%d %d", &r1, &c1);

printf("Enter rows and columns for second matrix: ");

scanf("%d %d", &r2, &c2);

if (c1 != r2) {

printf("Error! Column of first matrix not equal to row of second matrix.\n");

return -1;

}

int a[r1][c1], b[r2][c2], product[r1][c2];

printf("Enter elements of first matrix:\n");

for (int i = 0; i < r1; ++i) {

for (int j = 0; j < c1; ++j) {

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

}

}

printf("Enter elements of second matrix:\n");

for (int i = 0; i < r2; ++i) {

for (int j = 0; j < c2; ++j) {

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

}

}

for (int i = 0; i < r1; ++i) {

for (int j = 0; j < c2; ++j) {

product[i][j] = 0;

for (int k = 0; k < c1; ++k) {

product[i][j] += a[i][k] \* b[k][j];

}

}

}

printf("Product of matrices:\n");

for (int i = 0; i < r1; ++i) {

for (int j = 0; j < c2; ++j) {

printf("%d ", product[i][j]);

}

printf("\n");

}

return 0;

}

1. Write a C program to find the transpose of a matrix.

#include <stdio.h>

int main() {

int r, c;

printf("Enter rows and columns of matrix: ");

scanf("%d %d", &r, &c);

int matrix[r][c], transpose[c][r];

printf("Enter elements of matrix:\n");

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

for (int j = 0; j < c; ++j) {

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

}

}

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

for (int j = 0; j < c; ++j) {

transpose[j][i] = matrix[i][j];

}

}

printf("Transpose of matrix:\n");

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

for (int j = 0; j < r; ++j) {

printf("%d ", transpose[i][j]);

}

printf("\n");

}

return 0;

}

1. Write a C program to find the sum of diagonals in a matrix.

#include <stdio.h>

int main() {

int n, sum1 = 0, sum2 = 0;

printf("Enter the size of matrix: ");

scanf("%d", &n);

int matrix[n][n];

printf("Enter elements of matrix:\n");

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

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

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

if (i == j) {

sum1 += matrix[i][j];

}

if (i + j == n - 1) {

sum2 += matrix[i][j];

}

}

}

printf("Sum of primary diagonal: %d\n", sum1);

printf("Sum of secondary diagonal: %d\n", sum2);

return 0;

}

1. Write a C program to find the factorial of a number using recursion.

#include <stdio.h>

unsigned long long factorial(int n) {

if (n == 0) {

return 1;

} else {

return n \* factorial(n - 1);

}

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

printf("Factorial of %d = %llu\n", num, factorial(num));

return 0;

}

1. Write a C program to display the Fibonacci series using recursion.

#include <stdio.h>

int fibonacci(int n) {

if (n == 0) {

return 0;

} else if (n == 1) {

return 1;

} else {

return (fibonacci(n - 1) + fibonacci(n - 2));

}

}

int main() {

int n;

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

scanf("%d", &n);

printf("Fibonacci Series: ");

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

printf("%d ", fibonacci(i));

}

printf("\n");

return 0;

}

1. Write a C program to print all prime numbers between a given range using recursion.

#include <stdio.h>

int isPrime(int n, int i) {

if (n <= 2) {

return (n == 2) ? 1 : 0;

}

if (n % i == 0) {

return 0;

}

if (i \* i > n) {

return 1;

}

return isPrime(n, i + 1);

}

void printPrimes(int start, int end) {

if (start > end) {

return;

}

if (isPrime(start, 2)) {

printf("%d ", start);

}

printPrimes(start + 1, end);

}

int main() {

int start, end;

printf("Enter the range (start and end): ");

scanf("%d %d", &start, &end);

printf("Prime numbers between %d and %d: ", start, end);

printPrimes(start, end);

printf("\n");

return 0;

}

1. Write a C program to check if a number is a palindrome using recursion

#include <stdio.h>

#include <string.h>

int isPalindrome(char str[], int start, int end) {

if (start >= end) {

return 1;

}

if (str[start] != str[end]) {

return 0;

}

return isPalindrome(str, start + 1, end - 1);

}

int main() {

char str[100];

printf("Enter a number: ");

gets(str);

int len = strlen(str);

if (isPalindrome(str, 0, len - 1)) {

printf("%s is a palindrome.\n", str);

} else {

printf("%s is not a palindrome.\n", str);

}

return 0;

}

1. Write a C program to find the power of a number using recursion.

#include <stdio.h>

int power(int base, int exp) {

if (exp == 0) {

return 1;

} else {

return base \* power(base, exp - 1);

}

}

int main() {

int base, exp;

printf("Enter base and exponent: ");

scanf("%d %d", &base, &exp);

printf("%d^%d = %d\n", base, exp, power(base, exp));

return 0;

}

1. Write a C program to check if a number is an Armstrong number.

#include <stdio.h>

#include <math.h>

int main() {

int num, originalNum, remainder, n = 0;

float result = 0.0;

printf("Enter an integer: ");

scanf("%d", &num);

originalNum = num;

while (originalNum != 0) {

originalNum /= 10;

++n;

}

originalNum = num;

while (originalNum != 0) {

remainder = originalNum % 10;

result += pow(remainder, n);

originalNum /= 10;

}

if ((int)result == num) {

printf("%d is an Armstrong number.\n", num);

} else {

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

}

return 0;

}

1. Write a C program to find the sum of digits of a number using recursion.

#include <stdio.h>

int sumOfDigits(int n) {

if (n == 0) {

return 0;

} else {

return (n % 10) + sumOfDigits(n / 10);

}

}

int main() {

int num;

printf("Enter a number: ");

scanf("%d", &num);

printf("Sum of digits: %d\n", sumOfDigits(num));

return 0;

}

1. Write a C program to convert a decimal number to binary.

#include <stdio.h>

void decimalToBinary(int n) {

if (n > 1) {

decimalToBinary(n / 2);

}

printf("%d", n % 2);

}

int main() {

int num;

printf("Enter a decimal number: ");

scanf("%d", &num);

printf("Binary representation: ");

decimalToBinary(num);

printf("\n");

return 0;

}

1. Write a C program to convert a binary number to decimal.

#include <stdio.h>

#include <math.h>

int binaryToDecimal(int n) {

int decimal = 0, i = 0, remainder;

while (n != 0) {

remainder = n % 10;

n /= 10;

decimal += remainder \* pow(2, i);

++i;

}

return decimal;

}

int main() {

int num;

printf("Enter a binary number: ");

scanf("%d", &num);

printf("Decimal representation: %d\n", binaryToDecimal(num));

return 0;

}

1. Write a C program to count the occurrences of a character in a string.

#include <stdio.h>

int main() {

char str[100], ch;

int count = 0;

printf("Enter a string: ");

gets(str);

printf("Enter a character to find its frequency: ");

scanf("%c", &ch);

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

if (str[i] == ch) {

++count;

}

}

printf("Frequency of %c = %d\n", ch, count);

return 0;

}

1. Write a C program to remove all whitespaces from a string.

#include <stdio.h>

int main() {

char str[100], newStr[100];

int j = 0;

printf("Enter a string: ");

gets(str);

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

if (str[i] != ' ') {

newStr[j++] = str[i];

}

}

newStr[j] = '\0';

printf("String without whitespaces: %s\n", newStr);

return 0;

}

1. Write a C program to find the second largest element in an array.

#include <stdio.h>

int main() {

int n;

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

scanf("%d", &n);

int arr[n];

printf("Enter the elements: ");

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

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

}

int first, second;

if (arr[0] > arr[1]) {

first = arr[0];

second = arr[1];

} else {

first = arr[1];

second = arr[0];

}

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

if (arr[i] > first) {

second = first;

first = arr[i];

} else if (arr[i] > second && arr[i] != first) {

second = arr[i];

}

}

printf("Second largest element: %d\n", second);

return 0;

}

1. Write a C program to check if a number is a perfect number.

#include <stdio.h>

int main() {

int num, sum = 0;

printf("Enter a number: ");

scanf("%d", &num);

for (int i = 1; i <= num / 2; ++i) {

if (num % i == 0) {

sum += i;

}

}

if (sum == num) {

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

} else {

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

}

return 0;

}

1. Write a C program to find the sum of elements above the main diagonal of a matrix.

#include <stdio.h>

int main() {

int n, sum = 0;

printf("Enter the size of matrix: ");

scanf("%d", &n);

int matrix[n][n];

printf("Enter elements of matrix:\n");

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

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

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

if (i < j) {

sum += matrix[i][j];

}

}

}

printf("Sum of elements above the main diagonal: %d\n", sum);

return 0;

}