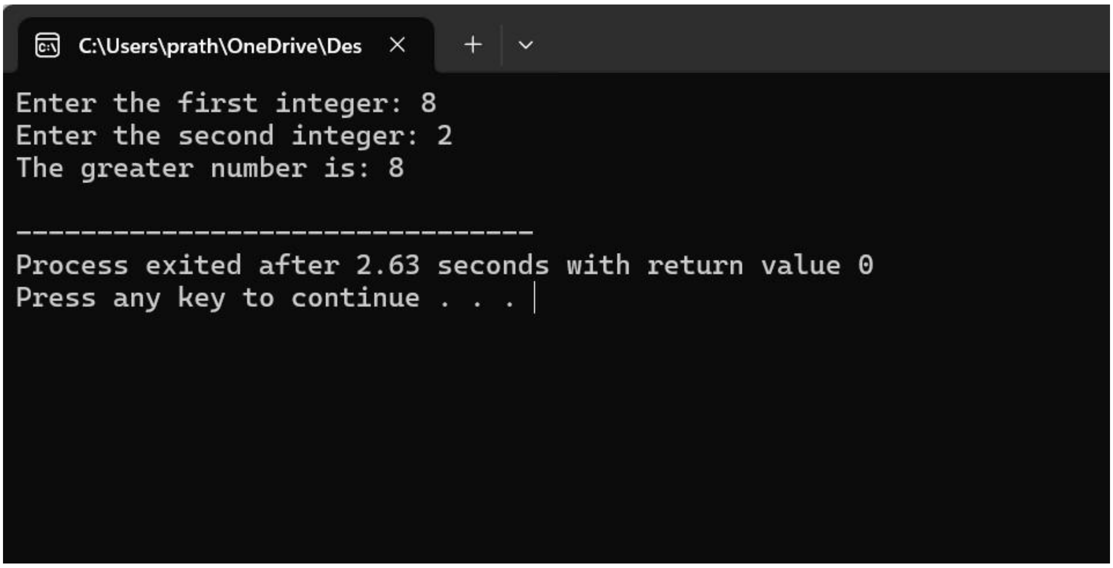
1. Write a program to read in two integers and use the conditional operator to determine which number is greater.

Program:

```
#include <iostream>
using namespace std;

int main() {
    int num1, num2;

    cout << "Enter the first integer: ";
    cin >> num1;
    cout << "Enter the second integer: ";
    cin >> num2;

    int greater = (num1 > num2) ? num1 : num2;

    cout << "The greater number is: " << greater << endl;

    return 0;
}
```

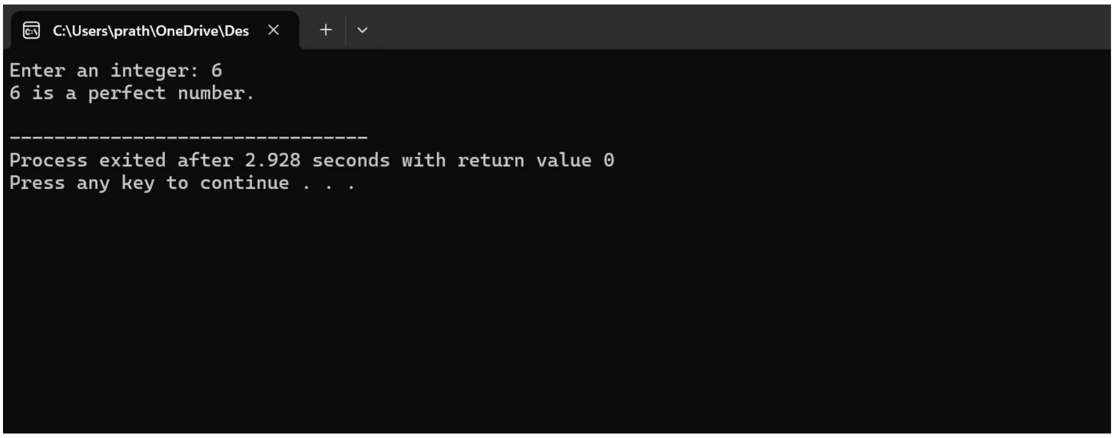


```
Enter the first integer: 8
Enter the second integer: 2
The greater number is: 8

--------------------------------
Process exited after 2.63 seconds with return value 0
Press any key to continue . . .
```

2. Write a program to read in an integer and determine if it is a perfect number or not.

Program:

```
#include <iostream>
using namespace std;

bool isPerfectNumber(int num) {
    int sum = 0;

    for (int i = 1; i <= num / 2; i++) {
        if (num % i == 0) {
            sum += i;
        }
    }

    return sum == num;
}

int main() {
    int number;

    cout << "Enter an integer: ";
    cin >> number;

    if (isPerfectNumber(number)) {
        cout << number << " is a perfect number." << endl;
    } else {
        cout << number << " is not a perfect number." << endl;
    }

    return 0;
}
```

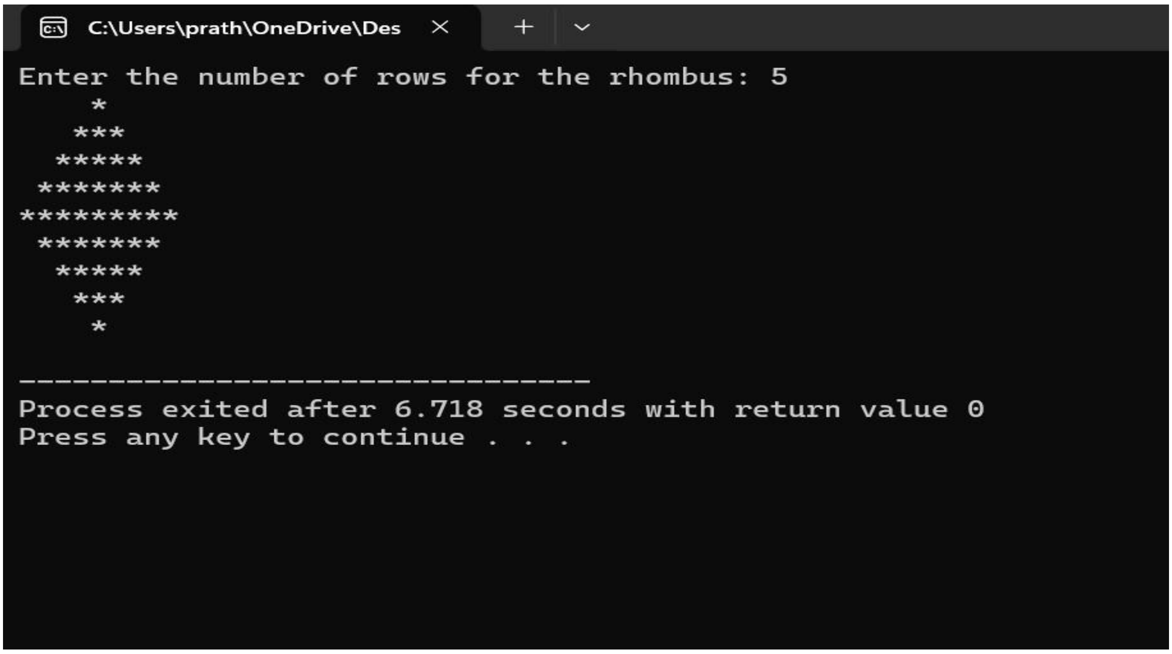


```
Enter an integer: 6
6 is a perfect number.

--------------------------------
Process exited after 2.928 seconds with return value 0
Press any key to continue . . .
```

3. Write a program to print the following pattern using nested for loops.

Program:

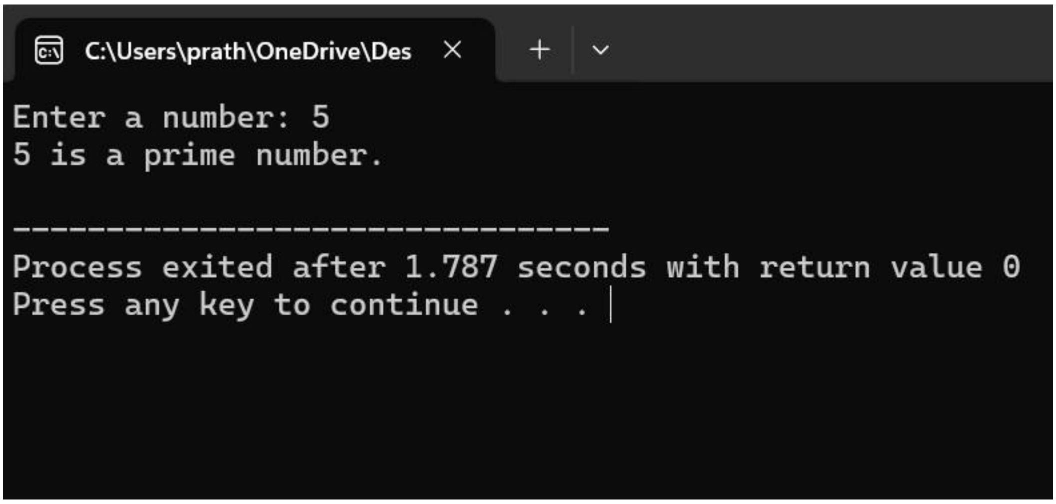
```
#include <iostream>
using namespace std;

void printRhombus(int n) {
    for (int i = 1; i <= n; i++) {
        for (int j = i; j < n; j++) {
            cout << " ";
        }
        for (int j = 1; j <= (2 * i - 1); j++) {
            cout << "*";
        }
        cout << endl;
    }

    for (int i = n - 1; i >= 1; i--) {
        for (int j = n; j > i; j--) {
            cout << " ";
        }
        for (int j = 1; j <= (2 * i - 1); j++) {
            cout << "*";
        }
        cout << endl;
    }
}

int main() {
    int n;
    cout << "Enter the number of rows for the rhombus: ";
    cin >> n;

    printRhombus(n);

    return 0;
    }
```



```
Enter the number of rows for the rhombus: 5
    *
   ***
  *****
 *******
*********
 *******
  *****
   ***
    *

--------------------------------
Process exited after 6.718 seconds with return value 0
Press any key to continue . . .
```

4. Write a function to determine if a given integer is a prime number or not.

```
#include <iostream>
using namespace std;

bool isPrime(int num) {
    if (num <= 1) {
        return false;
    }

    for (int i = 2; i * i <= num; i++) {
        if (num % i == 0) {
            return false;
        }
    }

    return true;
}

int main() {
    int number;

    cout << "Enter a number: ";
    cin >> number;

    if (isPrime(number)) {
        cout << number << " is a prime number." << endl;
    } else {
        cout << number << " is not a prime number." << endl;
    }

    return 0;
}
```

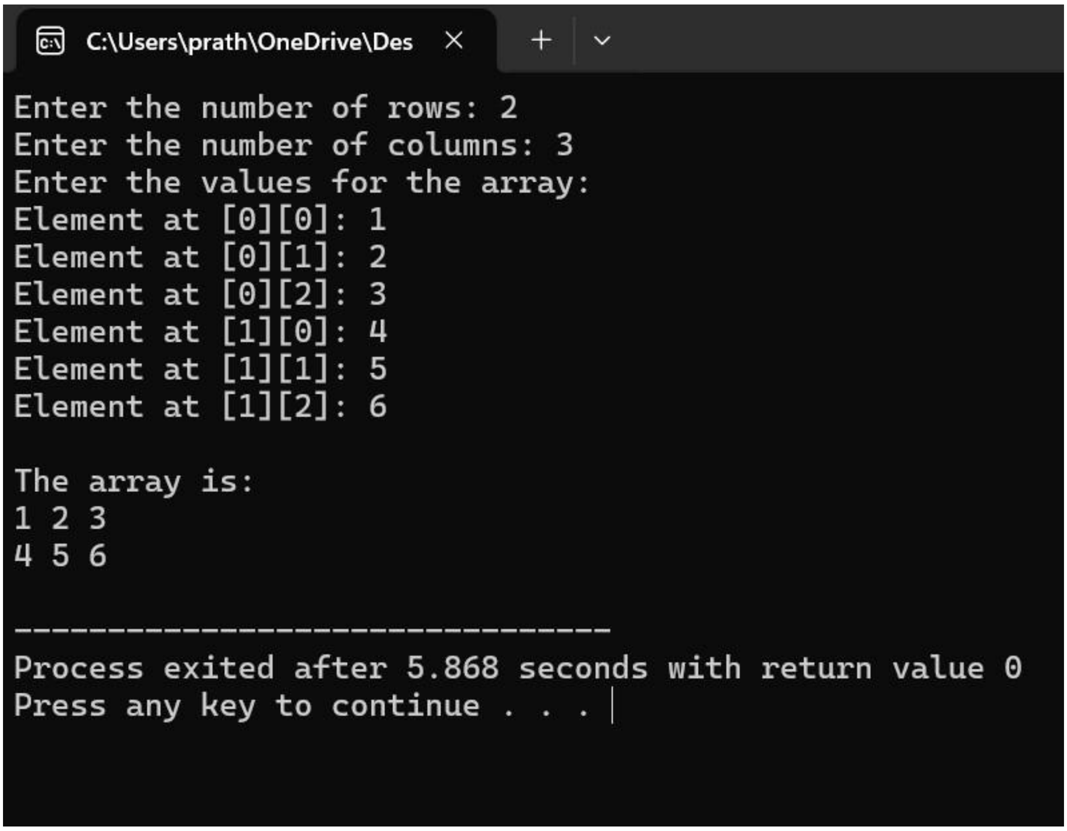


```
Enter a number: 5
5 is a prime number.

--------------------------------
Process exited after 1.787 seconds with return value 0
Press any key to continue . . .
```

5. write a c++ program to create a dynamic two-dimensional array using pointers and display its values.

Program:

```
#include <iostream>
using namespace std;

int main() {
    int rows, cols;

    cout << "Enter the number of rows: ";
    cin >> rows;
    cout << "Enter the number of columns: ";
    cin >> cols;

    int** arr = new int*[rows];
    for (int i = 0; i < rows; i++) {
        arr[i] = new int[cols];
    }

    cout << "Enter the values for the array:" << endl;
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            cout << "Element at [" << i << "][" << j << "]: ";
            cin >> arr[i][j];
        }
    }

    cout << "\nThe array is:" << endl;
    for (int i = 0; i < rows; i++) {
        for (int j = 0; j < cols; j++) {
            cout << arr[i][j] << " ";
        }
        cout << endl;
    }

    for (int i = 0; i < rows; i++) {
        delete[] arr[i];
    }
    delete[] arr;

    return 0;
}
```

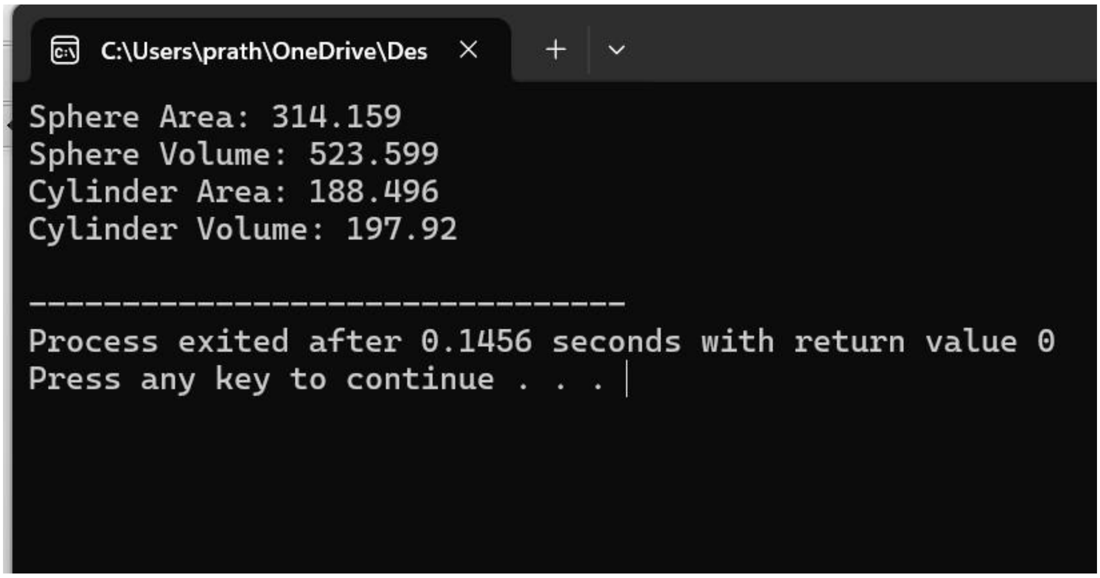


```
Enter the number of rows: 2
Enter the number of columns: 3
Enter the values for the array:
Element at [0][0]: 1
Element at [0][1]: 2
Element at [0][2]: 3
Element at [1][0]: 4
Element at [1][1]: 5
Element at [1][2]: 6

The array is:
1 2 3
4 5 6

--------------------------------
Process exited after 5.868 seconds with return value 0
Press any key to continue . . .
```

6. Create a base class called shape with virtual functions area() and volume().Derive a two classes sphere and cylinder from the base class.

Program:

```
#include <iostream>
#include <cmath>

using namespace std;

class Shape {
public:
    virtual double area() = 0;
    virtual double volume() = 0;
};

class Sphere : public Shape {
private:
    double radius;

public:
    Sphere(double r) : radius(r) {}

    double area() override {
        return 4 * M_PI * radius * radius;
    }

    double volume() override {
        return (4.0 / 3) * M_PI * pow(radius, 3);
    }
};

class Cylinder : public Shape {
private:
    double radius;
    double height;
public:
    Cylinder(double r, double h) : radius(r), height(h) {}

    double area() override {
        return 2 * M_PI * radius * (radius + height);
    }

    double volume() override {
        return M_PI * radius * radius * height;
    }
};

int main() {
    Sphere sphere(5.0);
    Cylinder cylinder(3.0, 7.0);

    cout << "Sphere Area: " << sphere.area() << endl;
    cout << "Sphere Volume: " << sphere.volume() << endl;

    cout << "Cylinder Area: " << cylinder.area() << endl;
    cout << "Cylinder Volume: " << cylinder.volume() << endl;

    return 0;
}
```

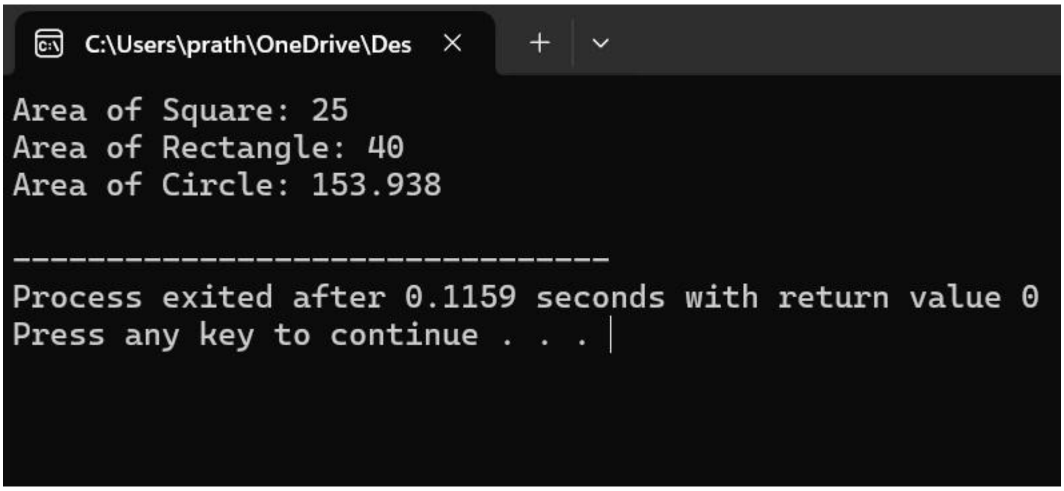


```
Sphere Area: 314.159
Sphere Volume: 523.599
Cylinder Area: 188.496
Cylinder Volume: 197.92

--------------------------------
Process exited after 0.1456 seconds with return value 0
Press any key to continue . . .
```

7. Write a c++ program to overload a function to calculate the area of as square ,a rectangle, and a circle separately.

Program:

```
#include <iostream>
#include <cmath>

using namespace std;

double area(double side) {
   return side * side;
}

double area(double length, double width) {
   return length * width;
}

double area(double radius, bool isCircle) {
   return M_PI * radius * radius;
}

int main() {
   double side = 5.0;
   double length = 10.0, width = 4.0;
   double radius = 7.0;

   cout << "Area of Square: " << area(side) << endl;

   cout << "Area of Rectangle: " << area(length, width) << endl;

   cout << "Area of Circle: " << area(radius, true) << endl;

   return 0;
}
```

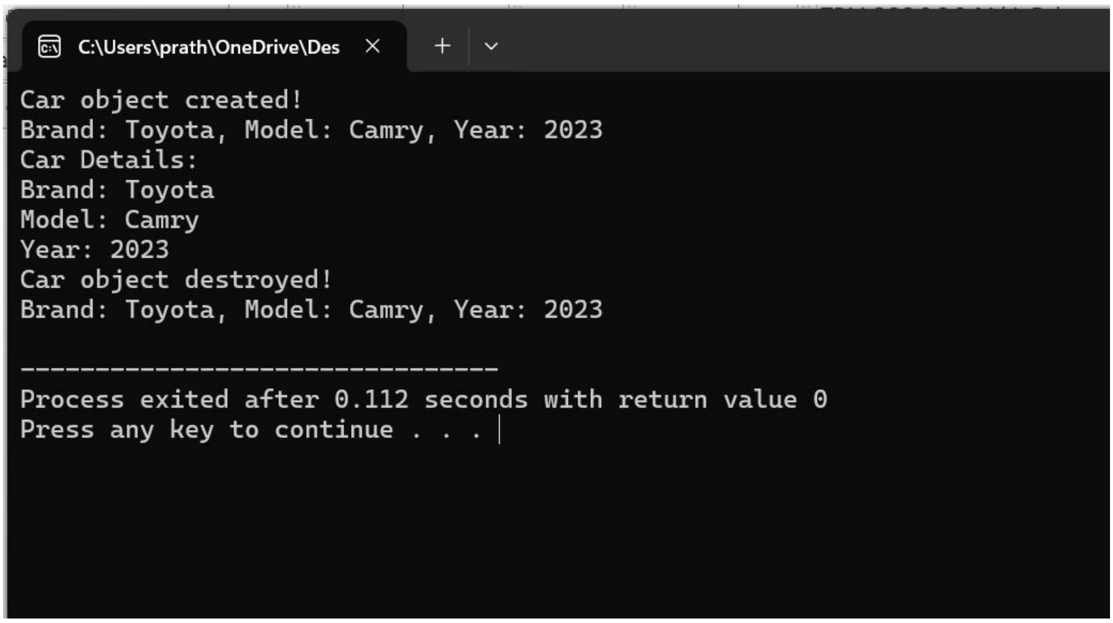


```
Area of Square: 25
Area of Rectangle: 40
Area of Circle: 153.938

--------------------------------
Process exited after 0.1159 seconds with return value 0
Press any key to continue . . .
```

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

Program:

```
#include <iostream>

using namespace std;

class Car {
private:
    string brand;
    string model;
    int year;

public:
    Car(string b, string m, int y) : brand(b), model(m), year(y) {
        cout << "Car object created!" << endl;
        cout << "Brand: " << brand << ", Model: " << model << ", Year: "
<< year << endl;
    }

    ~Car() {
        cout << "Car object destroyed!" << endl;
        cout << "Brand: " << brand << ", Model: " << model << ", Year: "
<< year << endl;
    }

    void displayDetails() {
        cout << "Car Details: " << endl;
        cout << "Brand: " << brand << endl;
        cout << "Model: " << model << endl;
        cout << "Year: " << year << endl;
    }
};

int main() {
    Car myCar("Toyota", "Camry", 2023);

    myCar.displayDetails();

    return 0;
}
```



```
Car object created!
Brand: Toyota, Model: Camry, Year: 2023
Car Details:
Brand: Toyota
Model: Camry
Year: 2023
Car object destroyed!
Brand: Toyota, Model: Camry, Year: 2023

--------------------------------
Process exited after 0.112 seconds with return value 0
Press any key to continue . . .
```