Program 1:

#include <iostream>

#include <string>

using namespace std;

class Student {

protected:

string name;

int age;

public:

Student(string name, int age) : name(name), age(age) {}

void printDetails() {

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

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

}

};

class DetailedStudent : public Student {

private:

char grade;

public:

DetailedStudent(string name, int age, char grade) : Student(name, age), grade(grade) {}

void printDetails() {

Student::printDetails();

cout << "Grade: " << grade << endl;

}

};

int main() {

Student student1("John", 20);

DetailedStudent student2("Alice", 22, 'A');

cout << "Student 1 Details:" << endl;

student1.printDetails();

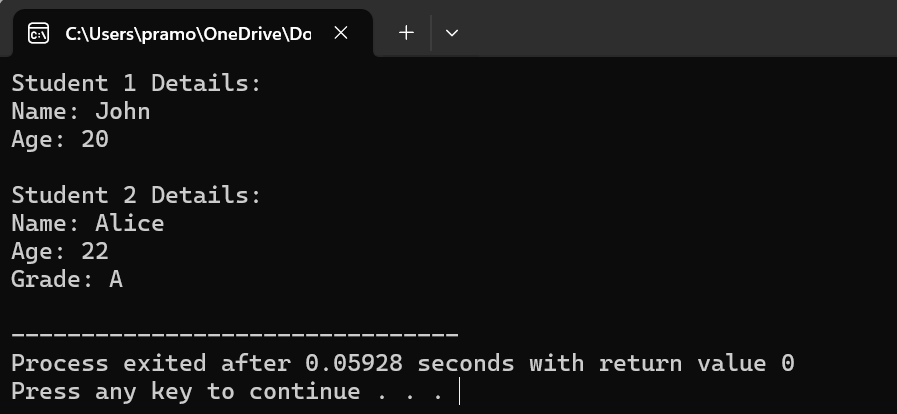
cout << endl;

cout << "Student 2 Details:" << endl;

student2.printDetails();

return 0;

}



Program 2:

#include <iostream>

#include <string>

using namespace std;

class Person {

private:

string name;

int age;

public:

Person(string name, int age) : name(name), age(age) {}

void printDetails() {

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

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

}

};

class Student : private Person {

private:

int studentID;

public:

Student(string name, int age, int studentID) : Person(name, age), studentID(studentID) {}

void printStudentDetails() {

cout << "Student ID: " << studentID << endl;

printDetails();

}

};

int main() {

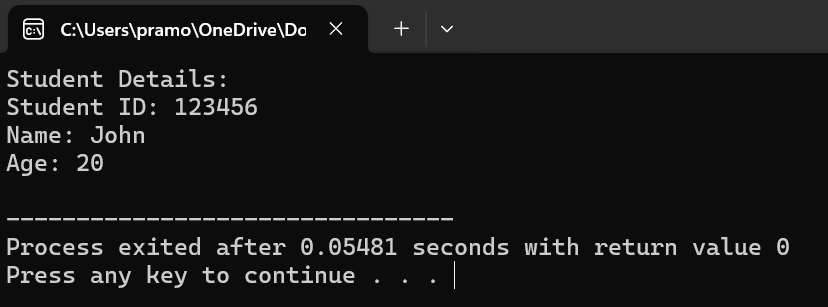
Student student("John", 20, 123456);

cout << "Student Details:" << endl;

student.printStudentDetails();

return 0;

}



Program 3:

#include <iostream>

int main() {

int num = 10;

double pi = 3.14;

char letter = 'A';

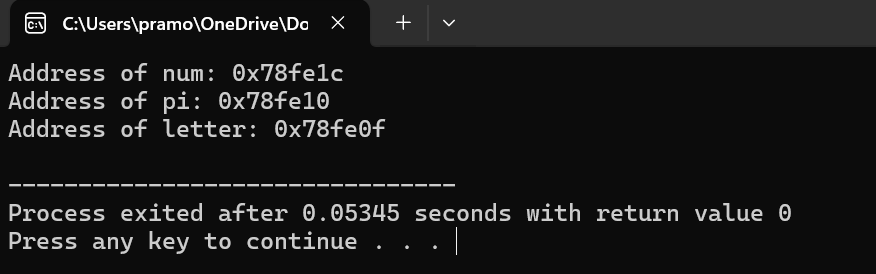
std::cout << "Address of num: " << &num << std::endl;

std::cout << "Address of pi: " << &pi << std::endl;

std::cout << "Address of letter: " << static\_cast<void\*>(&letter) << std::endl;

return 0;

}



Program 4:

#include <iostream>

using namespace std;

class Base1 {

public:

void displayBase1() {

cout << "Base1 class function" << endl;

}

};

class Base2 {

public:

void displayBase2() {

cout << "Base2 class function" << endl;

}

};

class Derived : public Base1, public Base2 {

public:

void displayDerived() {

cout << "Derived class function" << endl;

}

};

int main() {

Derived obj;

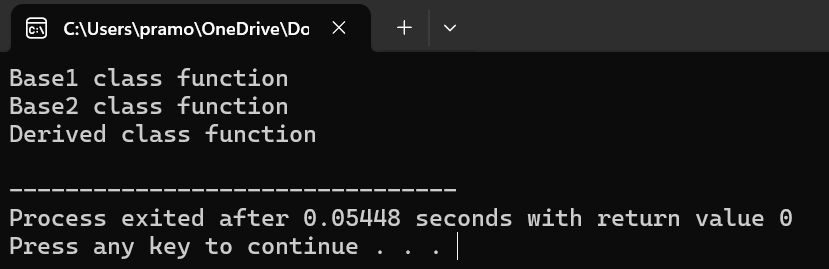
obj.displayBase1();

obj.displayBase2();

obj.displayDerived();

return 0;

}



Program 5:

#include <iostream>

class Shape {

public:

virtual float calculateArea() = 0;

};

class Square : public Shape {

private:

float side;

public:

Square(float s) : side(s) {}

float calculateArea() override {

return side \* side;

}

};

class Circle : public Shape {

private:

float radius;

public:

Circle(float r) : radius(r) {}

float calculateArea() override {

return 3.14159 \* radius \* radius;

}

};

int main() {

Square square(5);

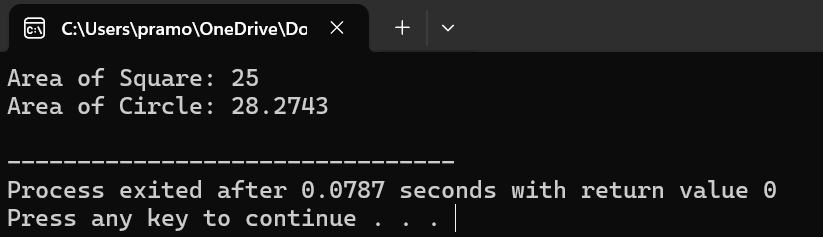
Circle circle(3);

std::cout << "Area of Square: " << square.calculateArea() << std::endl;

std::cout << "Area of Circle: " << circle.calculateArea() << std::endl;

return 0;

}



Program 6:

#include <iostream>

class MyClass {

private:

int value;

public:

MyClass(int value) : value(value) {}

void setValue(int value) {

this->value = value;

}

void printValue() {

std::cout << "Value: " << this->value << std::endl;

}

};

int main() {

MyClass obj(10);

std::cout << "Initial value:" << std::endl;

obj.printValue();

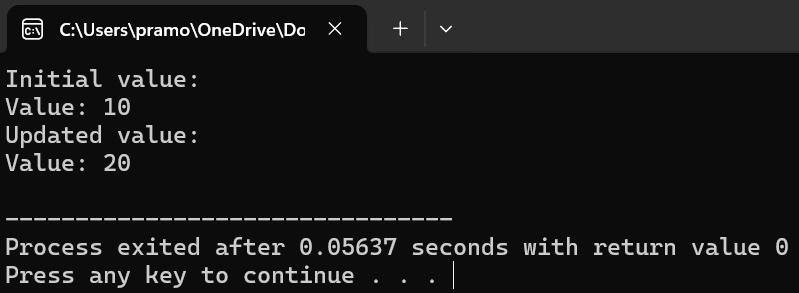
obj.setValue(20);

std::cout << "Updated value:" << std::endl;

obj.printValue();

return 0;

}



Program 7:

#include <iostream>

int main() {

int arr[] = {10, 20, 30, 40, 50};

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

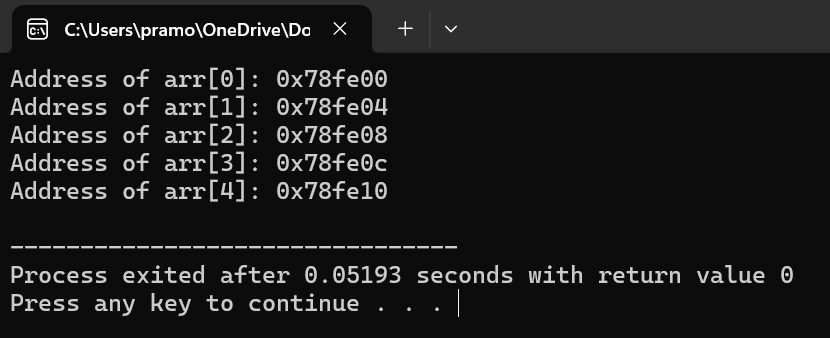
for (int i = 0; i < size; ++i) {

std::cout << "Address of arr[" << i << "]: " << &arr[i] << std::endl;

}

return 0;

}



Program 8:

#include <iostream>

class Number1 {

protected:

int num1;

public:

void setNum1(int n) {

num1 = n;

}

};

class Number2 {

protected:

int num2;

public:

void setNum2(int n) {

num2 = n;

}

};

class Sum : public Number1, public Number2 {

public:

int getSum() {

return num1 + num2;

}

};

int main() {

Sum sumObj;

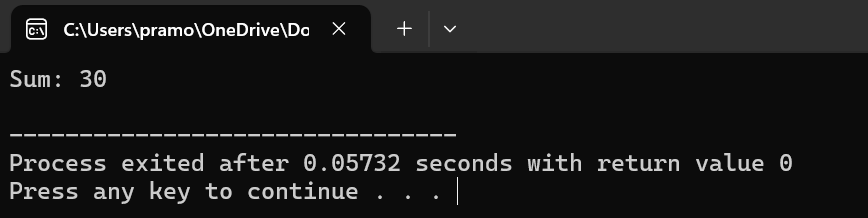
sumObj.setNum1(10);

sumObj.setNum2(20);

std::cout << "Sum: " << sumObj.getSum() << std::endl;

return 0;

}



Program 9:

#include <iostream>

class Rectangle {

private:

double length;

double width;

public:

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

double calculateArea() {

return length \* width;

}

};

int main() {

const int numRectangles = 3;

Rectangle rectangles[numRectangles] = {

Rectangle(5.0, 4.0),

Rectangle(6.0, 3.0),

Rectangle(7.0, 2.0)

};

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

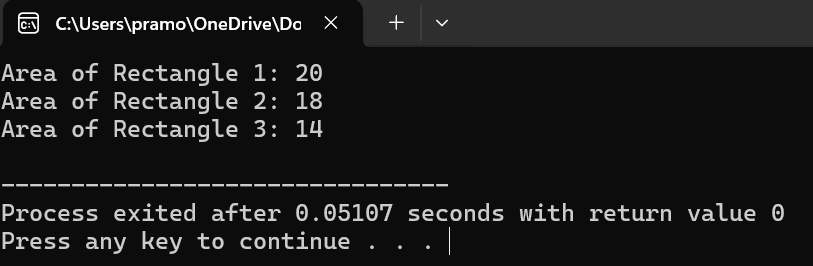
double area = rectangles[i].calculateArea();

std::cout << "Area of Rectangle " << i + 1 << ": " << area << std::endl;

}

return 0;

}



Program 10:

#include <iostream>

class MyClass {

private:

int value;

public:

MyClass(int val) : value(val) {}

void printValue() {

std::cout << "Value: " << value << std::endl;

}

};

int main() {

MyClass obj(42);

MyClass \*ptr = &obj;

ptr->printValue();

return 0;

}

