1.

#include <iostream>

using namespace std;

bool isprime(int num){

if(num<2) return false;

for(int i=2;i\*i<=num;i++){

if(num%i==0) return false;

}

return true;

}

int nextprime(int num){

int next =num+1;

while(!isprime(next)){

next++;

}

return next;

}

void printfactors(int num){

cout<<"factors of "<<num<<" are";

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

if(num%i==0){

cout<<i<<"";

}

}

cout<<endl;

}

int main() {

int n;

cout<<"enter the integer:";

cin>>n;

if(isprime(n)){

cout <<"is a prime no";

cout<<"the next prime no is"<<nextprime(n)<<endl;

}

else{

cout<<"is not a prime no.";

printfactors(n);

}

return 0;

}

OUTPUT:

enter the integer:10

is not a prime no.factors of 10 are12510

2.

#include <iostream>

#include <algorithm> // For sorting

using namespace std;

void reverseArray(int arr[], int size) {

for (int i = 0; i < size / 2; i++) {

swap(arr[i], arr[size - i - 1]);

}

}

void findSecondLargestSmallest(int arr[], int size) {

sort(arr, arr + size);

int smallest = arr[0], secondSmallest = -1;

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

if (arr[i] > smallest) {

secondSmallest = arr[i];

break;

}

}

int largest = arr[size - 1], secondLargest = -1;

for (int i = size - 2; i >= 0; i--) {

if (arr[i] < largest) {

secondLargest = arr[i];

break;

}

}

if (secondSmallest == -1 || secondLargest == -1) {

cout << "Not enough distinct elements to find second largest and second smallest.\n";

} else {

cout << "Second smallest element: " << secondSmallest << endl;

cout << "Second largest element: " << secondLargest << endl;

}

}

int main() {

int size;

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

cin >> size;

int arr[size];

cout << "Enter " << size << " integers: ";

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

cin >> arr[i];

}

reverseArray(arr, size);

cout << "Reversed array: ";

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

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

}

cout << endl;

findSecondLargestSmallest(arr, size);

return 0;

}

OUTPUT:

Enter the size of the array: 6

Enter 6 integers: 5 1 8 3 2 8

Reversed array: 8 2 3 8 1 5

Second smallest element: 2

Second largest element: 5

3.

#include <iostream>

#include <algorithm>

using namespace std;

char toLowerCase(char ch) {

if (ch >= 'A' && ch <= 'Z') {

return ch + ('a' - 'A'); // Convert to lowercase

}

return ch;

}

bool isVowel(char ch) {

char lowerCh = toLowerCase(ch);

return (lowerCh == 'a' || lowerCh == 'e' || lowerCh == 'i' || lowerCh == 'o' || lowerCh == 'u');

}

bool isPalindrome(const string& str) {

string cleanStr, reversedStr;

for (char ch : str) {

if (ch != ' ') {

cleanStr += toLowerCase(ch);

}

}

reversedStr = cleanStr;

reverse(reversedStr.begin(), reversedStr.end());

return cleanStr == reversedStr;

}

void countFrequency(const string& str) {

int freq[256] = {0}; //

for (char ch : str) {

if ((ch >= 'A' && ch <= 'Z') || (ch >= 'a' && ch <= 'z') || (ch >= '0' && ch <= '9')) {

freq[toLowerCase(ch)]++; // Convert to lowercase before counting

}

}

cout << "\nCharacter Frequency:\n";

for (int i = 0; i < 256; i++) {

if (freq[i] > 0) {

cout << (char)i << ": " << freq[i] << endl;

}

}

}

string replaceVowels(string str, char replacement = '\*') {

for (char& ch : str) {

if (isVowel(ch)) {

ch = replacement;

}

}

return str;

}

int main() {

string input;

cout << "Enter a string: ";

getline(cin, input);

if (isPalindrome(input)) {

cout << "\nThe string is a palindrome.\n";

} else {

cout << "\nThe string is NOT a palindrome.\n";

}

countFrequency(input);

string modifiedString = replaceVowels(input);

cout << "\nString after replacing vowels: " << modifiedString << endl;

return 0;

}

OUTPUT:

Enter a string: A Santa at NASA

The string is a palindrome.

Character Frequency:

a: 6

s: 2

n: 2

t: 2

String after replacing vowels: \* S\*nt\* t N\*S

4.

#include <iostream>

using namespace std;

void generateSpiral(int n) {

int matrix[n][n];

int value = 1;

int top = 0, bottom = n - 1, left = 0, right = n - 1;

while (value <= n \* n) {

// Fill top row

for (int i = left; i <= right; i++)

matrix[top][i] = value++;

top++;

// Fill right column

for (int i = top; i <= bottom; i++)

matrix[i][right] = value++;

right--;

// Fill bottom row

for (int i = right; i >= left; i--)

matrix[bottom][i] = value++;

bottom--;

// Fill left column

for (int i = bottom; i >= top; i--)

matrix[i][left] = value++;

left++;

}

// Print the spiral matrix

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

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

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

cout << endl;

}

}

int main() {

int n;

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

cin >> n;

generateSpiral(n);

return 0;

}

OUTPUT:

1 2 3 4

12 13 14 5

11 16 15 6

10 9 8 7

5.

#include <iostream>

using namespace std;

#define N 3 // Define the size of the matrix (can be changed)

void rotateMatrix(int matrix[N][N]) {

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

for (int j = i; j < N; j++) {

swap(matrix[i][j], matrix[j][i]);

}

}

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

for (int j = 0; j < N / 2; j++) {

swap(matrix[i][j], matrix[i][N - j - 1]);

}

}

}

void printMatrix(int matrix[N][N]) {

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

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

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

}

cout << endl;

}

}

int main() {

int matrix[N][N] = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

cout << "Original Matrix:\n";

printMatrix(matrix);

rotateMatrix(matrix);

cout << "\nRotated Matrix:\n";

printMatrix(matrix);

return 0;

}

OUTPUT:

1 2 3

4 5 6

7 8 9

7 4 1

8 5 2

9 6 3