**Assignment 5: Virtual functions File Handling**

1. Create a class Shape with abstract methods area() and volume(). Derive classes Cube(side) and Cone(radius, height) from Shape and calculate area and volume for each shape.

#include <iostream>

#include <cmath>

using namespace std;

class Shape {

public:

  virtual double area() = 0;

  virtual double volume() = 0;

};

class Cube : public Shape {

private:

  double side;

public:

  Cube(double s) : side(s) {}

  double area() {

    return 6 \* side \* side;

  }

  double volume() {

    return side \* side \* side;

  }

};

class Cone : public Shape {

private:

  double radius;

  double height;

public:

  Cone(double r, double h) : radius(r), height(h) {}

  double area() {

    return M\_PI \* radius \* (radius + sqrt(height \* height + radius \* radius));

  }

  double volume() {

    return M\_PI \* radius \* radius \* height / 3;

  }

};

int main() {

  Cube c(2);

  Cone co(3, 4);

  cout << "Area of cube: " << c.area() << endl;

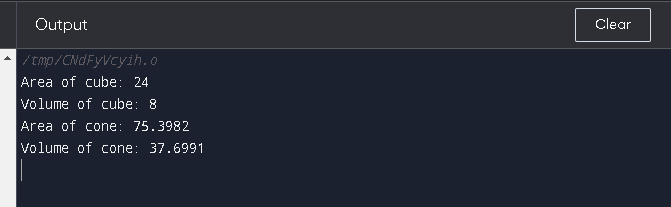
  cout << "Volume of cube: " << c.volume() << endl;

  cout << "Area of cone: " << co.area() << endl;

  cout << "Volume of cone: " << co.volume() << endl;

  return 0;

}



1. Create a class Author(name, category). Derive a class Book(Title, version, price) from A. Write print() method in both classes. Use method overriding. Show how to access the print using pointers.

#include <iostream>

#include <string>

using namespace std;

class Author

{

  private:

    string name;

    string category;

  public:

    Author(string name, string category)

    {

        this->name = name;

        this->category = category;

    }

    void print()

    {

        cout << "Author: " << name << " (" << category << ")" << endl;

    }

};

class Book : public Author

{

  private:

    string title;

    int version;

    double price;

  public:

    Book(string name, string category, string title, int version, double price)

        : Author(name, category)

    {

        this->title = title;

        this->version = version;

        this->price = price;

    }

    void print()

    {

        cout << "Book: " << title << " (version " << version << ", $" << price << ")" << endl;

    }

};

int main()

{

    Author \*author = new Author("Jane Smith", "Mystery");

    Book \*book = new Book("Jane Smith", "Mystery", "The Mysterious Case", 1, 19.99);

    author->print();

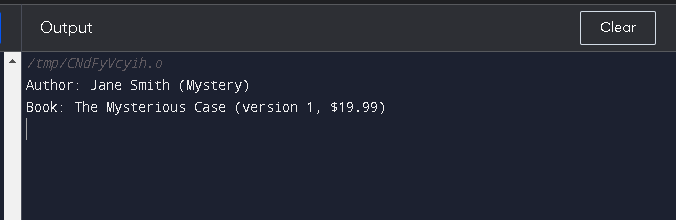
    book->print();

    // Deallocate the memory for the objects

    delete author;

    delete book;

    return 0;



1. Write a program to handle following file operations
   1. Read file name. Check if file exists.
   2. If file already exists read data from it and display it.
   3. Append data to the file.

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

int main() {

  // Read file name

  cout << "Enter file name: ";

  string fileName;

  getline(cin, fileName);

  // Check if file exists

  ifstream file(fileName);

  if (!file.good()) {

    cout << "File does not exist." << endl;

    return 1;

  }

  // Read data from file and display it

  cout << "Data from file:" << endl;

  string line;

  while (getline(file, line)) {

    cout << line << endl;

  }

  file.close();

  // Append data to the file

  cout << "Enter data to append to file: ";

  string dataToAppend;

  getline(cin, dataToAppend);

  ofstream outFile(fileName, ios\_base::app);

  outFile << dataToAppend << endl;

  outFile.close();

  return 0;

}

