**RAAHAT ARORA**

**230957216**

**Roll No – 72**

**WEEK 3**

**Question 1 –**

#include <iostream>

using namespace std;

// Recursive function to determine the number of binary digits

int countBinaryDigits(int n) {

// Base case: If n is 0, return 0 because it has no binary digits

if (n == 0) {

return 0;

}

// Recursive case: count digits in n / 2, and add 1 for the current digit

return 1 + countBinaryDigits(n / 2);

}

int main() {

int n;

// Input: a positive integer n

cout << "Enter a positive integer: ";

cin >> n;

// Edge case: if n is 0, the binary representation is 0 and has 1 bit.

if (n == 0) {

cout << "The number of binary digits in " << n << " is: 1" << endl;

} else {

// Call the recursive function and output the result

int result = countBinaryDigits(n);

cout << "The number of binary digits in " << n << " is: " << result << endl;

}

return 0;

}

**Output-**

****

**Question 2-**

#include <iostream>

using namespace std;

int findUniqueElement(int arr[], int size) {

int uniqueElement = 0;

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

uniqueElement ^= arr[i];

}

return uniqueElement;

}

int main() {

int arr[] = {4, 3, 2, 4, 1, 3, 2};

int size = sizeof(arr) / sizeof(arr[0]);

int uniqueElement = findUniqueElement(arr, size);

cout << "The unique element in the array is: " << uniqueElement << endl;

return 0;

}

**Output-**

****

**Question 3-**

#include <iostream>

using namespace std;

void multiplyMatrices(int A[][10], int B[][10], int C[][10], int n) {

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

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

C[i][j] = 0;

}

}

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

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

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

C[i][j] += A[i][k] \* B[k][j];

}

}

}

}

void printMatrix(int matrix[][10], int n) {

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

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

cout << matrix[i][j] << " ";

}

cout << endl;

}

}

int main() {

int n = 3;

int A[10][10] = {{1, 2, 3},

{4, 5, 6},

{7, 8, 9}};

int B[10][10] = {{9, 8, 7},

{6, 5, 4},

{3, 2, 1}};

int C[10][10];

multiplyMatrices(A, B, C, n);

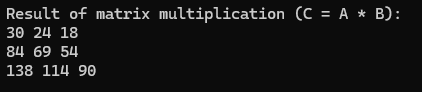
cout << "Result of matrix multiplication (C = A \* B):" << endl;

printMatrix(C, n);

return 0;

}

**Output-**



**Question 4-**

#include <iostream>

using namespace std;

int countBinaryDigits(int n) {

int count = 0;

while (n > 0) {

n = n / 2;

count++;

}

return count;

}

int main() {

int n;

cout << "Enter a positive integer: ";

cin >> n;

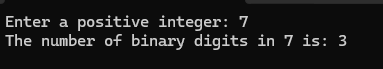
int binaryDigits = countBinaryDigits(n);

cout << "The number of binary digits in " << n << " is: " << binaryDigits << endl;

return 0;

}

**Output-**



**Question 5-**

#include <iostream>

using namespace std;

int factorial(int n) {

if (n == 0) {

return 1;

}

else {

return n \* factorial(n - 1);

}

}

int main() {

int n;

cout << "Enter a non-negative integer: ";

cin >> n;

int result = factorial(n);

cout << "The factorial of " << n << " is: " << result << endl;

return 0;

}

**Output-**

****

**Question 6-**

#include <iostream>

using namespace std;

void towerOfHanoi(int n, char source, char auxiliary, char destination) {

if (n == 1) {

cout << "Move disk 1 from " << source << " to " << destination << endl;

return;

}

towerOfHanoi(n - 1, source, destination, auxiliary);

cout << "Move disk " << n << " from " << source << " to " << destination << endl;

towerOfHanoi(n - 1, auxiliary, source, destination);

}

int main() {

int n;

cout << "Enter the number of disks: ";

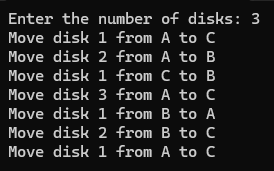
cin >> n;

towerOfHanoi(n, 'A', 'B', 'C');

return 0;

}

**Output-**

****

**Question 7-**

#include <iostream>

using namespace std;

int countBinaryDigits(int n) {

if (n == 0) {

return 0;

}

return 1 + countBinaryDigits(n / 2);

}

int main() {

int n;

cout << "Enter a positive integer: ";

cin >> n;

if (n == 0) {

cout << "The number of binary digits in " << n << " is: 1" << endl;

} else {

int result = countBinaryDigits(n);

cout << "The number of binary digits in " << n << " is: " << result << endl;

}

return 0;

}

**Output-**

****