Fibonacci Series in C++

**Fibonacci Series in C++:** In case of fibonacci series, next number is the sum of previous two numbers for example 0, 1, 1, 2, 3, 5, 8, 13, 21 etc. The first two numbers of fibonacci series are 0 and 1.

There are two ways to write the fibonacci series program:

* Fibonacci Series without recursion
* Fibonacci Series using recursion

Fibonaccci Series in C++ without Recursion

Let's see the fibonacci series program in C++ without recursion.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main() {
4. **int** n1=0,n2=1,n3,i,number;
5. cout<<"Enter the number of elements: ";
6. cin>>number;
7. cout<<n1<<" "<<n2<<" "; //printing 0 and 1
8. **for**(i=2;i<number;++i) //loop starts from 2 because 0 and 1 are already printed
9. {
10. n3=n1+n2;
11. cout<<n3<<" ";
12. n1=n2;
13. n2=n3;
14. }
15. **return** 0;
16. }

Output:

C++ vs Java

Enter the number of elements: 10

0 1 1 2 3 5 8 13 21 34

Fibonnaci series using recursion in C++

Let's see the fibonacci series program in C++ using recursion.

1. #include<iostream>
2. **using** **namespace** std;
3. **void** printFibonacci(**int** n){
4. **static** **int** n1=0, n2=1, n3;
5. **if**(n>0){
6. n3 = n1 + n2;
7. n1 = n2;
8. n2 = n3;
9. cout<<n3<<" ";
10. printFibonacci(n-1);
11. }
12. }
13. **int** main(){
14. **int** n;
15. cout<<"Enter the number of elements: ";
16. cin>>n;
17. cout<<"Fibonacci Series: ";
18. cout<<"0 "<<"1 ";
19. printFibonacci(n-2);  //n-2 because 2 numbers are already printed
20. **return** 0;
21. }

Output:

Enter the number of elements: 15

Fibonacci Series: 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377

Prime Number Program in C++

Prime number is a number that is greater than 1 and divided by 1 or itself. In other words, prime numbers can't be divided by other numbers than itself or 1. For example 2, 3, 5, 7, 11, 13, 17, 19, 23.... are the prime numbers.

Let's see the prime number program in C++. In this C++ program, we will take an input from the user and check whether the number is prime or not.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** n, i, m=0, flag=0;
6. cout << "Enter the Number to check Prime: ";
7. cin >> n;
8. m=n/2;
9. **for**(i = 2; i <= m; i++)
10. {
11. **if**(n % i == 0)
12. {
13. cout<<"Number is not Prime."<<endl;
14. flag=1;
15. **break**;
16. }
17. }
18. **if** (flag==0)
19. cout << "Number is Prime."<<endl;
20. **return** 0;
21. }

Output:

Enter the Number to check Prime: 17

Number is Prime.

Enter the Number to check Prime: 57

Number is not Prime.

Palindrome program in C++

A **palindrome number** is a number that is same after reverse. For example 121, 34543, 343, 131, 48984 are the palindrome numbers.

Palindrome number algorithm

* Get the number from user
* Hold the number in temporary variable
* Reverse the number
* Compare the temporary number with reversed number
* If both numbers are same, print palindrome number
* Else print not palindrome number

Let's see the palindrome program in C++. In this program, we will get an input from the user and check whether number is palindrome or not.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** n,r,sum=0,temp;
6. cout<<"Enter the Number=";
7. cin>>n;
8. temp=n;
9. **while**(n>0)
10. {
11. r=n%10;
12. sum=(sum\*10)+r;
13. n=n/10;
14. }
15. **if**(temp==sum)
16. cout<<"Number is Palindrome.";
17. **else**
18. cout<<"Number is not Palindrome.";
19. **return** 0;
20. }

Output:

Enter the Number=121

Number is Palindrome.

Enter the number=113

Number is not Palindrome.

Factorial program in C++

**Factorial Program in C++:** Factorial of n is the product of all positive descending integers. Factorial of n is denoted by n!. For example:

1. 4! = 4\*3\*2\*1 = 24
2. 6! = 6\*5\*4\*3\*2\*1 = 720

Here, 4! is pronounced as "4 factorial", it is also called "4 bang" or "4 shriek".

The factorial is normally used in Combinations and Permutations (mathematics).

There are many ways to write the factorial program in C++ language. Let's see the 2 ways to write the factorial program.

History of Java

* Factorial Program using loop
* Factorial Program using recursion

Factorial Program using Loop

Let's see the factorial Program in C++ using loop.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** i,fact=1,number;
6. cout<<"Enter any Number: ";
7. cin>>number;
8. **for**(i=1;i<=number;i++){
9. fact=fact\*i;
10. }
11. cout<<"Factorial of " <<number<<" is: "<<fact<<endl;
12. **return** 0;
13. }

Output:

Enter any Number: 5

Factorial of 5 is: 120

Factorial Program using Recursion

Let's see the factorial program in C++ using recursion.

1. #include<iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** factorial(**int**);
6. **int** fact,value;
7. cout<<"Enter any number: ";
8. cin>>value;
9. fact=factorial(value);
10. cout<<"Factorial of a number is: "<<fact<<endl;
11. **return** 0;
12. }
13. **int** factorial(**int** n)
14. {
15. **if**(n<0)
16. **return**(-1); /\*Wrong value\*/
17. **if**(n==0)
18. **return**(1);  /\*Terminating condition\*/
19. **else**
20. {
21. **return**(n\*factorial(n-1));
22. }
23. }

Output:

Enter any number: 6

Factorial of a number is: 720

Armstrong Number in C++

Before going to write the C++ program to check whether the number is Armstrong or not, let's understand what is Armstrong number.

**Armstrong number** is a number that is equal to the sum of cubes of its digits. For example 0, 1, 153, 370, 371 and 407 are the Armstrong numbers.

Let's try to understand why **371** is an Armstrong number.

1. 371 = (3\*3\*3)+(7\*7\*7)+(1\*1\*1)
2. where:
3. (3\*3\*3)=27
4. (7\*7\*7)=343
5. (1\*1\*1)=1
6. So:
7. 27+343+1=371

Let's see the C++ program to check Armstrong Number.

Features of Java - Javatpoint

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** n,r,sum=0,temp;
6. cout<<"Enter the Number=  ";
7. cin>>n;
8. temp=n;
9. **while**(n>0)
10. {
11. r=n%10;
12. sum=sum+(r\*r\*r);
13. n=n/10;
14. }
15. **if**(temp==sum)
16. cout<<"Armstrong Number."<<endl;
17. **else**
18. cout<<"Not Armstrong Number."<<endl;
19. **return** 0;
20. }

Output:

Enter the Number= 371

Armstrong Number.

Enter the Number= 342

Not Armstrong Number.

Sum of digits program in C++

We can write the sum of digits program in C++ language by the help of loop and mathematical operation only.

Sum of digits algorithm

To get sum of each digit by C++ program, use the following algorithm:

* **Step 1:** Get number by user
* **Step 2:** Get the modulus/remainder of the number
* **Step 3:** sum the remainder of the number
* **Step 4:** Divide the number by 10
* **Step 5:** Repeat the step 2 while number is greater than 0.

Let's see the sum of digits program in C++.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** n,sum=0,m;
6. cout<<"Enter a number: ";
7. cin>>n;
8. **while**(n>0)
9. {
10. m=n%10;
11. sum=sum+m;
12. n=n/10;
13. }
14. cout<<"Sum is= "<<sum<<endl;
15. **return** 0;
16. }

Output:

OOPs Concepts in Java

Enter a number: 23

Sum is= 5

Enter a number: 624

Sum is= 12

C++ Program to reverse number

We can reverse a number in C++ using loop and arithmetic operators. In this program, we are getting number as input from the user and reversing that number.

Let's see a simple C++ example to reverse a given number.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** n, reverse=0, rem;
6. cout<<"Enter a number: ";
7. cin>>n;
8. **while**(n!=0)
9. {
10. rem=n%10;
11. reverse=reverse\*10+rem;
12. n/=10;
13. }
14. cout<<"Reversed Number: "<<reverse<<endl;
15. **return** 0;
16. }

Output:

Enter a number: 234

Reversed Number: 432

C++ Program to swap two numbers without third variable

We can swap two numbers without using third variable. There are two common ways to swap two numbers without using third variable:

1. By + and -
2. By \* and /

Using \* and /

Let's see a simple C++ example to swap two numbers without using third variable.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** a=5, b=10;
6. cout<<"Before swap a= "<<a<<" b= "<<b<<endl;
7. a=a\*b; //a=50 (5\*10)
8. b=a/b; //b=5 (50/10)
9. a=a/b; //a=10 (50/5)
10. cout<<"After swap a= "<<a<<" b= "<<b<<endl;
11. **return** 0;
12. }

Output:

Before swap a= 5 b= 10

After swap a= 10 b= 5

Using + and -

Let's see another example to swap two numbers using + and -.

HTML Tutorial

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** a=5, b=10;
6. cout<<"Before swap a= "<<a<<" b= "<<b<<endl;
7. a=a+b; //a=15 (5+10)
8. b=a-b; //b=5 (15-10)
9. a=a-b; //a=10 (15-5)
10. cout<<"After swap a= "<<a<<" b= "<<b<<endl;
11. **return** 0;
12. }

Output:

Before swap a= 5 b= 10

After swap a= 10 b= 5

Matrix multiplication in C++

We can add, subtract, multiply and divide 2 matrices. To do so, we are taking input from the user for row number, column number, first matrix elements and second matrix elements. Then we are performing multiplication on the matrices entered by the user.

In matrix multiplication first matrix one row element is multiplied by second matrix all column elements.

Let's try to understand the matrix multiplication of **3\*3 and 3\*3** matrices by the figure given below:



Let's see the program of matrix multiplication in C++.

HTML Tutorial

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** a[10][10],b[10][10],mul[10][10],r,c,i,j,k;
6. cout<<"enter the number of row=";
7. cin>>r;
8. cout<<"enter the number of column=";
9. cin>>c;
10. cout<<"enter the first matrix element=\n";
11. **for**(i=0;i<r;i++)
12. {
13. **for**(j=0;j<c;j++)
14. {
15. cin>>a[i][j];
16. }
17. }
18. cout<<"enter the second matrix element=\n";
19. **for**(i=0;i<r;i++)
20. {
21. **for**(j=0;j<c;j++)
22. {
23. cin>>b[i][j];
24. }
25. }
26. cout<<"multiply of the matrix=\n";
27. **for**(i=0;i<r;i++)
28. {
29. **for**(j=0;j<c;j++)
30. {
31. mul[i][j]=0;
32. **for**(k=0;k<c;k++)
33. {
34. mul[i][j]+=a[i][k]\*b[k][j];
35. }
36. }
37. }
38. //for printing result
39. **for**(i=0;i<r;i++)
40. {
41. **for**(j=0;j<c;j++)
42. {
43. cout<<mul[i][j]<<" ";
44. }
45. cout<<"\n";
46. }
47. **return** 0;
48. }

Output:

enter the number of row=3

enter the number of column=3

enter the first matrix element=

1 2 3

1 2 3

1 2 3

enter the second matrix element=

1 1 1

2 1 2

3 2 1

multiply of the matrix=

14 9 8

14 9 8

14 9 8

C++ Program to convert Decimal to Binary

We can convert any decimal number (base-10 (0 to 9)) into binary number (base-2 (0 or 1)) by C++ program.

Decimal Number

Decimal number is a base 10 number because it ranges from 0 to 9, there are total 10 digits between 0 to 9. Any combination of digits is decimal number such as 223, 585, 192, 0, 7 etc.

Binary Number

Binary number is a base 2 number because it is either 0 or 1. Any combination of 0 and 1 is binary number such as 1001, 101, 11111, 101010 etc.

Let's see the some binary numbers for the decimal number.

Features of Java - Javatpoint

|  |  |
| --- | --- |
| **Decimal** | **Binary** |
| 1 | 0 |
| 2 | 10 |
| 3 | 11 |
| 4 | 100 |
| 5 | 101 |
| 6 | 110 |
| 7 | 111 |
| 8 | 1000 |
| 9 | 1001 |
| 10 | 1010 |

Decimal to Binary Conversion Algorithm

**Step 1:** Divide the number by 2 through % (modulus operator) and store the remainder in array

**Step 2:** Divide the number by 2 through / (division operator)

**Step 3:** Repeat the step 2 until the number is greater than zero

Let's see the C++ example to convert decimal to binary.

1. #include <iostream>
2. **using** **namespace** std;
3. **int** main()
4. {
5. **int** a[10], n, i;
6. cout<<"Enter the number to convert: ";
7. cin>>n;
8. **for**(i=0; n>0; i++)
9. {
10. a[i]=n%2;
11. n= n/2;
12. }
13. cout<<"Binary of the given number= ";
14. **for**(i=i-1 ;i>=0 ;i--)
15. {
16. cout<<a[i];
17. }
18. }

Output:

Enter the number to convert: 9

Binary of the given number= 1001

C++ program to find greatest of four numbers

In this tutorial, we will write a C++ program to find the greatest of four numbers.

**For example**

a = 10, b = 50, c = 20, d = 25

The greatest number is b 50

Java Try Catch

a = 35, b = 50, c = 99, d = 2

The greatest number is c 99

**Approach 1**

The approach is the traditional way of searching for the greatest among four numbers. The if condition checks whether a is greater and then use if-else to check for b, another if-else to check for c, and the last else to print d as the greatest.

**Algorithm**

* START
* INPUT FOUR NUMBERS A, B, C, D
* IF A > B THEN  
  IF A > C THEN  
  IF A > D THEN  
  A IS THE GREATEST  
  ELSE  
  D IS THE GREATEST
* ELSE IF B > C THEN  
  IF B > D THEN  
  B IS THE GREATEST  
  ELSE  
  D IS THE GREATEST
* ELSE IF C > D THEN  
  C IS THE GREATEST
* ELSE  
  D IS THE GREATEST

**C++ Code**

1. #include <bits/stdc++.h>
2. **using** **namespace** std;
3. **void** find\_greatest(**int** a, **int** b, **int** c, **int** d)
4. {
5. **if** (a > b) {
6. **if** (a > c) {
7. **if** (a > d) {
8. cout << "a is greatest";
9. }
10. **else** {
11. cout << "d is greatest";
12. }
13. }
14. }
15. **else** **if** (b > c) {
16. **if** (b > d) {
17. cout << "b is greatest";
18. }
19. **else** {
20. cout << "d is greatest";
21. }
22. }
23. **else** **if** (c > d) {
24. cout << "c is greatest";
25. }
26. **else** {
27. cout << "d is greatest";
28. }
29. }
31. **int** main()
32. {
33. **int** a = 10, b = 50, c = 20, d = 25;
34. cout << "a=" << a << " b=" << b << " c=" << c << " d=" << d;
35. cout << "\n";
36. find\_greatest(a, b, c, d);
37. a = 35, b = 50, c = 99, d = 2;
38. cout << "\n";
39. cout << "a=" << a << " b=" << b << " c=" << c << " d=" << d;
40. cout << "\n";
41. find\_greatest(a, b, c, d);
43. **return** 0;
44. }

**Output**

a=10 b=50 c=20 d=25

b is greatest

a=35 b=50 c=99 d=2

c is greatest

**Approach 2**

This approach uses the inbuilt max function.

Here is the syntax of max function

**template constexpr const T& max (const T& a, const T& b);**

Here, a and b are the numbers to be compared.

**Return:** Larger of the two values.

**For example**

**std :: max(2,5)** will return 5

So, to find out the maximum of 4 numbers, we can use chaining of a max function as follows -

**int x = max(a, max(b, max(c, d)));**

**C++ code**

1. #include <bits/stdc++.h>
2. **using** **namespace** std;
3. **void** find\_greatest(**int** a, **int** b, **int** c, **int** d)
4. {
5. **int** x = max(a, max(b, max(c, d)));
6. **if** (x == a)
7. cout << "a is greatest";
8. **if** (x == b)
9. cout << "b is greatest";
10. **if** (x == c)
11. cout << "c is greatest";
12. **if** (x == d)
13. cout << "d is greatest";
14. }
16. **int** main()
17. {
18. **int** a = 10, b = 50, c = 20, d = 25;
19. cout << "a=" << a << " b=" << b << " c=" << c << " d=" << d;
20. cout << "\n";
21. find\_greatest(a, b, c, d);
22. a = 35, b = 50, c = 99, d = 2;
23. cout << "\n";
24. cout << "a=" << a << " b=" << b << " c=" << c << " d=" << d;
25. cout << "\n";
26. find\_greatest(a, b, c, d);
28. **return** 0;
29. }

**Output**

a=10 b=50 c=20 d=25

b is greatest

a=35 b=50 c=99 d=2

c is greatest