**LAB 3**

**M. Raahim**

**24K-0543**

**Task 1:**

#include <iostream>

#include <string>

using namespace std;

class Wallet{

private:

string userName;

double currentBalance;

double startingBalance;

string transactionLog[100];

int transactionCount;

public:

Wallet(string name, double initBalance){

userName = name;

currentBalance = initBalance;

startingBalance = initBalance;

transactionCount = 0;

}

void deposit(double amount){

if (amount > 0){

currentBalance += amount;

transactionLog[transactionCount] = "Deposited: " + to\_string(amount);

transactionCount++;

cout << "Amount Deposited" << endl;

} else {

cout << "Invalid amount. Please enter a positive value." << endl;

}

}

void withdraw(double amount){

if(amount > currentBalance){

cout << "Insufficient funds." << endl;

} else if(amount < 0){

cout << "Error. Please enter a positive amount." << endl;

} else {

currentBalance -= amount;

transactionLog[transactionCount] = "Withdrawn: " + to\_string(amount);

transactionCount++;

cout << "Withdrawal Successful. Remaining Balance: " << currentBalance << endl;

}

}

void showTransactionHistory(){

cout << "Transaction History: " << endl << endl;

if(transactionCount == 0){

cout << "No transactions yet." << endl;

} else {

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

cout << transactionLog[i] << endl;

}

}

}

void checkLowBalance(){

if(currentBalance <= startingBalance / 2){

cout << "Warning: Low Balance" << endl;

} else {

cout << "Balance is sufficient" << endl;

}

}

void showBalance(){

cout << "Current Balance: " << currentBalance << endl;

}

};

int main(){

Wallet userWallet("Saad", 50000);

userWallet.deposit(10000);

userWallet.showBalance();

userWallet.withdraw(20000);

userWallet.withdraw(10000);

userWallet.showBalance();

userWallet.checkLowBalance();

userWallet.showTransactionHistory();

}

**Task 2**:

#include <iostream>

#include <string>

using namespace std;

class ActivityMonitor {

private:

string participantName;

int targetSteps;

int recordedSteps;

double burnedCalories;

public:

ActivityMonitor(string name, int stepTarget) {

participantName = name;

targetSteps = stepTarget;

recordedSteps = 0;

burnedCalories = 0.0;

}

void registerSteps(int steps) {

if (steps > 0) {

recordedSteps += steps;

burnedCalories += steps \* 0.05;

cout << steps << " steps logged. Total steps: " << recordedSteps << endl;

} else {

cout << "Invalid step count!" << endl;

}

}

void showProgress() {

cout << "Progress for " << participantName << ":\n";

cout << "Steps Taken: " << recordedSteps << endl;

cout << "Calories Burned: " << burnedCalories << endl;

cout << "Daily Step Goal: " << targetSteps << endl;

if (recordedSteps >= targetSteps) {

cout << "Congratulations! You've met your daily step goal!" << endl;

} else {

cout << "Keep going! You need " << (targetSteps - recordedSteps) << " more steps to reach your goal." << endl;

}

}

bool isGoalAchieved() {

return recordedSteps >= targetSteps;

}

};

int main() {

ActivityMonitor laibasMonitor("Laiba", 10000);

laibasMonitor.registerSteps(3000);

laibasMonitor.registerSteps(5000);

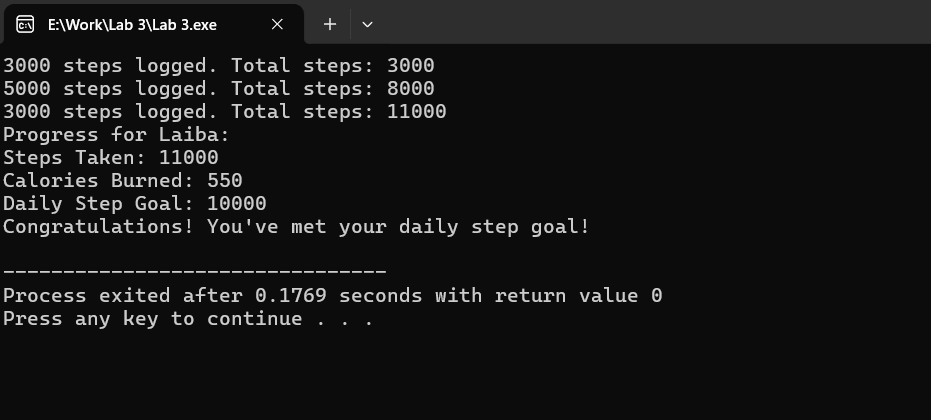
laibasMonitor.registerSteps(3000);

laibasMonitor.showProgress();

return 0;

}

**Output:**



**Task 3:**

#include <iostream>

#include <string>

using namespace std;

class Library {

private:

string bookArray[10];

string takenBooks[10];

int bookCount;

int takenCount;

public:

Library() : bookCount(0), takenCount(0) {}

void insertBook(string book) {

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

if (bookArray[i] == book) {

cout << "Book \"" << book << "\" already exists.\n";

return;

}

}

if (bookCount < 10) {

bookArray[bookCount++] = book;

cout << "Inserted book: " << book << endl;

} else {

cout << "Library is full!\n";

}

}

void takeBook(string book) {

if (takenCount >= 10) {

cout << "Cannot take more books!\n";

return;

}

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

if (bookArray[i] == book) {

takenBooks[takenCount++] = book;

for (int j = i; j < bookCount - 1; j++) {

bookArray[j] = bookArray[j + 1];

}

bookCount--;

cout << "Taken book: " << book << endl;

return;

}

}

cout << "Book \"" << book << "\" not found.\n";

}

void giveBackBook(string book) {

if (bookCount >= 10) {

cout << "Library is full!\n";

return;

}

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

if (takenBooks[i] == book) {

bookArray[bookCount++] = book;

for (int j = i; j < takenCount - 1; j++) {

takenBooks[j] = takenBooks[j + 1];

}

takenCount--;

cout << "Returned book: " << book << endl;

return;

}

}

cout << "Book \"" << book << "\" was not taken.\n";

}

void showBooks() {

cout << "\nBooks in Library:\n";

if (bookCount == 0) {

cout << "No books available.\n";

} else {

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

cout << "- " << bookArray[i] << endl;

}

}

}

void showTakenBooks() {

cout << "\nTaken Books:\n";

if (takenCount == 0) {

cout << "No books are taken.\n";

} else {

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

cout << "- " << takenBooks[i] << endl;

}

}

}

};

int main() {

Library hamzasLibrary;

hamzasLibrary.insertBook("C++ Programming");

hamzasLibrary.insertBook("Data Structures");

hamzasLibrary.insertBook("Algorithms");

hamzasLibrary.showBooks();

hamzasLibrary.takeBook("Data Structures");

hamzasLibrary.showBooks();

hamzasLibrary.showTakenBooks();

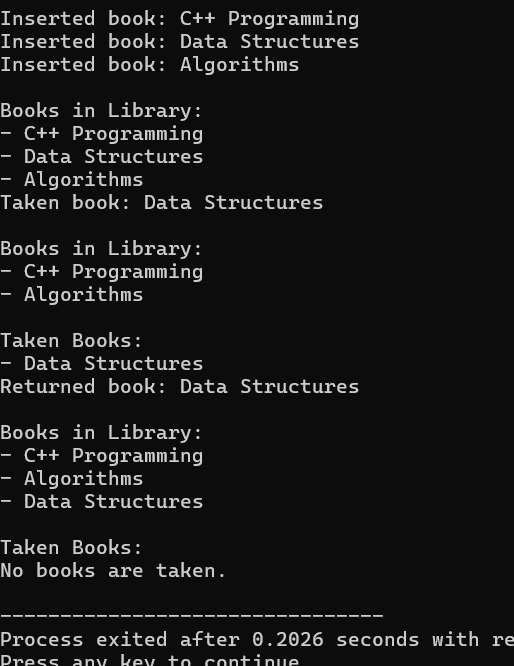
hamzasLibrary.giveBackBook("Data Structures");

hamzasLibrary.showBooks();

hamzasLibrary.showTakenBooks();

return 0;

}

**OUTPUT:**  


**Task 4:**

#include <iostream>

#include <string>

using namespace std;

class Vehicle {

private:

string make;

string type;

float tankSize;

float fuelLevel;

public:

Vehicle(string b, string m, float capacity) {

make = b;

type = m;

tankSize = capacity;

fuelLevel = capacity;

}

void move(float distance) {

float fuelUsed = distance \* 0.1;

if (fuelUsed > fuelLevel) {

cout << "Not enough fuel to move " << distance << " km!\n";

} else {

fuelLevel -= fuelUsed;

cout << "Moved " << distance << " km. Fuel used: " << fuelUsed << " liters.\n";

checkFuel();

}

}

void fillTank(float amount) {

if (amount <= 0) {

cout << "Invalid fuel amount!\n";

return;

}

if (fuelLevel + amount > tankSize) {

cout << "Tank overflow! Filling up to full capacity.\n";

fuelLevel = tankSize;

} else {

fuelLevel += amount;

cout << "Added " << amount << " liters of fuel.\n";

}

checkFuel();

}

void checkFuel() {

cout << "Current fuel level: " << fuelLevel << " liters.\n";

if (fuelLevel < 5) {

cout << "Warning: Low fuel! Please refill soon.\n";

}

}

void showVehicle() {

cout << "Vehicle: " << make << " " << type << "\n";

cout << "Tank Size: " << tankSize << " liters\n";

cout << "Current Fuel Level: " << fuelLevel << " liters\n";

}

};

int main() {

Vehicle zoyaVehicle("Honda", "city", 55);

zoyaVehicle.showVehicle();

zoyaVehicle.move(110);

zoyaVehicle.move(190);

zoyaVehicle.fillTank(20);

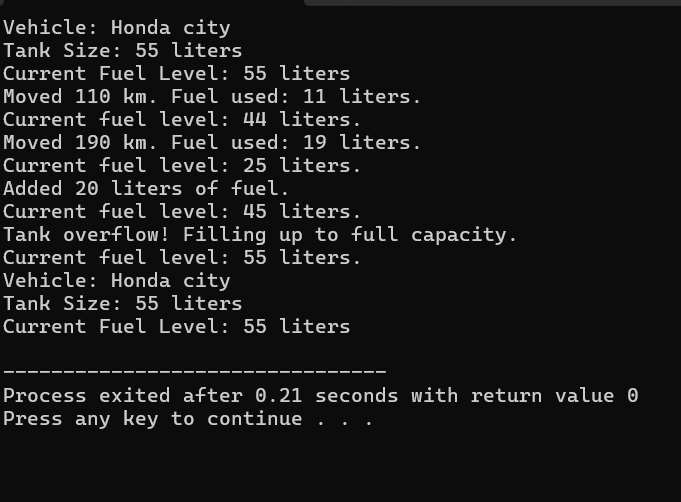
zoyaVehicle.fillTank(40);

zoyaVehicle.showVehicle();

return 0;

}

**Output:**



**Task 5:**

#include <iostream>

#include <string>

using namespace std;

class AudioPlayer {

private:

string tracklist[10];

string nowPlaying;

int trackCount;

public:

AudioPlayer() : trackCount(0), nowPlaying("No track playing") {}

void insertTrack(string track) {

if (trackCount < 10) {

tracklist[trackCount++] = track;

cout << "Inserted track: " << track << endl;

} else {

cout << "Tracklist is full! Cannot insert more tracks.\n";

}

}

void deleteTrack(string track) {

bool found = false;

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

if (tracklist[i] == track) {

found = true;

for (int j = i; j < trackCount - 1; j++) {

tracklist[j] = tracklist[j + 1];

}

trackCount--;

cout << "Deleted track: " << track << endl;

if (nowPlaying == track) {

nowPlaying = "No track playing";

}

break;

}

}

if (!found) {

cout << "Track \"" << track << "\" not found in tracklist.\n";

}

}

void startTrack(string track) {

bool found = false;

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

if (tracklist[i] == track) {

nowPlaying = track;

cout << "Now playing: " << track << endl;

found = true;

break;

}

}

if (!found) {

cout << "Track \"" << track << "\" is not in the tracklist.\n";

}

}

void showTracklist() {

cout << "\nTracklist:\n";

if (trackCount == 0) {

cout << "No tracks in the tracklist.\n";

} else {

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

cout << i + 1 << ". " << tracklist[i] << endl;

}

}

}

void nowPlayingTrack() {

cout << "\nNow Playing: " << nowPlaying << endl;

}

};

int main() {

AudioPlayer imransPlayer;

imransPlayer.insertTrack("Love Me Again");

imransPlayer.insertTrack("Christmas Tree");

imransPlayer.insertTrack("HayLoft");

imransPlayer.showTracklist();

imransPlayer.startTrack("Love Me Again");

imransPlayer.nowPlayingTrack();

imransPlayer.deleteTrack("HayLoft");

imransPlayer.showTracklist();

imransPlayer.startTrack("HayLoft");

return 0;

}

**Output**:

