Q2)#include <iostream>

using namespace std;

int factorial(int n) {

if (n == 0) {

return 1;

} else {

return n \* factorial(n-1);

}

}

int power(int x, int y) {

int result = 1;

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

result \*= x;

}

return result;

}

int fibonacci(int n) {

if (n <= 1) {

return n;

} else {

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

}

}

int multiplication(int x, int y) {

return x \* y;

}

int main() {

int choice;

cout << "کدام تابع را می‌خواهید اجرا کنید؟ (1: factorial, 2: power, 3: fibonacci, 4: multiplication): ";

cin >> choice;

switch (choice) {

case 1:

int num1;

cout << "عدد مورد نظر را وارد کنید: ";

cin >> num1;

cout << "حاصل عدد فاکتوریل " << num1 << " برابر است با: " << factorial(num1) << endl;

break;

case 2:

int base, exponent;

cout << "پایه را وارد کنید: ";

cin >> base;

cout << "توان را وارد کنید: ";

cin >> exponent;

cout << "حاصل عملیات " << base << " به توان " << exponent << " برابر است با: " << power(base, exponent) << endl;

break;

case 3:

int num2;

cout << "تعداد عناصر دنباله فیبوناچی را وارد کنید: ";

cin >> num2;

cout << "سری فیبوناچی:" << endl;

for (int i = 0; i < num2; i++) {

cout << fibonacci(i) << " ";

}

cout << endl;

break;

case 4:

int num3, num4;

cout << "عدد اول را وارد کنید: ";

cin >> num3;

cout << "عدد دوم را وارد کنید: ";

cin >> num4;

cout << "حاصل ضرب " << num3 << " در " << num4 << " برابر است با: " << multiplication(num3, num4) << endl;

break;

default:

cout << "ورودی نامعتبر!" << endl;

}

return 0;

}

Q3)#include <iostream>

using namespace std;

int main() {

const int SIZE = 9;

int input;

int counts[SIZE] = {0};

while (true) {

cout << "لطفا یک عدد بین ۱ تا ۹ وارد کنید (برای خروج 0 را وارد کنید): ";

cin >> input;

if (input == 0) {

break;

}

if (input >= 1 && input <= 9) {

counts[input-1]++;

} else {

cout << "ورودی نامعتبر!" << endl;

}

}

int maxIndex = 0;

int minIndex = 0;

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

if (counts[i] > counts[maxIndex]) {

maxIndex = i;

}

if (counts[i] < counts[minIndex]) {

minIndex = i;

}

}

cout << "تعداد تکرار هر عدد:" << endl;

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

cout << "عدد " << (i+1) << ": " << counts[i] << " بار" << endl;

}

cout << "بیشترین تکرار: عدد " << (maxIndex+1) << " با " << counts[maxIndex] << " بار" << endl;

cout << "کمترین تکرار: عدد " << (minIndex+1) << " با " << counts[minIndex] << " بار" << endl;

return 0;

}

Q4)#include <iostream>

#include <cmath>

using namespace std;

struct Point3D {

double x;

double y;

double z;

};

double distance(Point3D p1, Point3D p2) {

return sqrt(pow(p2.x - p1.x, 2) + pow(p2.y - p1.y, 2) + pow(p2.z - p1.z, 2));

}

bool areOnSamePlane(Point3D p1, Point3D p2) {

return p1.z == p2.z;

}

int main() {

Point3D point1, point2;

cout << "Enter coordinates of the first point (x y z): ";

cin >> point1.x >> point1.y >> point1.z;

cout << "Enter coordinates of the second point (x y z): ";

cin >> point2.x >> point2.y >> point2.z;

cout << "The distance between the two points is: " << distance(point1, point2) << endl;

if (areOnSamePlane(point1, point2)) {

cout << "The two points are on the same plane." << endl;

} else {

cout << "The two points are not on the same plane." << endl;

}

return 0;

}

Q5)#include <iostream>

#include <algorithm>

using namespace std;

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

cout << "Enter " << n << " elements: ";

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

cin >> arr[i];

}

}

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

cout << "Array elements: ";

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

cout << arr[i] << " ";

}

cout << endl;

}

int findMax(int arr[], int n) {

int max = arr[0];

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

if (arr[i] > max) {

max = arr[i];

}

}

return max;

}

int findMin(int arr[], int n) {

int min = arr[0];

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

if (arr[i] < min) {

min = arr[i];

}

}

return min;

}

int calculateSum(int arr[], int n) {

int sum = 0;

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

sum += arr[i];

}

return sum;

}

double calculateAverage(int arr[], int n) {

return static\_cast<double>(calculateSum(arr, n)) / n;

}

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

sort(arr, arr + n);

}

int searchValue(int arr[], int n, int value) {

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

if (arr[i] == value) {

return i;

}

}

return -1;

}

int main() {

int n;

cout << "Enter the size of the array: ";

cin >> n;

int arr[n];

inputArray(arr, n);

printArray(arr, n);

cout << "Maximum element: " << findMax(arr, n) << endl;

cout << "Minimum element: " << findMin(arr, n) << endl;

cout << "Sum of elements: " << calculateSum(arr, n) << endl;

cout << "Average of elements: " << calculateAverage(arr, n) << endl;

int sortedArr[n];

copy(arr, arr + n, sortedArr);

sortArray(sortedArr, n);

cout << "Sorted array: ";

printArray(sortedArr, n);

int searchValue;

cout << "Enter a value to search in the array: ";

cin >> searchValue;

int index = searchValue(arr, n, searchValue);

if (index != -1) {

cout << "Value found at index: " << index << endl;

} else {

cout << "Value not found in the array." << endl;

}

return 0;

}