**LAB 4**

1. **Raahim**

**24K-0543**

**Task 1:**

#include <iostream>

#include <stdbool.h>

#include <string>

using namespace std;

class Car{

private:

string Brand;

string Model;

float RentalPrice;

bool Availability;

public:

Car(){

Brand= "unknown";

Model= "Generic";

RentalPrice=0.0;

Availability= true;

}

void UpdateCarDetails(string BrandName,string ModelName,float RentPrice){

Brand = BrandName;

Model = ModelName;

RentalPrice=RentPrice;

}

void RentCar(){

if(Availability){

Availability= false;

cout<<"You have rented the car\n"<<endl;

}

else{

cout<<"The car is already Rented\n"<<endl;

}

}

void displayCarInfo() {

cout << "Car Brand: " << Brand <<endl;

cout << "Car Model: " << Model <<endl;

cout << "Rental Price: $" << RentalPrice <<endl;

cout << "Availability: " << (Availability ? "Car is Available" : "Sorry,The Car is Rented") <<endl;

}

};

int main(){

Car Product1;

cout<< "Car Details"<<endl;

Product1.displayCarInfo();

Product1.RentCar();

Product1.UpdateCarDetails("Honda","City",100.0);

Product1.displayCarInfo();

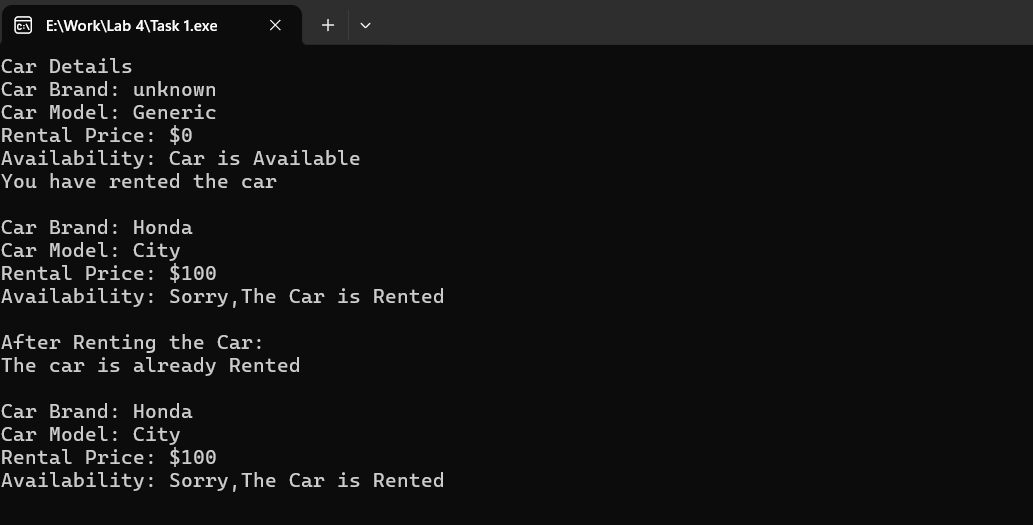
cout << "\nAfter Renting the Car:\n";

Product1.RentCar();

Product1.displayCarInfo();

}

**Output:**



**Task 2:**

#include <iostream>

#include <stdbool.h>

#include <string>

using namespace std;

class Car{

private:

string Brand;

string Model;

float RentalPrice;

bool Availability;

public:

Car(){

Brand= "unknown";

Model= "Generic";

RentalPrice=0.0;

Availability= true;

}

Car(string BrandName, string ModelName, float RentPrice, bool Avail) {

Brand = BrandName;

Model = ModelName;

RentalPrice = RentPrice;

Availability = Avail;

}

void UpdateCarDetails(string BrandName,string ModelName,float RentPrice){

Brand = BrandName;

Model = ModelName;

RentalPrice=RentPrice;

}

void RentCar(){

if(Availability){

Availability= false;

cout<<"You have rented the car\n"<<endl;

}

else{

cout<<"The car is already Rented\n"<<endl;

}

}

void displayCarInfo() {

cout << "Car Brand: " << Brand <<endl;

cout << "Car Model: " << Model <<endl;

cout << "Rental Price: $" << RentalPrice <<endl;

cout << "Availability: " << (Availability ? "Car is Available" : "Sorry,The Car is Rented") <<endl;

}

};

int main(){

Car Product1;

cout<< "Car Details"<<endl;

Product1.displayCarInfo();

Product1.RentCar();

Product1.UpdateCarDetails("Honda","City",100.0);

Product1.displayCarInfo();

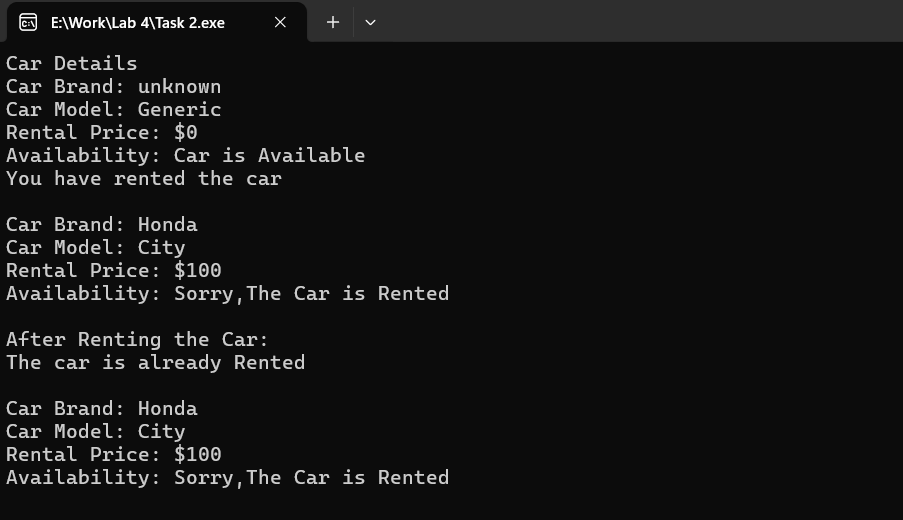
cout << "\nAfter Renting the Car:\n";

Product1.RentCar();

Product1.displayCarInfo();

}

**Output**:



**Task 3**:

#include <iostream>

#include <stdbool.h>

#include <string>

using namespace std;

class Car{

private:

string Brand;

string Model;

float RentalPrice;

bool Availability;

public:

Car(){

Brand= "unknown";

Model= "Generic";

RentalPrice=0.0;

Availability= true;

}

Car(string BrandName, string ModelName, float RentPrice, bool Avail) {

Brand = BrandName;

Model = ModelName;

RentalPrice = RentPrice;

Availability = Avail;

}

Car(const Car &c) {

Brand = c.Brand;

Model = c.Model;

RentalPrice = c.RentalPrice;

Availability = c.Availability;

}

~Car() {

cout << "Car object (" << Brand << " " << Model << ") is being deleted." << endl;

}

void UpdateCarDetails(string BrandName,string ModelName,float RentPrice){

Brand = BrandName;

Model = ModelName;

RentalPrice=RentPrice;

}

void displayCarInfo() {

cout << "\nCar Brand: " << Brand <<endl;

cout << "Car Model: " << Model <<endl;

cout << "Rental Price: $" << RentalPrice <<endl;

cout << "Availability: " << (Availability ? "Car is Available" : "Sorry,The Car is Rented") <<endl;

}

};

int main(){

Car carnumberone("Toyota", "Corolla", 50.0, true);

cout << "Original Car Details:" << endl;

carnumberone.displayCarInfo();

Car carnumbertwo = carnumberone;

carnumberone.UpdateCarDetails("Honda","City",100.0);

cout << "\nUpdated Car Details:" << endl;

carnumberone.displayCarInfo();

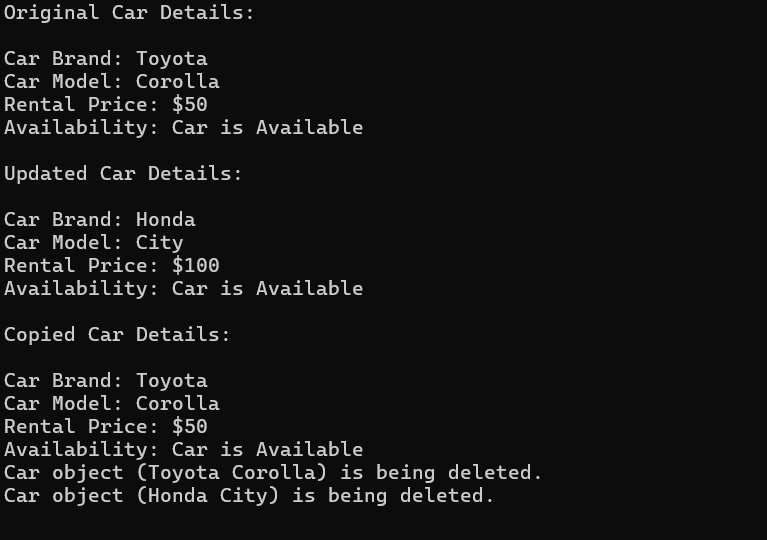
cout << "\nCopied Car Details:" << endl;

carnumbertwo.displayCarInfo();

return 0;

}

**Output:**



**Task 4:**

#include <iostream>

#include <stdbool.h>

#include <string>

using namespace std;

class Car{

private:

string Brand;

string Model;

float RentalPrice;

bool Availability;

int rentedDays;

float totalRevenue;

public:

Car(){

Brand= "unknown";

Model= "Generic";

RentalPrice=0.0;

Availability= true;

rentedDays = 0;

totalRevenue = 0.0;

}

Car(string BrandName, string ModelName, float RentPrice, bool Avail) {

Brand = BrandName;

Model = ModelName;

RentalPrice = RentPrice;

Availability = Avail;

rentedDays = 0;

totalRevenue = 0.0;

}

Car(const Car &c) {

Brand = c.Brand;

Model = c.Model;

RentalPrice = c.RentalPrice;

Availability = c.Availability;

rentedDays = c.rentedDays;

totalRevenue = c.totalRevenue;

}

~Car() {

cout << "Car object (" << Brand << " " << Model << ") is being deleted." << endl;

}

void UpdateCarDetails(string BrandName,string ModelName,float RentPrice){

Brand = BrandName;

Model = ModelName;

RentalPrice=RentPrice;

}

void RentCar(int days) {

if (Availability) {

Availability = false;

rentedDays += days;

totalRevenue += this->RentalPrice \* days;

cout << "You have rented the car for " << days << " days.\n" << endl;

} else {

cout << "The car is already rented.\n" << endl;

}

}

void displayCarInfo() {

cout << "\nCar Brand: " << Brand <<endl;

cout << "Car Model: " << Model <<endl;

cout << "Rental Price: $" << RentalPrice <<endl;

cout << "Availability: " << (Availability ? "Car is Available" : "Sorry,The Car is Rented") <<endl;

cout << "Total Revenue: $" << totalRevenue << endl;

}

};

int main() {

Car carnumberone("Toyota", "Corolla", 50.0, true);

cout << "Original Car Details:" << endl;

carnumberone.displayCarInfo();

carnumberone.RentCar(5);

Car carnumbertwo = carnumberone;

carnumberone.UpdateCarDetails("Honda", "City", 100.0);

cout << "\nUpdated Car Details:" << endl;

carnumberone.displayCarInfo();

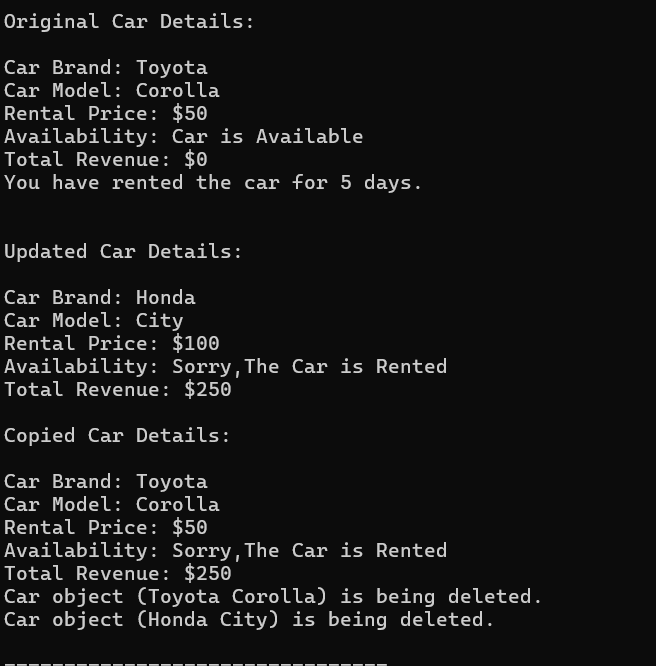
cout << "\nCopied Car Details:" << endl;

carnumbertwo.displayCarInfo();

return 0;

}

**Output:**



**Task 5:**

#include <iostream>

#include <string>

using namespace std;

class Car {

private:

string Brand;

string Model;

float RentalPrice;

bool Availability;

int rentedDays;

float totalRevenue;

string carID;

public:

Car() : carID("0000"), Brand("Unknown"), Model("Generic"), RentalPrice(0.0), Availability(true), rentedDays(0), totalRevenue(0.0) {}

Car(string ID, string BrandName, string ModelName, float RentPrice, bool Avail)

: carID(ID), Brand(BrandName), Model(ModelName), RentalPrice(RentPrice), Availability(Avail), rentedDays(0), totalRevenue(0.0) {}

Car(const Car &c) : carID(c.carID), Brand(c.Brand), Model(c.Model), RentalPrice(c.RentalPrice), Availability(c.Availability), rentedDays(c.rentedDays), totalRevenue(c.totalRevenue) {}

~Car() {

cout << "Car object (" << Brand << " " << Model << ") with ID " << carID << " is being deleted." << endl;

}

void UpdateCarDetails(string BrandName, string ModelName, float RentPrice) {

Brand = BrandName;

Model = ModelName;

RentalPrice = RentPrice;

}

void RentCar(int days) {

if (Availability) {

Availability = false;

rentedDays += days;

totalRevenue += RentalPrice \* days;

cout << "You have rented the car for " << days << " days.\n" << endl;

} else {

cout << "The car is already rented.\n" << endl;

}

}

void displayCarInfo() {

cout << "\nCar ID: " << carID << endl;

cout << "Car Brand: " << Brand << endl;

cout << "Car Model: " << Model << endl;

cout << "Rental Price: $" << RentalPrice << " per day" << endl;

cout << "Availability: " << (Availability ? "Car is Available" : "Car is Rented") << endl;

}

void displayFullCarInfo() {

displayCarInfo();

cout << "Total Days Rented: " << rentedDays << endl;

cout << "Total Revenue Generated: $" << totalRevenue << endl;

}

};

int main() {

Car carnumberone("A123", "Toyota", "Corolla", 50.0, true);

cout << "Original Car Details:" << endl;

carnumberone.displayFullCarInfo();

carnumberone.RentCar(5);

Car carnumbertwo = carnumberone;

carnumberone.UpdateCarDetails("Honda", "City", 100.0);

cout << "\nUpdated Car Details:" << endl;

carnumberone.displayFullCarInfo();

cout << "\nCopied Car Details:" << endl;

carnumbertwo.displayFullCarInfo();

return 0;

}

**Output:**

