**Constructor and De-Constructor**

1. Write a C++ program to create a class for a student with a constructor and a destructor.

**Code:**

#include <iostream>

#include <string>

class Student {

private:

std::string name;

int age;

public:

Student(std::string n, int a) : name(n), age(a) {

std::cout << "Student object created: " << name << " (" << age << ")" << std::endl;

}

~Student() {

std::cout << "Student object destroyed: " << name << " (" << age << ")" << std::endl;

}

void displayInfo() {

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

}

};

int main() {

Student student1("John Doe", 20);

student1.displayInfo();

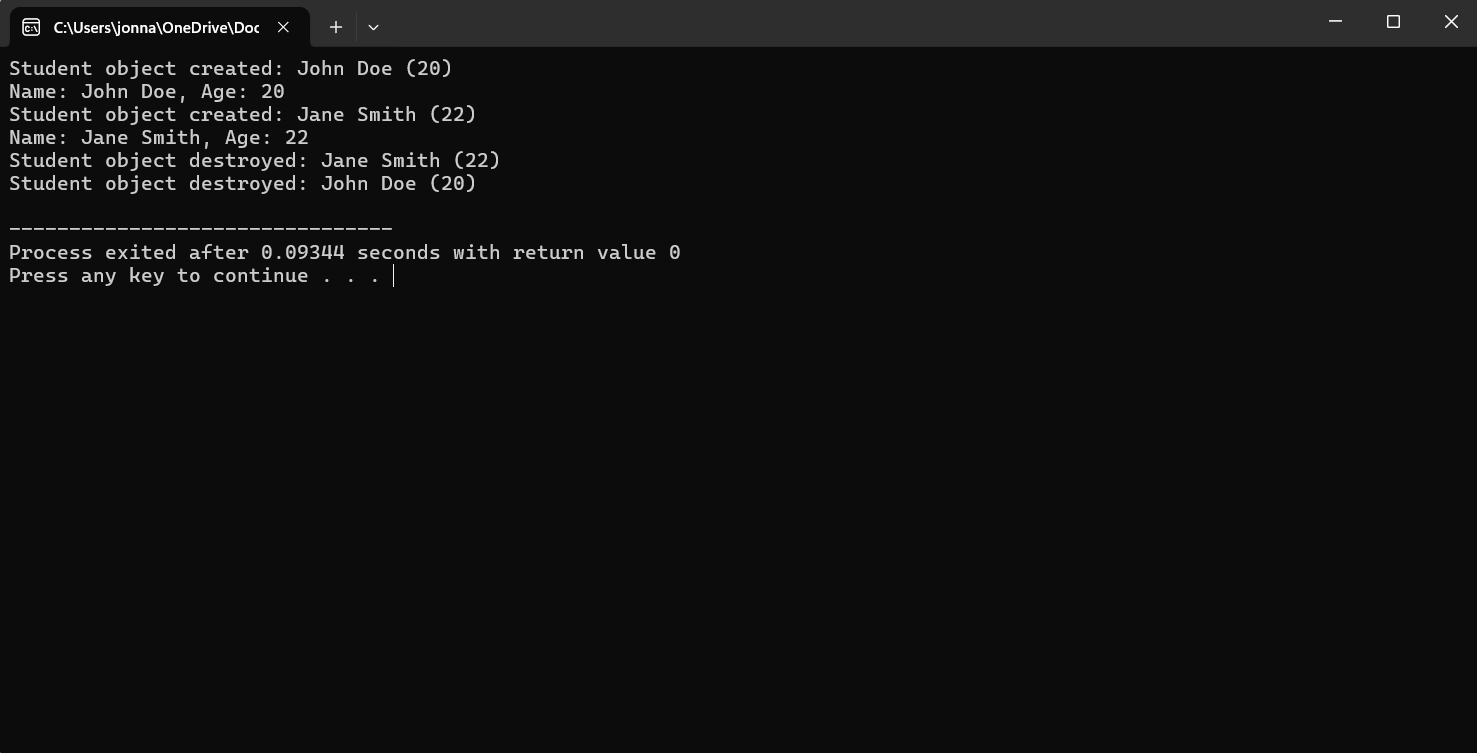
Student student2("Jane Smith", 22);

student2.displayInfo();

return 0;

}

**Output:**



1. Write a C++ program to create a class for a book with a constructor and a destructor.

**Code:**

**#include <iostream>**

**#include <string>**

**class Book {**

**private:**

**std::string title;**

**std::string author;**

**double price;**

**public:**

**Book(const std::string& bookTitle, const std::string& bookAuthor, double bookPrice) {**

**title = bookTitle;**

**author = bookAuthor;**

**price = bookPrice;**

**std::cout << "Book \"" << title << "\" by " << author << " created.\n";**

**}**

**~Book() {**

**std::cout << "Book \"" << title << "\" by " << author << " destroyed.\n";**

**}**

**void display() const {**

**std::cout << "Title: " << title << ", Author: " << author << ", Price: $" << price << "\n";**

**}**

**};**

**int main() {**

**Book book1("1984", "George Orwell", 9.99);**

**book1.display();**

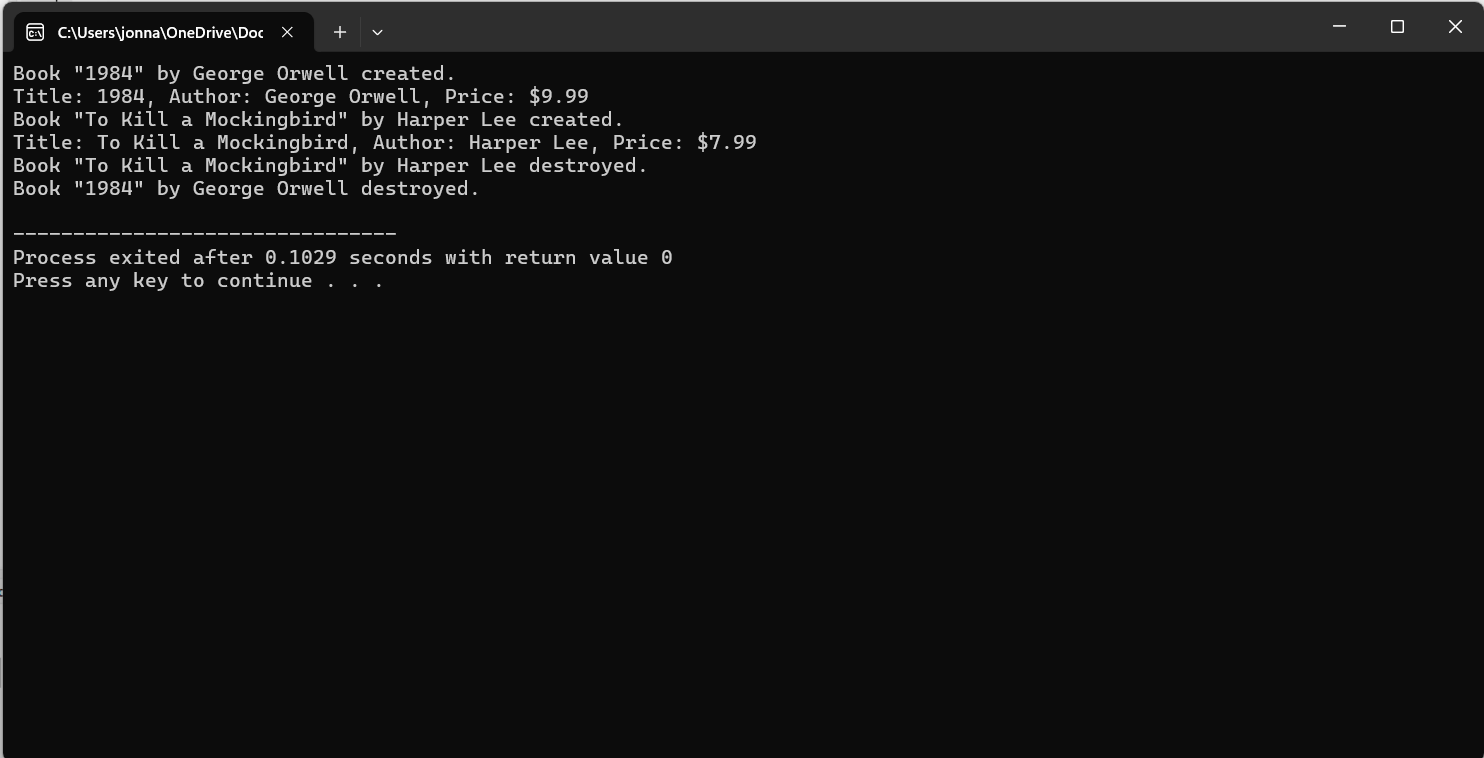
**Book book2("To Kill a Mockingbird", "Harper Lee", 7.99);**

**book2.display();**

**return 0;**

**}**

**Output:**



3.Write a C++ program to create a class for a rectangle with a constructor and a destructor.

**Code:**

**#include <iostream>**

**class Rectangle {**

**private:**

**double length;**

**double width;**

**public:**

**Rectangle(double rectLength, double rectWidth) {**

**length = rectLength;**

**width = rectWidth;**

**std::cout << "Rectangle with length " << length << " and width " << width << " created.\n";**

**}**

**~Rectangle() {**

**std::cout << "Rectangle with length " << length << " and width " << width << " destroyed.\n";**

**}**

**double area() const {**

**return length \* width;**

**}**

**double perimeter() const {**

**return 2 \* (length + width);**

**}**

**void display() const {**

**std::cout << "Length: " << length << ", Width: " << width**

**<< ", Area: " << area() << ", Perimeter: " << perimeter() << "\n";**

**}**

**};**

**int main() {**

**Rectangle rect1(10.0, 5.0);**

**rect1.display();**

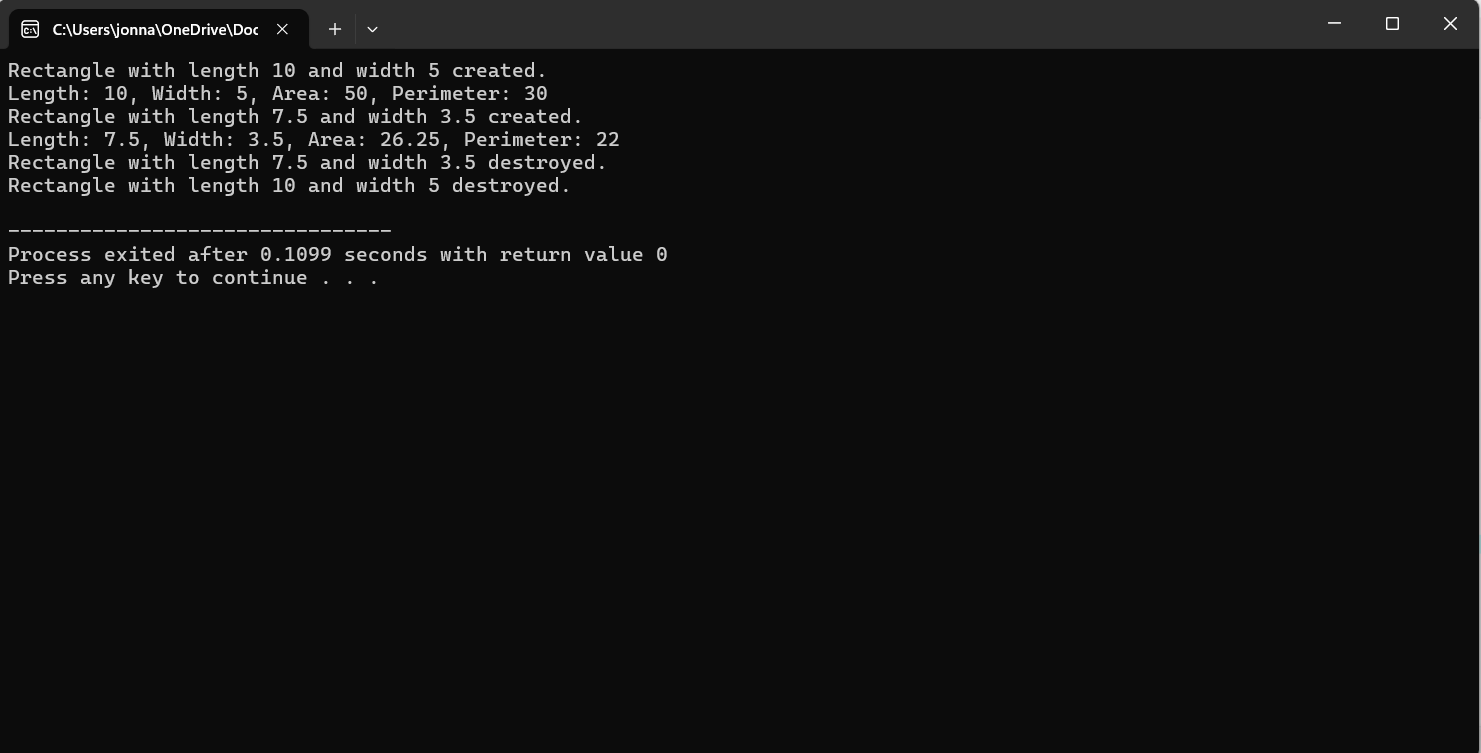
**Rectangle rect2(7.5, 3.5);**

**rect2.display();**

**return 0;**

**}**

**Output:**



**4**.Write a C++ program to create a class for a car with a constructor and a destructor.

**Code:**

**#include <iostream>**

**#include <string>**

**class Car {**

**private:**

**std::string make;**

**std::string model;**

**int year;**

**public:**

**Car(const std::string& carMake, const std::string& carModel, int carYear) {**

**make = carMake;**

**model = carModel;**

**year = carYear;**

**std::cout << "Car " << make << " " << model << " " << year << " created.\n";**

**}**

**~Car() {**

**std::cout << "Car " << make << " " << model << " " << year << " destroyed.\n";**

**}**

**void display() const {**

**std::cout << "Make: " << make << ", Model: " << model << ", Year: " << year << "\n";**

**}**

**};**

**int main() {**

**Car car1("Toyota", "Camry", 2020);**

**car1.display();**

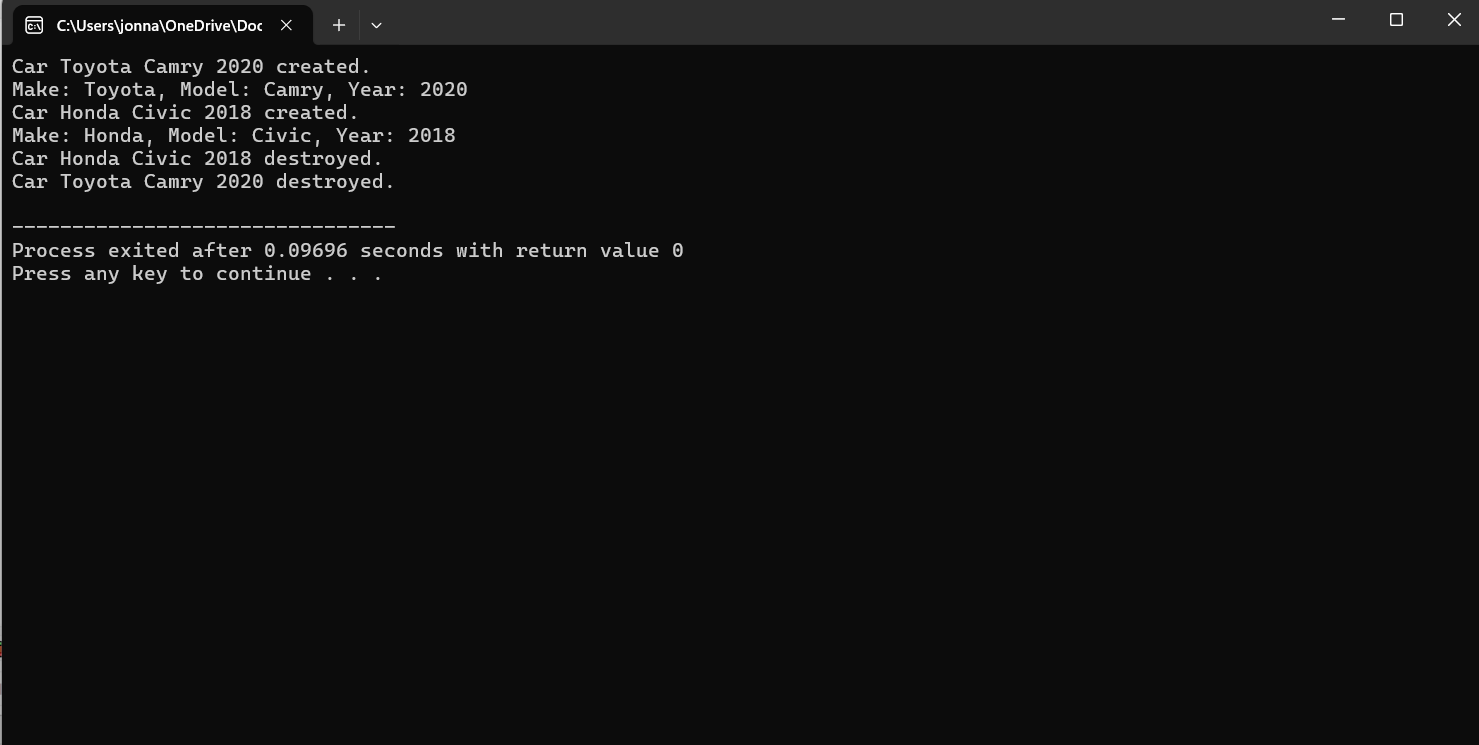
**Car car2("Honda", "Civic", 2018);**

**car2.display();**

**return 0;**

**}**

**Output:**



**5.Write a C++ program to create a class for a bank account with a constructor and a destructor.**

**Code:**

**#include <iostream>**

**#include <string>**

**class BankAccount {**

**private:**

**std::string accountHolder;**

**std::string accountNumber;**

**double balance;**

**public:**

**BankAccount(const std::string& holder, const std::string& number, double initialBalance) {**

**accountHolder = holder;**

**accountNumber = number;**

**balance = initialBalance;**

**std::cout << "Bank account for " << accountHolder << " with account number " << accountNumber << " created.\n";**

**}**

**~BankAccount() {**

**std::cout << "Bank account for " << accountHolder << " with account number " << accountNumber << " destroyed.\n";**

**}**

**void display() const {**

**std::cout << "Account Holder: " << accountHolder << ", Account Number: " << accountNumber << ", Balance: $" << balance << "\n";**

**}**

**void deposit(double amount) {**

**if (amount > 0) {**

**balance += amount;**

**std::cout << "Deposited $" << amount << " into account " << accountNumber << ". New balance: $" << balance << "\n";**

**} else {**

**std::cout << "Invalid deposit amount.\n";**

**}**

**}**

**void withdraw(double amount) {**

**if (amount > 0 && amount <= balance) {**

**balance -= amount;**

**std::cout << "Withdrew $" << amount << " from account " << accountNumber << ". New balance: $" << balance << "\n";**

**} else {**

**std::cout << "Invalid withdraw amount or insufficient funds.\n";**

**}**

**}**

**};**

**int main() {**

**BankAccount account1("Alice", "123456789", 1000.0);**

**account1.display();**

**account1.deposit(500.0);**

**account1.withdraw(200.0);**

**BankAccount account2("Bob", "987654321", 500.0);**

**account2.display();**

**account2.deposit(150.0);**

**account2.withdraw(100.0);**

**return 0;**

**}**

**Output:**

