**Inheritance and pointers**

1. Create a base class called Shape with data members for height and width. Derive two classes Rectangle and Triangle from the base class. Write member functions to calculate the area and perimeter of each class

#include<iostream>

#include<cmath>

using namespace std;

class shape{

protected:

double height,width;

public:

shape(double h,double w){

height=h;

width=w;

}

virtual double area() const=0;

virtual double perimeter() const=0;

virtual~shape(){

}

};

class rectangle:public shape{

public:

rectangle(double h,double w):shape(h,w){

}

double area()const override{

cout<<"area of rectangle="<<(height\*width)<<endl;

}

double perimeter()const override{

cout<<"perimeter of rectangle="<<2\*(height+width)<<endl;

}

};

class triangle:public shape{

public:

triangle(double side1,double side2):shape(side1,side2){

}

double area()const override{

cout<<"area of triangle="<<height\*width<<endl;

}

double perimeter()const override{

double hypo=std::sqrt(height\*height+width\*width);

cout<<"perimeter of triangle="<<height+width+hypo<<endl;

}

};

int main(){

shape\* shape1=new rectangle(3,4);

shape\* shape2=new triangle(4,5);

shape1->area();

shape1->perimeter();

shape2->area();

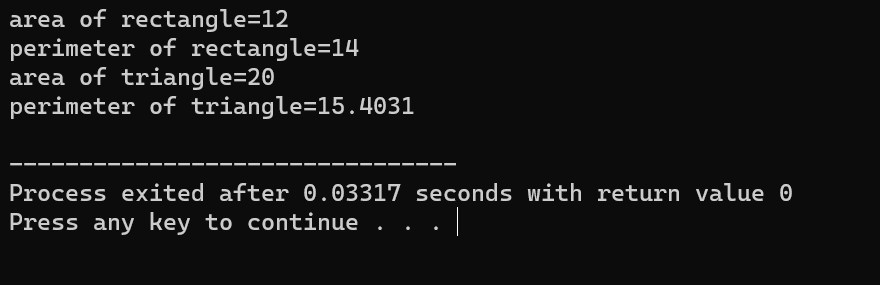
shape2->perimeter();

delete shape1;

delete shape2;

return 0;

}



1. Create a base class called vehicle with data members for make, model, and year. Derive two classes Car and Truck from the base class. The Car class should have additional data members for seating capacity and fuel type, while the Truck class should have additional data members for payload capacity and towing capacity. Write member functions to get and set the data members for each class.

#include <iostream>

#include <string>

using namespace std;

// Base class Vehicle

class Vehicle {

protected:

string make;

string model;

int year;

public:

Vehicle(string mk = "", string mdl = "", int yr = 0) : make(mk), model(mdl), year(yr) {}

// Setter functions

void setMake(string mk) { make = mk; }

void setModel(string mdl) { model = mdl; }

void setYear(int yr) { year = yr; }

// Getter functions

string getMake() { return make; }

string getModel() { return model; }

int getYear() { return year; }

};

// Derived class Car

class Car : public Vehicle {

private:

int seatingCapacity;

string fuelType;

public:

Car(string mk, string mdl, int yr, int seatCap, string fuel)

: Vehicle(mk, mdl, yr), seatingCapacity(seatCap), fuelType(fuel) {}

// Setter functions

void setSeatingCapacity(int seatCap) { seatingCapacity = seatCap; }

void setFuelType(string fuel) { fuelType = fuel; }

// Getter functions

int getSeatingCapacity() { return seatingCapacity; }

string getFuelType() { return fuelType; }

};

// Derived class Truck

class Truck : public Vehicle {

private:

double payloadCapacity;

double towingCapacity;

public:

Truck(string mk, string mdl, int yr, double payload, double towing)

: Vehicle(mk, mdl, yr), payloadCapacity(payload), towingCapacity(towing) {}

// Setter functions

void setPayloadCapacity(double payload) { payloadCapacity = payload; }

void setTowingCapacity(double towing) { towingCapacity = towing; }

// Getter functions

double getPayloadCapacity() { return payloadCapacity; }

double getTowingCapacity() { return towingCapacity; }

};

int main() {

Car car("Toyota", "Camry", 2020, 5, "Gasoline");

cout << "Car Make: " << car.getMake() << ", Model: " << car.getModel() << ", Year: " << car.getYear() << endl;

cout << "Seating Capacity: " << car.getSeatingCapacity() << ", Fuel Type: " << car.getFuelType() << endl;

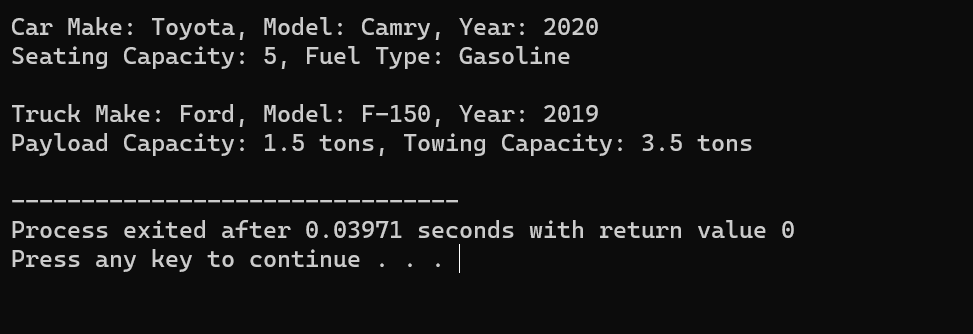
Truck truck("Ford", "F-150", 2019, 1.5, 3.5);

cout << "\nTruck Make: " << truck.getMake() << ", Model: " << truck.getModel() << ", Year: " << truck.getYear() << endl;

cout << "Payload Capacity: " << truck.getPayloadCapacity() << " tons, Towing Capacity: " << truck.getTowingCapacity() << " tons" << endl;

return 0;

}



1. Create a base class called Animal with data members for name, species, and age. Derive two classes Cat and Dog from the base class. The Cat class should have additional data members for color and breed, while the Dog class should have additional data members for weight and breed. Write member functions to get and set the data members for each class.

#include <iostream>

#include <string>

using namespace std;

// Base class Animal

class Animal {

protected:

string name;

string species;

int age;

public:

Animal(string n = "", string sp = "", int a = 0) : name(n), species(sp), age(a) {}

// Setter functions

void setName(string n) { name = n; }

void setSpecies(string sp) { species = sp; }

void setAge(int a) { age = a; }

// Getter functions

string getName() { return name; }

string getSpecies() { return species; }

int getAge() { return age; }

};

// Derived class Cat

class Cat : public Animal {

private:

string color;

string breed;

public:

Cat(string n, string sp, int a, string clr, string br)

: Animal(n, sp, a), color(clr), breed(br) {}

// Setter and Getter for color and breed

void setColor(string clr) { color = clr; }

void setBreed(string br) { breed = br; }

string getColor() { return color; }

string getBreed() { return breed; }

};

// Derived class Dog

class Dog : public Animal {

private:

double weight;

string breed;

public:

Dog(string n, string sp, int a, double w, string br)

: Animal(n, sp, a), weight(w), breed(br) {}

// Setter and Getter for weight and breed

void setWeight(double w) { weight = w; }

void setBreed(string br) { breed = br; }

double getWeight() { return weight; }

string getBreed() { return breed; }

};

int main() {

Cat cat("Whiskers", "Cat", 3, "White", "Persian");

cout << "Cat Name: " << cat.getName() << ", Species: " << cat.getSpecies() << ", Age: " << cat.getAge() << endl;

cout << "Color: " << cat.getColor() << ", Breed: " << cat.getBreed() << endl;

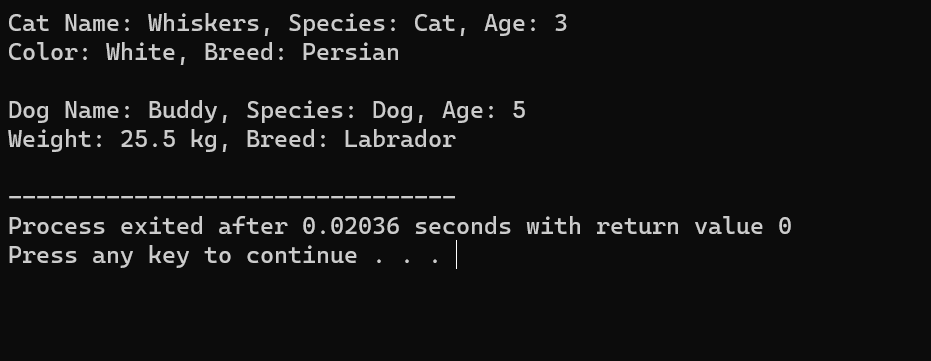
Dog dog("Buddy", "Dog", 5, 25.5, "Labrador");

cout << "\nDog Name: " << dog.getName() << ", Species: " << dog.getSpecies() << ", Age: " << dog.getAge() << endl;

cout << "Weight: " << dog.getWeight() << " kg, Breed: " << dog.getBreed() << endl;

return 0;

}



1. Create a base class called Employee with data members for name, d, and salary Derive two classes Manager and Engineer from the base class. The Manager class should have additional data members for department and bonus, while the Engineer class should have additional data members for specialty and hours. Write member functions to get and set the data members for each class.

#include <iostream>

#include <string>

using namespace std;

// Base class Employee

class Employee {

protected:

string name;

int id;

double salary;

public:

Employee(string n = "", int i = 0, double sal = 0.0) : name(n), id(i), salary(sal) {}

// Setter functions

void setName(string n) { name = n; }

void setID(int i) { id = i; }

void setSalary(double sal) { salary = sal; }

// Getter functions

string getName() { return name; }

int getID() { return id; }

double getSalary() { return salary; }

};

// Derived class Manager

class Manager : public Employee {

private:

string department;

double bonus;

public:

Manager(string n, int i, double sal, string dept, double b)

: Employee(n, i, sal), department(dept), bonus(b) {}

// Setter and Getter for department and bonus

void setDepartment(string dept) { department = dept; }

void setBonus(double b) { bonus = b; }

string getDepartment() { return department; }

double getBonus() { return bonus; }

};

// Derived class Engineer

class Engineer : public Employee {

private:

string specialty;

int hours;

public:

Engineer(string n, int i, double sal, string spec, int hrs)

: Employee(n, i, sal), specialty(spec), hours(hrs) {}

// Setter and Getter for specialty and hours

void setSpecialty(string spec) { specialty = spec; }

void setHours(int hrs) { hours = hrs; }

string getSpecialty() { return specialty; }

int getHours() { return hours; }

};

int main() {

Manager mgr("Alice", 101, 75000, "Sales", 5000);

cout << "Manager Name: " << mgr.getName() << ", ID: " << mgr.getID() << ", Salary: $" << mgr.getSalary() << endl;

cout << "Department: " << mgr.getDepartment() << ", Bonus: $" << mgr.getBonus() << endl;

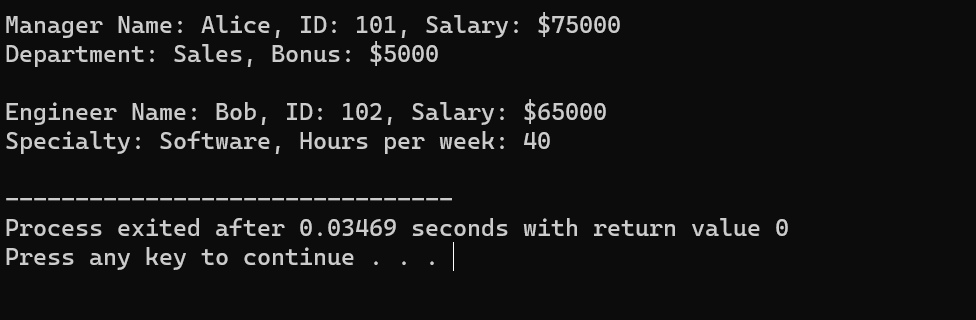
Engineer eng("Bob", 102, 65000, "Software", 40);

cout << "\nEngineer Name: " << eng.getName() << ", ID: " << eng.getID() << ", Salary: $" << eng.getSalary() << endl;

cout << "Specialty: " << eng.getSpecialty() << ", Hours per week: " << eng.getHours() << endl;

return 0;

}



1. Create a base class called Person with data members for name, age, and gender. Derive two classes Student and Teacher from the base class. The Student class should have additional data members for roll number and class, while the Teacher class should have additional data members for subject and salary. Write member functions to get and set the data members for each class.

#include <iostream>

#include <string>

using namespace std;

// Base class Person

class Person {

protected:

string name;

int age;

char gender;

public:

// Constructor

Person(string n = "", int a = 0, char g = 'M') : name(n), age(a), gender(g) {}

// Setter functions

void setName(string n) { name = n; }

void setAge(int a) { age = a; }

void setGender(char g) { gender = g; }

// Getter functions

string getName() { return name; }

int getAge() { return age; }

char getGender() { return gender; }

};

// Derived class Student

class Student : public Person {

private:

int rollNumber;

string className;

public:

// Constructor

Student(string n, int a, char g, int roll, string cls)

: Person(n, a, g), rollNumber(roll), className(cls) {}

// Setter functions

void setRollNumber(int roll) { rollNumber = roll; }

void setClassName(string cls) { className = cls; }

// Getter functions

int getRollNumber() { return rollNumber; }

string getClassName() { return className; }

};

// Derived class Teacher

class Teacher : public Person {

private:

string subject;

double salary;

public:

// Constructor

Teacher(string n, int a, char g, string subj, double sal)

: Person(n, a, g), subject(subj), salary(sal) {}

// Setter functions

void setSubject(string subj) { subject = subj; }

void setSalary(double sal) { salary = sal; }

// Getter functions

string getSubject() { return subject; }

double getSalary() { return salary; }

};

// Main function to test the classes

int main() {

// Create a Student object

Student student("John", 20, 'M', 101, "Computer Science");

cout << "Student Details:\n";

cout << "Name: " << student.getName() << ", Age: " << student.getAge() << ", Gender: " << student.getGender() << endl;

cout << "Roll Number: " << student.getRollNumber() << ", Class: " << student.getClassName() << endl;

// Create a Teacher object

Teacher teacher("Alice", 35, 'F', "Mathematics", 55000.00);

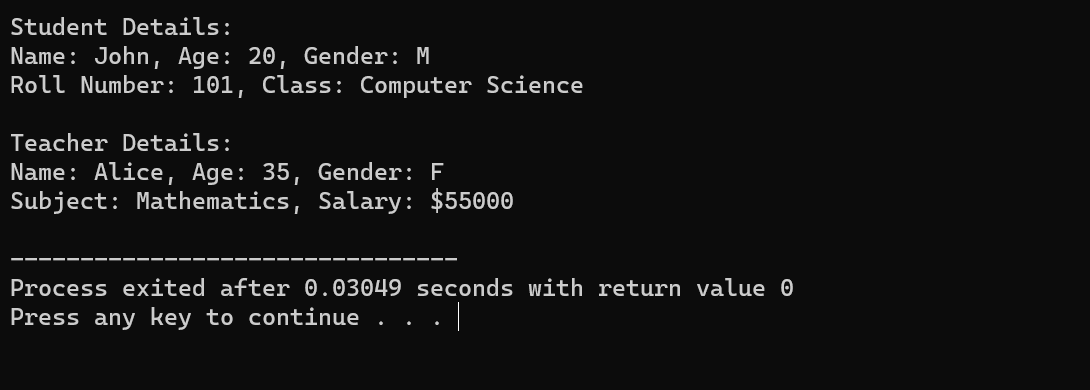
cout << "\nTeacher Details:\n";

cout << "Name: " << teacher.getName() << ", Age: " << teacher.getAge() << ", Gender: " << teacher.getGender() << endl;

cout << "Subject: " << teacher.getSubject() << ", Salary: $" << teacher.getSalary() << endl;

return 0;

}



1. Write a C++ program to create a pointer to an integer and display its value

#include<iostream>

using namespace std;

int main(){

int n=3;

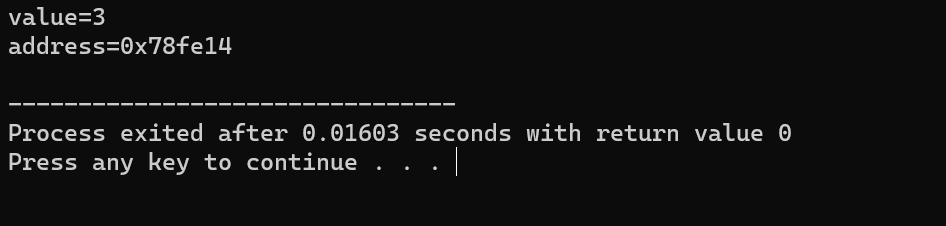
int \*ptr=&n;

cout<<"value="<<\*ptr<<endl;

cout<<"address="<<ptr<<endl;

return 0;

}



1. Write a C++ program to create a pointer to a float and display its value.

#include<iostream>

using namespace std;

int main(){

float n=3.32;

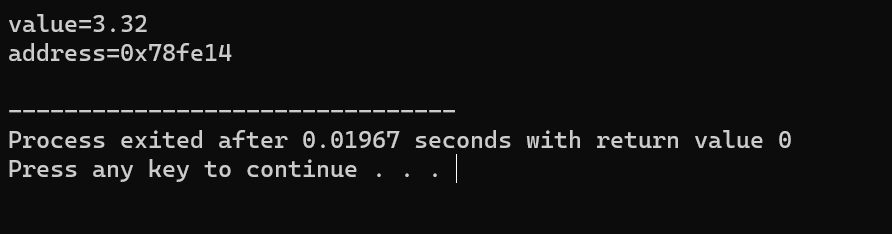
float \*ptr=&n;

cout<<"value="<<\*ptr<<endl;

cout<<"address="<<ptr<<endl;

return 0;

}



1. Write a C++ program to create a pointer to a char and display its value.

#include<iostream>

using namespace std;

int main(){

char n='a';

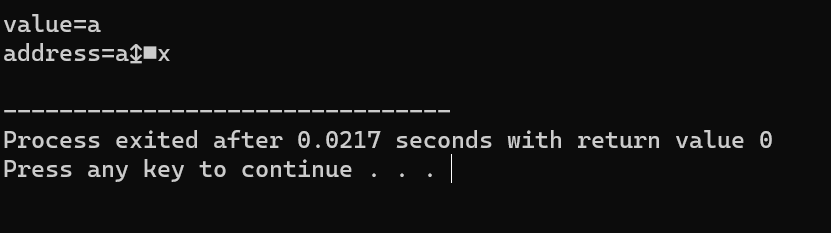
char \*ptr=&n;

cout<<"value="<<\*ptr<<endl;

cout<<"address="<<ptr<<endl;

return 0;

}



1. Write a C++ program to create a pointer to a double and display its value.

#include<iostream>

using namespace std;

int main(){

double n=4.23;

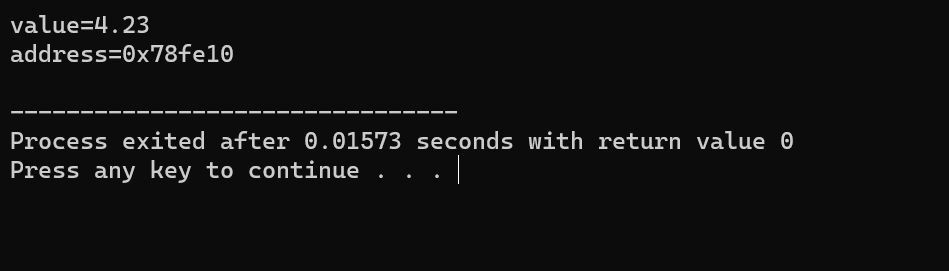
double \*ptr=&n;

cout<<"value="<<\*ptr<<endl;

cout<<"address="<<ptr<<endl;

return 0;

}



1. Write a C++ program to create a pointer to a string and display its value.

#include<iostream>

#include<string>

using namespace std;

int main(){

string s="hello";

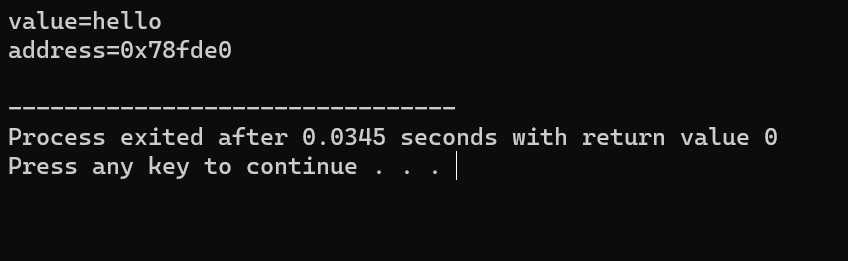
string \*ptr=&s;

cout<<"value="<<\*ptr<<endl;

cout<<"address="<<ptr<<endl;

return 0;

}



1. Write a C++ program to create a pointer to an array of elements and display its value.

#include<iostream>

using namespace std;

int main(){

int arr[]={1,2,3,4,5};

int \*ptr=arr;

for(int i=0;i<5;i++)

{

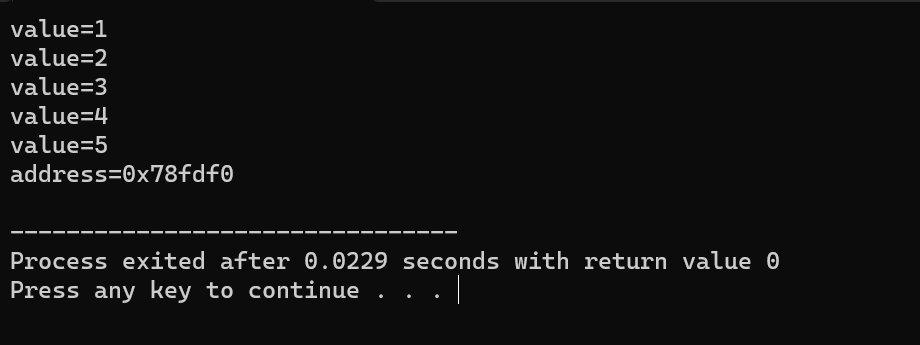
cout<<"value="<<\*ptr+i<<endl;

}

cout<<"address="<<ptr<<endl;

return 0;

}



1. Write a C++ program to create a pointer to an array of character and display its value.

#include<iostream>

using namespace std;

int main(){

char str[]="hello";

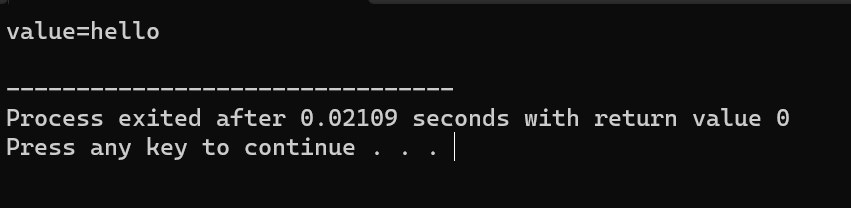
char \*ptr=str;

cout<<"value="<<ptr<<endl;

//cout<<"address="<<ptr<<endl;

return 0;

}



1. Write a C++ program to create a pointer to an array of floats and display its value.

#include<iostream>

using namespace std;

int main(){

float str[]={1.1f,2.2f,3.3f};

float \*ptr=str;

for(int i=0;i<3;i++){

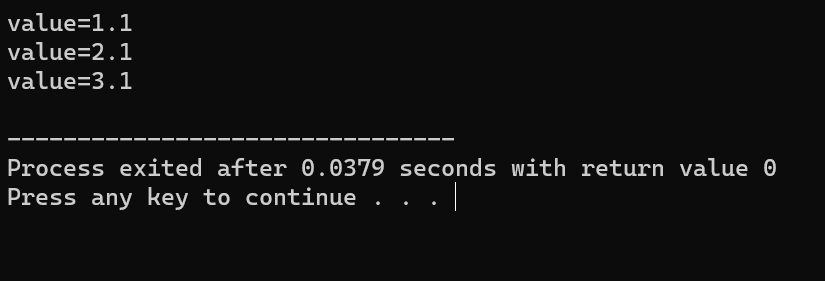
cout<<"value="<<\*ptr+i<<endl;

}

//cout<<"address="<<ptr<<endl;

return 0;

}



1. Write a C++ program to create a pointer to an object and display its attributes.

#include <iostream>

using namespace std;

class Person {

public:

string name;

int age;

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

void display() const {

cout << "Name: " << name << ", Age: " << age << endl;

}

};

int main() {

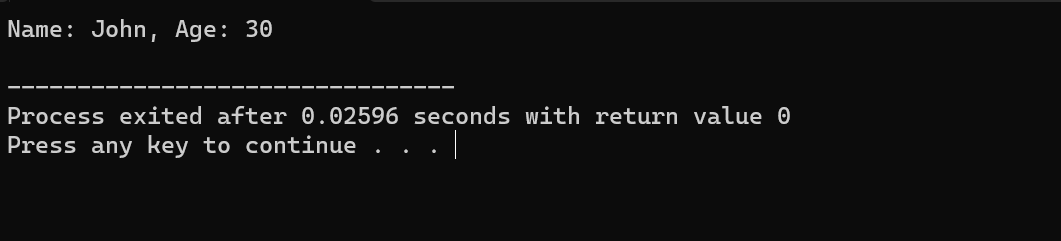
Person p("John", 30);

Person \*ptr = &p;

ptr->display();

return 0;

}



1. Write a C++ program to create a pointer to a function and call the function using the pointer.

#include <iostream>

using namespace std;

void sayHello() {

cout << "Hello from a function pointer!" << endl;

}

int main() {

void (\*funcPtr)() = sayHello;

funcPtr();

return 0;

}

