1. Write a C++ program to print the following output using for loops.

1

22

333

4444

55555

#include <iostream>

int main()

{

    int i, j;

    for (i = 1; i <= 5; i++)

    {

        for (j = 1; j <= i; j++)

        {

            printf("%d", i);

        }

        printf("\n");

    }

    return 0;

}

1. Write a program in C++ to input a single digit number and print a rectangular form of 4 columns and 6 rows

5 5 5 5

5 5

5 5

5 5

5 5

5 5 5 5

#include <iostream>

using namespace std;

int main()

{

    int x;

    cout << "\n\n Make a rectangular shape by a single digit number :\n";

    cout << "--------------------------------------------------------\n";

    cout << " Input the number : ";

    cin >> x;

    cout << " " << x << x << x << x << endl;

    cout << " " << x << " "

         << " " << x << endl;

    cout << " " << x << " "

         << " " << x << endl;

    cout << " " << x << " "

         << " " << x << endl;

    cout << " " << x << " "

         << " " << x << endl;

    cout << " " << x << x << x << x << endl;

    cout << endl;

    return 0;

}

1. Write a C++ program to display the current date and time.

#include <iostream>

#include <cmath>

#include <ctime>

using namespace std;

int main()

{

    time\_t t = time(NULL);

    tm \*tPtr = localtime(&t);

    cout << "\n\n Display the Current Date and Time :\n";

    cout << "----------------------------------------\n";

    cout << " seconds = " << (tPtr->tm\_sec) << endl;

    cout << " minutes = " << (tPtr->tm\_min) << endl;

    cout << " hours = " << (tPtr->tm\_hour) << endl;

    cout << " day of month = " << (tPtr->tm\_mday) << endl;

    cout << " month of year = " << (tPtr->tm\_mon) + 1 << endl;

    cout << " year = " << (tPtr->tm\_year) + 1900 << endl;

    cout << " weekday = " << (tPtr->tm\_wday) << endl;

    cout << " day of year = " << (tPtr->tm\_yday) << endl;

    cout << " daylight savings = " << (tPtr->tm\_isdst) << endl;

    cout << endl;

    cout << endl;

    cout << " Current Date: " << (tPtr->tm\_mday) << "/" << (tPtr->tm\_mon) + 1 << "/" << (tPtr->tm\_year) + 1900 << endl;

    cout << " Current Time: " << (tPtr->tm\_hour) << ":" << (tPtr->tm\_min) << ":" << (tPtr->tm\_sec) << endl;

    cout << endl;

    return 0;

}

1. Write a program in C++ to enter P, T, R and calculate Simple Interest.

#include <iostream>

using namespace std;

int main()

{

    int p, r, t, i;

    cout << "\n\n Calculate the Simple Interest :\n";

    cout << " -----------------------------------\n";

    cout << " Input the Principle: ";

    cin >> p;

    cout << " Input the Rate of Interest: ";

    cin >> r;

    cout << " Input the Time: ";

    cin >> t;

    i = (p \* r \* t) / 100;

    cout << " The Simple interest for the amount " << p << " for " << t << " years @ " << r << " % is: " << i;

    cout << endl;

    return 0;

}

1. Write a C++ program for unary minus (-) operator overloading
2. #include <iostream>
3. using namespace std;
4. class NUM
5. {
6. private:
7. int n;
8. public:
9. // function to get number
10. void getNum(int x)
11. {
12. n = x;
13. }
14. // function to display number
15. void dispNum(void)
16. {
17. cout << "value of n is: " << n;
18. }
19. // unary - operator overloading
20. void operator-(void)
21. {
22. n = -n;
23. }
24. };
25. int main()
26. {
27. NUM num;
28. num.getNum(10);
29. -num;
30. num.dispNum();
31. cout << endl;
32. return 0;
33. }

2. Write a C++ program for unary increment (++) and decrement (--) operator overloading.

#include <iostream>

using namespace std;

class NUM

{

private:

    int n;

public:

    // function to get number

    void getNum(int x)

    {

        n = x;

    }

    // function to display number

    void dispNum(void)

    {

        cout << "value of n is: " << n;

    }

    // unary ++ operator overloading

    void operator++(void)

    {

        n = ++n;

    }

    // unary -- operator overloading

    void operator--(void)

    {

        n = --n;

    }

};

int main()

{

    NUM num;

    num.getNum(10);

    ++num;

    cout << "After increment - ";

    num.dispNum();

    cout << endl;

    --num;

    cout << "After decrement - ";

    num.dispNum();

    cout << endl;

    return 0;

}

3. Write a C++ program for unary logical NOT (!) operator overloading

#include <iostream>

using namespace std;

class NUM

{

private:

    int n;

public:

    // function to get number

    void getNum(int x)

    {

        n = x;

    }

    // function to display number

    void dispNum(void)

    {

        cout << "value of n is: " << n;

    }

    // unary ! operator overloading

    void operator!(void)

    {

        n = !n;

    }

};

int main()

{

    NUM num;

    num.getNum(10);

    cout << "Before calling Operator Overloading:";

    num.dispNum();

    cout << endl;

    !num;

    cout << "After  calling Operator Overloading:";

    num.dispNum();

    cout << endl;

    return 0;

}