**26. strong number**

#include <iostream>

using namespace std;

int factorial(int num) {

if (num == 0 || num == 1) {

return 1;

} else {

return num \* factorial(num - 1);

}

}

int main() {

int num, originalNum, remainder, sum = 0;

cout << "Enter a number: ";

cin >> num;

originalNum = num;

while (num != 0) {

remainder = num % 10;

sum += factorial(remainder);

num /= 10;

}

if (sum == originalNum) {

cout << originalNum << " is a strong number." << endl;

} else {

cout << originalNum << " is not a strong number." << endl;

}

return 0;

}

**OUTPUT:**

Enter a number: 34

34 is not a strong number.

**27. buzz number**

#include <iostream>

using namespace std;

int main() {

int num;

cout << "Enter a number: ";

cin >> num;

if (num % 7 == 0 || num % 10 == 7) {

cout << num << " is a Buzz number." << endl;

} else {

cout << num << " is not a Buzz number." << endl;

}

return 0;

}

OUTPUT:

Enter a number: 456

456 is not a Buzz number.

**28. neon number**

#include <iostream>

#include <cmath>

using namespace std;

int main() {

int num;

int square, digitSum = 0;

cout << "Enter a positive integer: ";

cin >> num;

if (num <= 0) {

cout << "Invalid input. Please enter a positive integer." << endl;

return 0;

}

square = num \* num;

while (square != 0) {

digitSum += square % 10;

square /= 10;

}

if (digitSum == num) {

cout << num << " is a neon number." << endl;

} else {

cout << num << " is not a neon number." << endl;

}

return 0;

}

OUTPUT:

Enter a positive integer: 56

56 is not a neon number.

**29. abundant number**

#include <iostream>

using namespace std;

int main() {

int num;

int sum = 0;

cout << "Enter a positive integer: ";

cin >> num;

if (num <= 0) {

cout << "Invalid input. Please enter a positive integer." << endl;

return 0;

}

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

if (num % i == 0) {

sum += i;

}

}

if (sum > num) {

cout << num << " is an abundant number." << endl;

} else {

cout << num << " is not an abundant number." << endl;

}

return 0;

}

OUTPUT:

Enter a positive integer: 67

67 is not an abundant number.

**30. narcissistic number**

#include <iostream>

#include <cmath>

using namespace std;

int countDigits(int num) {

int count = 0;

while (num != 0) {

num /= 10;

++count;

}

return count;

}

bool isNarcissistic(int num) {

int originalNum = num;

int sum = 0;

int power = countDigits(num);

while (num != 0) {

int digit = num % 10;

sum += pow(digit, power);

num /= 10;

}

return sum == originalNum;

}

int main() {

int num;

cout << "Enter a positive integer: ";

cin >> num;

if (num <= 0) {

cout << "Invalid input. Please enter a positive integer." << endl;

return 0;

}

if (isNarcissistic(num)) {

cout << num << " is a narcissistic number." << endl;

} else {

cout << num << " is not a narcissistic number." << endl;

}

return 0;

}

OUTPUT:

Enter a positive integer: 35

35 is not a narcissistic number.

**31. print the pattern 1 22 333 4444 55555**

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter the number of rows for the pattern: ";

cin >> n;

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

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

cout << i;

}

cout << endl;

}

return 0;

}

OUTPUT:

Enter the number of rows for the pattern: 6

1

22

333

4444

55555

666666

----------

**32. print the pattern \* \* \*\* \*\* \*\*\***

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter the number of rows for the pattern: ";

cin >> n;

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

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

cout << "\*";

}

cout << endl;

}

return 0;

}

OUTPUT:

Enter the number of rows for the pattern: 7

\*

\*\*

\*\*\*

\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*

\*\*\*\*\*\*\*

**33. Print pascal triangle pattern nested for loop**

#include <iostream>

using namespace std;

int main() {

int rows;

cout << "Enter the number of rows for Pascal's triangle: ";

cin >> rows;

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

int number = 1;

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

cout << " ";

}

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

cout << number << " ";

number = number \* (i - j) / (j + 1);

}

cout << endl;

}

return 0;

}

OUTPUT:

Enter the number of rows for Pascal's triangle: 5

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

**34. Print diamond pattern with \* using nested for loop**

#include <iostream>

using namespace std;

int main() {

int n;

cout << "Enter the number of rows for the diamond pattern: ";

cin >> n;

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

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

cout << " ";

}

for (int k = 1; k <= 2 \* i - 1; ++k) {

cout << "\*";

}

cout << endl;

}

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

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

cout << " ";

}

for (int k = 1; k <= 2 \* i - 1; ++k) {

cout << "\*";

}

cout << endl;

}

return 0;

}

OUTPUT:

Enter the number of rows for the diamond pattern: 5

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

**35. Program to reverse the elements in an array**

#include <iostream>

using namespace std;

int main() {

const int MAX\_SIZE = 100;

int arr[MAX\_SIZE];

int size;

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

cin >> size;

cout << "Enter " << size << " elements into the array:" << endl;

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

cout << "Element " << i + 1 << ": ";

cin >> arr[i];

}

int start = 0;

int end = size - 1;

while (start < end) {

int temp = arr[start];

arr[start] = arr[end];

arr[end] = temp;

start++;

end--;

}

cout << "Reversed array:" << endl;

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

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

}

cout << endl;

return 0;

}

OUTPUT:

Enter the size of the array: 5

Enter 5 elements into the array:

Element 1: 2

Element 2: 3

Element 3: 5

Element 4: 7

Element 5: 8

Reversed array:

8 7 5 3 2

**36. Program to insert an element in an array at a specific position**

#include <iostream>

using namespace std;

int main() {

const int MAX\_SIZE = 100;

int arr[MAX\_SIZE];

int size, position, newValue;

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

cin >> size;

cout << "Enter " << size << " elements into the array:" << endl;

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

cout << "Element " << i + 1 << ": ";

cin >> arr[i];

}

cout << "Enter the position to insert (1 to " << size + 1 << "): ";

cin >> position;

cout << "Enter the new value to insert: ";

cin >> newValue;

if (position < 1 || position > size + 1) {

cout << "Invalid position. Please enter a position between 1 and " << size + 1 << "." << endl;

return 0;

}

for (int i = size; i >= position; --i) {

arr[i] = arr[i - 1];

}

arr[position - 1] = newValue;

size++;

cout << "Array after insertion:" << endl;

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

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

}

cout << endl;

return 0;

}

OUTPUT:

Enter the size of the array: 4

Enter 4 elements into the array:

Element 1: 2

Element 2: 3

Element 3: 7

Element 4: 8

Enter the position to insert (1 to 5): 3

Enter the new value to insert: 10

Array after insertion:

2 3 10 7 8

**37. Program to Delete an element in an array at a specific position**

#include <iostream>

using namespace std;

int main() {

const int MAX\_SIZE = 100;

int arr[MAX\_SIZE];

int size, pos;

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

cin >> size;

cout << "Enter " << size << " elements into the array:" << endl;

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

cout << "Element " << i + 1 << ": ";

cin >> arr[i];

}

cout << "Enter the position of the element to delete (1 to " << size << "): ";

cin >> pos;

if (pos < 1 || pos > size) {

cout << "Invalid position! Please enter a position between 1 and " << size << "." << endl;

return 0;

}

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

arr[i] = arr[i + 1];

}

size--;

cout << "Array after deletion:" << endl;

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

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

}

cout << endl;

return 0;

}

OUTPUT:

Enter the size of the array: 5

Enter 5 elements into the array:

Element 1: 23

Element 2: 45

Element 3: 9

Element 4: 67

Element 5: 56

Enter the position of the element to delete (1 to 5): 3

Array after deletion:

23 45 67 56

**38. Find the sum of all elements in an array**

#include <iostream>

using namespace std;

int main() {

const int MAX\_SIZE = 100;

int arr[MAX\_SIZE];

int size, sum = 0;

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

cin >> size;

cout << "Enter " << size << " elements into the array:" << endl;

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

cout << "Element " << i + 1 << ": ";

cin >> arr[i];

}

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

sum += arr[i];

}

cout << "Sum of all elements in the array: " << sum << endl;

return 0;

}

OUTPUT:

Enter the size of the array: 5

Enter 5 elements into the array:

Element 1: 45

Element 2: 89

Element 3: 67

Element 4: 23

Element 5: 50

Sum of all elements in the array: 274

**39. Find the average of all elements in an array**

#include <iostream>

using namespace std;

int main() {

const int MAX\_SIZE = 100;

int arr[MAX\_SIZE];

int size;

double sum = 0, avg;

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

cin >> size;

cout << "Enter " << size << " elements into the array:" << endl;

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

cout << "Element " << i + 1 << ": ";

cin >> arr[i];

sum += arr[i];

}

avg = sum / size;

cout << "Average of all elements in the array: " << avg << endl;

return 0;

}

OUTPUT:

Enter the size of the array: 5

Enter 5 elements into the array:

Element 1: 45

Element 2: 89

Element 3: 67

Element 4: 23

Element 5: 50

Average of all elements in the array: 54.8

**40. Find the second largest element in an array**

#include <iostream>

#include <climits>

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

int firstLargest = INT\_MIN;

int secondLargest = INT\_MIN;

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

if (arr[i] > firstLargest) {

secondLargest = firstLargest;

firstLargest = arr[i];

} else if (arr[i] > secondLargest && arr[i] != firstLargest) {

secondLargest = arr[i];

}

}

return secondLargest;

}

int main() {

int arr[] = { 3, 8, 1, 6, 2, 5 };

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

int secondLargest = findSecondLargest(arr, size);

if (secondLargest == INT\_MIN) {

std::cout << "There is no second largest element in the array." << std::endl;

} else {

std::cout << "The second largest element in the array is: " << secondLargest << std::endl;

}

return 0;

}

OUTPUT:

The second largest element in the array is: 6

**41. Find the number of occurrences of a value in an array**

#include <iostream>

int countOccurrences(int arr[], int size, int value) {

int count = 0;

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

if (arr[i] == value) {

count++;

}

}

return count;

}

int main() {

int arr[] = { 3, 7, 2, 7, 8, 7, 1, 5 };

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

int valueToFind = 7;

int occurrences = countOccurrences(arr, size, valueToFind);

std::cout << "The value " << valueToFind << " occurs " << occurrences << " times in the array." << std::endl;

return 0;

}

OUTPUT:

The value 7 occurs 3 times in the array.

**42. Merge two array**

#include <iostream>

void mergeArrays(int arr1[], int size1, int arr2[], int size2, int mergedArr[]) {

int i = 0, j = 0, k = 0;

while (i < size1 && j < size2) {

if (arr1[i] <= arr2[j]) {

mergedArr[k++] = arr1[i++];

} else {

mergedArr[k++] = arr2[j++];

}

}

while (i < size1) {

mergedArr[k++] = arr1[i++];

}

while (j < size2) {

mergedArr[k++] = arr2[j++];

}

}

int main() {

int arr1[] = { 1, 3, 5, 7 };

int size1 = sizeof(arr1) / sizeof(arr1[0]);

int arr2[] = { 2, 4, 6 };

int size2 = sizeof(arr2) / sizeof(arr2[0]);

int mergedSize = size1 + size2;

int mergedArr[mergedSize];

mergeArrays(arr1, size1, arr2, size2, mergedArr);

std::cout << "Merged array: ";

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

std::cout << mergedArr[i] << " ";

}

std::cout << std::endl;

return 0;

}

OUTPUT:

Merged array: 1 2 3 4 5 6 7

**43. Create a dynamic array using pointers and display the values**

#include <iostream>

int main() {

int size;

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

std::cin >> size;

int\* dynamicArray = new int[size];

std::cout << "Enter " << size << " integer values:" << std::endl;

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

std::cin >> dynamicArray[i];

}

std::cout << "Values in the array:" << std::endl;

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

std::cout << dynamicArray[i] << " ";

}

std::cout << std::endl;

delete[] dynamicArray;

return 0;

}

OUTPUT:

Enter the size of the array: 4

Enter 4 integer values:

5

8

9

6

Values in the array:

5 8 9 6

**44. Create a dynamic 2D (Two dimensional) array using pointers and display the values**

#include <iostream>

int main() {

int rows, cols;

std::cout << "Enter the number of rows: ";

std::cin >> rows;

std::cout << "Enter the number of columns: ";

std::cin >> cols;

int\*\* dynamic2DArray = new int\*[rows];

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

dynamic2DArray[i] = new int[cols];

}

std::cout << "Enter " << rows \* cols << " integer values for the 2D array:" << std::endl;

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

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

std::cin >> dynamic2DArray[i][j];

}

}

std::cout << "Values in the 2D array:" << std::endl;

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

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

std::cout << dynamic2DArray[i][j] << " ";

}

std::cout << std::endl;

}

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

delete[] dynamic2DArray[i];

}

delete[] dynamic2DArray;

return 0;

}

OUTPUT:

Enter the number of rows: 2

Enter the number of columns: 2

Enter 4 integer values for the 2D array:

1 2 3 4

Values in the 2D array:

1 2

3 4

**45. Add 2 matrices**

#include <bits/stdc++.h>

using namespace std;

#define N 2

void add(int A[][N], int B[][N], int C[][N])

{

int i, j;

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

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

C[i][j] = A[i][j] + B[i][j];

}

// Driver code

int main()

{

int A[N][N] = { { 4, 5},

{ 2, 3}, };

int B[N][N] = { { 8, 1},

{ 3, 6}, };

// To store the result

int C[N][N];

int i, j;

add(A, B, C);

cout << "Result matrix is " << endl;

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

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

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

cout << endl;

}

return 0;

}

OUTPUT:

Result matrix is

12 6

5 9

**46. Multiply 2 matrices**

#include <iostream>

using namespace std;

int main()

{

int a[10][10],b[10][10],mul[10][10],r,c,i,j,k;

cout<<"enter the number of row=";

cin>>r;

cout<<"enter the number of column=";

cin>>c;

cout<<"enter the first matrix element=\n";

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

{

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

{

cin>>a[i][j];

}

}

cout<<"enter the second matrix element=\n";

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

{

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

{

cin>>b[i][j];

}

}

cout<<"multiply of the matrix=\n";

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

{

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

{

mul[i][j]=0;

for(k=0;k<c;k++)

{

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

}

}

}

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

{

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

{

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

}

cout<<"\n";

}

return 0;

}

OUTPUT:

enter the number of row=2

enter the number of column=2

enter the first matrix element=

2

4

5

6

enter the second matrix element=

4

9

7

6

multiply of the matrix=

36 42

62 81

**47.Find the sum of diagonals of a matrix**

#include <iostream>

#include <cmath>

#define N 4

using namespace std;

float solve( int M[ N ][ N ] ){

int sum\_major = 0;

int sum\_minor = 0;

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

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

if( i == j ) {

sum\_major = sum\_major + M[ i ][ j ];

}

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

sum\_minor = sum\_minor + M[ i ][ j ];

}

}

}

cout << "The sum of major diagonal: " << sum\_major << endl;

cout << "The sum of minor diagonal: " << sum\_minor << endl;

}

int main(){

int mat1[ N ][ N ] = {

{5, 8, 74, 21},

{48, 2, 98, 6},

{85, 12, 10, 6},

{6, 12, 18, 32},

};

cout << "For the first matrix: " << endl;

solve( mat1 );

}

OUTPUT:

For the first matrix:

The sum of major diagonal: 49

The sum of minor diagonal: 137

**48.Find factorial using function**

#include<iostream>

using namespace std;

int factorial(int);

int main()

{

int n,result;

cout<<"enter the non-negative number:";

cin>>n;

result=factorial(n);

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

}

int factorial(int n)

{

if(n>1)

{

return n\*factorial(n-1);

}

else

{

return 1;

}

}

OUTPUT:

enter the number:5

factorial of 5:120

**49 Find prime number using function**

#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 main() {

int num;

cout << "Enter a positive integer: ";

cin >> num;

if (isPrime(num))

cout << num << " is a prime number." << endl;

else

cout << num << " is not a prime number." << endl;

return 0;

}

OUTPUT:

Enter a positive integer: 12

12 is not a prime number

**50. Find the reverse of a string using function**

#include <iostream>

#include <string>

using namespace std;

string reverseString(const string& str) {

string reversedStr;

for (int i = str.length() - 1; i >= 0; --i) {

reversedStr += str[i];

}

return reversedStr;

}

int main() {

string inputStr;

cout << "Enter a string: ";

getline(cin, inputStr);

string reversed = reverseString(inputStr);

cout << "Reversed string: " << reversed << endl;

return 0;

}

OUTPUT:

Enter a string: saveetha

Reversed string: ahteevas

**51. Find minimum and maximum element in an array using function**

#include <iostream>

using namespace std;

int findMinimum(const int arr[], int size) {

int minElement = arr[0];

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

if (arr[i] < minElement) {

minElement = arr[i];

}

}

return minElement;

}

int findMaximum(const int arr[], int size) {

int maxElement = arr[0];

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

if (arr[i] > maxElement) {

maxElement = arr[i];

}

}

return maxElement;

}

int main() {

int size;

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

cin >> size;

int arr[size];

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

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

cin >> arr[i];

}

int minElement = findMinimum(arr, size);

int maxElement = findMaximum(arr, size);

cout << "Minimum element: " << minElement << endl;

cout << "Maximum element: " << maxElement << endl;

return 0;

}

OUTPUT:

Enter the size of the array: 5

Enter elements of the array: 34

45

63

12

56

Minimum element: 12

Maximum element: 63

**52. Find GCD of two number using function**

#include <iostream>

using namespace std;

int main() {

int n1, n2,GCD;

cout << "Enter two numbers: ";

cin >> n1 >> n2;

if ( n2 > n1) {

int temp = n2;

n2 = n1;

n1 = temp;

}

for (int i = 1; i <= n2; ++i) {

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

hcf = i;

}

}

cout << "GCD = " << GCD;

return 0;

}

OUTPUT:

Enter two numbers: 12

45

GCD = 3

**53. Function to count the no of elements in a string**

#include <iostream>

#include <string>

int countElements(const std::string& str) {

return str.length();

}

int main() {

std::string inputString;

std::cout << "Enter a string: ";

std::getline(std::cin, inputString);

int count = countElements(inputString);

std::cout << "Number of elements in the string: " << count << std::endl;

return 0;

}

OUTPUT:

Enter a string: object oriented programming

Number of elements in the string: 27

**54. Convert Celsius and Fahrenheit using function**

#include <iostream>

double celsiusToFahrenheit(double celsius) {

return (celsius \* 9 / 5) + 32;

}

double fahrenheitToCelsius(double fahrenheit) {

return (fahrenheit - 32) \* 5 / 9;

}

int main() {

double temperature;

char choice;

std::cout << "Enter temperature: ";

std::cin >> temperature;

std::cout << "Convert to (C)elsius or (F)ahrenheit? ";

std::cin >> choice;

switch (choice) {

case 'C':

case 'c':

std::cout << temperature << " Fahrenheit is " << fahrenheitToCelsius(temperature) << " Celsius." << std::endl;

break;

case 'F':

case 'f':

std::cout << temperature << " Celsius is " << celsiusToFahrenheit(temperature) << " Fahrenheit." << std::endl;

break;

default:

std::cout << "Invalid choice." << std::endl;

}

return 0;

}

OUTPUT:

Enter temperature: 45

Convert to (C)elsius or (F)ahrenheit? f

45 Celsius is 113 Fahrenheit.

**55. Find the area of a circle using function**

#include <iostream>

#include <cmath>

double calculateArea(double radius) {

const double pi = 3.14159;

return pi \* radius \* radius;

}

int main() {

double radius;

std::cout << "Enter the radius of the circle: ";

std::cin >> radius;

if (radius < 0) {

std::cout << "Radius cannot be negative." << std::endl;

return 1;

}

double area = calculateArea(radius);

std::cout << "Area of the circle with radius " << radius << " is: " << area << std::endl;

return 0;

}

OUTPUT:

Enter the radius of the circle: 5

Area of the circle with radius 5 is: 78.5397

**56. Check whether the string is palindrome or not**

#include<iostream>

#include<string>

using namespace std;

int main() {

string str = "welcome";

string rev = "";

char ch;

int n = str.length();

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

ch = str[i];

rev = rev + ch;

}

if(rev == str) {

cout << str << " is a palindrome" << endl;

} else {

cout << str << " is not a palindrome" << endl;

}

return 0;

}

OUTPUT:

welcome is not a palindrome

**57. Write a c++ program to create a class for a bank account with a constructor and a destructor**

#include <iostream>

#include <string>

class BankAccount {

private:

std::string accountNumber;

double balance;

public:

BankAccount(const std::string& accNumber, double initialBalance) : accountNumber(accNumber), balance(initialBalance) {

std::cout << "Bank account created with account number: " << accountNumber << std::endl;

}

~BankAccount() {

std::cout << "Bank account with account number " << accountNumber << " is being destroyed." << std::endl;

}

void displayBalance() {

std::cout << "Balance in account " << accountNumber << " is: $" << balance << std::endl;

}

};

int main() {

BankAccount account1("123456", 1000.0);

BankAccount account2("654321", 2000.0);

account1.displayBalance();

account2.displayBalance();

return 0;

}

OUTPUT:

Bank account created with account number: 123456

Bank account created with account number: 654321

Balance in account 123456 is: $1000

Balance in account 654321 is: $2000

Bank account with account number 654321 is being destroyed.

Bank account with account number 123456 is being destroyed

**58. Write a c++ program to create a class for a car with a constructor and a destructor**

#include <iostream>

#include <string>

class Car {

private:

std::string brand;

std::string model;

int year;

public:

Car(const std::string& carBrand, const std::string& carModel, int carYear)

: brand(carBrand), model(carModel), year(carYear) {

std::cout << "A " << year << " " << brand << " " << model << " car has been created." << std::endl;

}

~Car() {

std::cout << "The " << year << " " << brand << " " << model << " car is being destroyed." << std::endl;

}

void drive() {

std::cout << "Driving the " << year << " " << brand << " " << model << " car." << std::endl;

}

};

int main() {

Car myCar("Toyota", "Camry", 2022);

Car anotherCar("Honda", "Civic", 2020);

myCar.drive();

anotherCar.drive();

return 0;

}

OUTPUT:

A 2022 Toyota Camry car has been created.

A 2020 Honda Civic car has been created.

Driving the 2022 Toyota Camry car.

Driving the 2020 Honda Civic car.

The 2020 Honda Civic car is being destroyed.

The 2022 Toyota Camry car is being destroyed.

**59. Write a c++ program to create a class for a rectangle with a constructor and a destructor**

#include <iostream>

class Rectangle {

private:

double length;

double width;

public:

Rectangle(double len, double wid) : length(len), width(wid) {

std::cout << "Rectangle object created." << std::endl;

}

~Rectangle() {

std::cout << "Rectangle object destroyed." << std::endl;

}

double area() {

return length \* width;

}

double perimeter() {

return 2 \* (length + width);

}

};

int main() {

Rectangle rectangle1(5.0, 3.0);

Rectangle rectangle2(7.0, 4.0);

std::cout << "Area of rectangle1: " << rectangle1.area() << std::endl;

std::cout << "Perimeter of rectangle1: " << rectangle1.perimeter() << std::endl;

std::cout << "Area of rectangle2: " << rectangle2.area() << std::endl;

std::cout << "Perimeter of rectangle2: " << rectangle2.perimeter() << std::endl;

return 0;

}

OUTPUT:

Rectangle object created.

Rectangle object created.

Area of rectangle1: 15

Perimeter of rectangle1: 16

Area of rectangle2: 28

Perimeter of rectangle2: 22

Rectangle object destroyed.

Rectangle object destroyed.

**60.Write a c++ program to create a class for a book with a constructor and a destructor**

#include <iostream>

#include <string>

class Book {

private:

std::string title;

std::string author;

int pageCount;

public:

Book(const std::string& t, const std::string& a, int pages) : title(t), author(a), pageCount(pages) {

std::cout << "Creating book: " << title << " by " << author << ", " << pageCount << " pages." << std::endl;

}

~Book() {

std::cout << "Destroying book: " << title << std::endl;

}

};

int main() {

Book myBook("The Great Gatsby", "F. Scott Fitzgerald", 218);

return 0;

}

OUTPUT:

Creating book: The Great Gatsby by F. Scott Fitzgerald, 218 pages.

Destroying book: The Great Gatsby

**61.Write a c++ program to create a class for student with a constructor and a destructor**

#include <iostream>

#include <string>

using namespace std;

class Student {

private:

string name;

int age;

public:

Student(const string& n, int a) : name(n), age(a) {

cout << "Constructor called for " << name << endl;

}

~Student() {

cout << "Destructor called for " << name << endl;

}

void displayInfo() const {

cout << "Name: " << name << endl;

cout << "Age: " << age << endl;

}

};

int main() {

Student student1("John Doe", 20);

Student student2("Jane Smith", 22);

student1.displayInfo();

student2.displayInfo();

return 0;

}

OUTPUT:

Age: 20

Name: Jane Smith

Age: 22

Destructor called for Jane Smith

Destructor called for John Doe