**Week 2:**

**Task 01:**

#include <iostream>

#include <iomanip>

#include <unordered\_map>

#include <string>

using namespace std;

const int totalSeatsPerCoach = 80;

const int freeTicketThreshold = 10;

unordered\_map<string, int> passengersUp;

unordered\_map<string, int> passengersDown;

unordered\_map<string, double> moneyUp;

unordered\_map<string, double> moneyDown;

void setupScreen() {

cout <<" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout<< " Screen Display at the Start of the Day:" << endl;

cout <<" \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Journey\t\tAvailable Seats\t Money Collected" << endl;

for (int i = 9; i <= 15; i += 2) {

string time = (i < 10 ? "0" : "") + to\_string(i) + ":00";

passengersUp[time] = 0;

moneyUp[time] = 0.0;

cout << time << "\t\t" << totalSeatsPerCoach \* 6 << "\t\t\t$" << fixed << setprecision(2) << moneyUp[time] << endl;

}

for (int i = 10; i <= 16; i += 2) {

string time = (i < 10 ? "0" : "") + to\_string(i) + ":00";

passengersDown[time] = 0;

moneyDown[time] = 0.0;

int totalSeats = (i == 16) ? totalSeatsPerCoach \* 8 : totalSeatsPerCoach \* 6; // Extra coaches for the last train down

cout << time << "\t\t" << totalSeats << "\t\t\t$" << fixed << setprecision(2) << moneyDown[time] << endl;

}

}

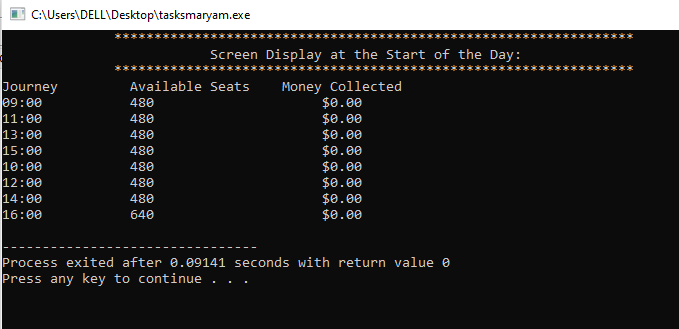
int main() {

setupScreen();

return 0;

}

**Output:**



**Task 2:**

#include <iostream>

#include <iomanip>

#include <unordered\_map>

#include <string>

#include <cmath>

using namespace std;

const int totalSeatsPerCoach = 80;

const int freeTicketThreshold = 10;

const double ticketPrice = 25.0;

unordered\_map<string, int> passengersUp;

unordered\_map<string, int> passengersDown;

unordered\_map<string, double> moneyUp;

unordered\_map<string, double> moneyDown;

void setupScreen() {

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " Screen Display at the Start of the Day:" << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Journey\t\tAvailable Seats\t Money Collected" << endl;

for (int i = 9; i <= 15; i += 2) {

string time = (i < 10 ? "0" : "") + to\_string(i) + ":00";

passengersUp[time] = 0;

moneyUp[time] = 0.0;

cout << time << "\t\t" << totalSeatsPerCoach \* 6 << "\t\t\t$" << fixed << setprecision(2) << moneyUp[time] << endl;

}

for (int i = 10; i <= 16; i += 2) {

string time = (i < 10 ? "0" : "") + to\_string(i) + ":00";

passengersDown[time] = 0;

moneyDown[time] = 0.0;

int totalSeats = (i == 16) ? totalSeatsPerCoach \* 8 : totalSeatsPerCoach \* 6;

cout << time << "\t\t" << totalSeats << "\t\t\t$" << fixed << setprecision(2) << moneyDown[time] << endl;

}

}

double calculateTotalPrice(int numPassengers, string journeyType, string time) {

double totalPrice = numPassengers \* ticketPrice;

if (numPassengers >= freeTicketThreshold) {

int numFreeTickets = floor(numPassengers / freeTicketThreshold);

totalPrice -= numFreeTickets \* ticketPrice;

}

if (journeyType == "up") {

moneyUp[time] += totalPrice;

passengersUp[time] += numPassengers;

} else {

moneyDown[time] += totalPrice;

passengersDown[time] += numPassengers;

}

return totalPrice;

}

void updateScreenDisplay(string journeyType, string time) {

int availableSeatsUp = totalSeatsPerCoach \* 6 - passengersUp[time];

int availableSeatsDown = (time == "16:00") ? totalSeatsPerCoach \* 8 - passengersDown[time] : totalSeatsPerCoach \* 6 - passengersDown[time];

cout << "Updated Screen Display:" << endl;

cout << "Journey\t\tAvailable Seats\t Money Collected" << endl;

cout << time << "\t\t" << ((journeyType == "up") ? availableSeatsUp : availableSeatsDown)

<< "\t\t\t$" << fixed << setprecision(2) << ((journeyType == "up") ? moneyUp[time] : moneyDown[time]) << endl;

}

void purchaseTickets(string journeyType, string time, int numPassengers) {

int availableSeats = (journeyType == "up") ? totalSeatsPerCoach \* 6 : totalSeatsPerCoach \* 8;

if (passengersUp[time] + passengersDown[time] + numPassengers <= availableSeats) {

double totalPrice = calculateTotalPrice(numPassengers, journeyType, time);

if (journeyType == "up") {

passengersUp[time] += numPassengers;

} else {

passengersDown[time] += numPassengers;

}

cout << "Tickets purchased for the " << time << " " << journeyType << " journey for "

<< numPassengers << " passengers. Total Price: $" << fixed << setprecision(2) << totalPrice << endl;

updateScreenDisplay(journeyType, time);

} else {

cout << "Tickets not available for the " << time << " " << journeyType << " journey." << endl;

}

}

void displayDailyTotals() {

cout << "\nDaily Totals:" << endl;

cout << "Total Passengers: " << passengersUp["09:00"] + passengersUp["11:00"] + passengersUp["13:00"] +

passengersUp["15:00"] + passengersDown["10:00"] + passengersDown["12:00"] +

passengersDown["14:00"] + passengersDown["16:00"] << endl;

cout << "Total Money Collected: $" << fixed << setprecision(2) <<

moneyUp["09:00"] + moneyUp["11:00"] + moneyUp["13:00"] + moneyUp["15:00"] +

moneyDown["10:00"] + moneyDown["12:00"] + moneyDown["14:00"] + moneyDown["16:00"] << endl;

string maxPassengersJourney;

int maxPassengers = 0;

for (const auto& entry : passengersUp) {

if (entry.second > maxPassengers) {

maxPassengers = entry.second;

maxPassengersJourney = entry.first + " up";

}

}

for (const auto& entry : passengersDown) {

if (entry.second > maxPassengers) {

maxPassengers = entry.second;

maxPassengersJourney = entry.first + " down";

}

}

cout << "Train journey with the most passengers: " << maxPassengersJourney << " (" << maxPassengers << " passengers)" << endl;

}

int main() {

setupScreen();

cout << "Enter the total number of tickets for the up journey: ";

int numPassengersUp;

cin >> numPassengersUp;

purchaseTickets("up", "09:00", numPassengersUp);

cout << "Enter the total number of tickets for the down journey: ";

int numPassengersDown;

cin >> numPassengersDown;

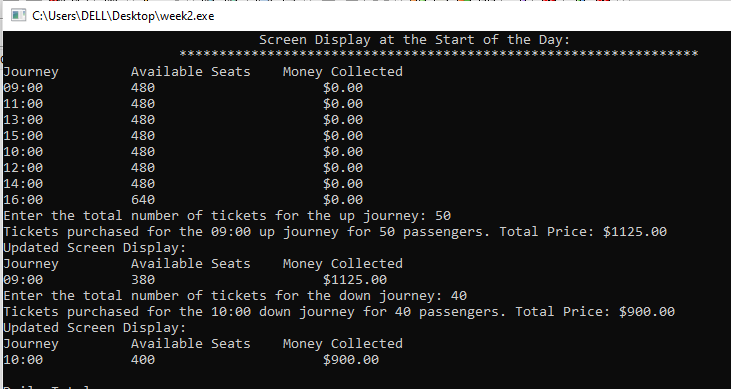
purchaseTickets("down", "10:00", numPassengersDown);

displayDailyTotals();

return 0;

}

**OUTPUT:**



**TASK 03:**

#include <iostream>

#include <iomanip>

#include <unordered\_map>

#include <string>

#include <cmath>

using namespace std;

const int totalSeatsPerCoach = 80;

const int freeTicketThreshold = 10;

const double ticketPrice = 25.0;

unordered\_map<string, int> passengersUp;

unordered\_map<string, int> passengersDown;

unordered\_map<string, double> moneyUp;

unordered\_map<string, double> moneyDown;

void setupScreen() {

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << " Screen Display at the Start of the Day:" << endl;

cout << " \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "Journey\t\tAvailable Seats\t Money Collected" << endl;

for (int i = 9; i <= 15; i += 2) {

string time = (i < 10 ? "0" : "") + to\_string(i) + ":00";

passengersUp[time] = 0;

moneyUp[time] = 0.0;

cout << time << "\t\t" << totalSeatsPerCoach \* 6 << "\t\t\t$" << fixed << setprecision(2) << moneyUp[time] << endl;

}

for (int i = 10; i <= 16; i += 2) {

string time = (i < 10 ? "0" : "") + to\_string(i) + ":00";

passengersDown[time] = 0;

moneyDown[time] = 0.0;

int totalSeats = (i == 16) ? totalSeatsPerCoach \* 8 : totalSeatsPerCoach \* 6;

cout << time << "\t\t" << totalSeats << "\t\t\t$" << fixed << setprecision(2) << moneyDown[time] << endl;

}

}

double calculateTotalPrice(int numPassengers, string journeyType, string time) {

double totalPrice = numPassengers \* ticketPrice;

if (numPassengers >= freeTicketThreshold) {

int numFreeTickets = floor(numPassengers / freeTicketThreshold);

totalPrice -= numFreeTickets \* ticketPrice;

}

if (journeyType == "up") {

moneyUp[time] += totalPrice;

passengersUp[time] += numPassengers;

} else {

moneyDown[time] += totalPrice;

passengersDown[time] += numPassengers;

}

return totalPrice;

}

void updateScreenDisplay(string journeyType, string time) {

int availableSeatsUp = totalSeatsPerCoach \* 6 - passengersUp[time];

int availableSeatsDown = (time == "16:00") ? totalSeatsPerCoach \* 8 - passengersDown[time] : totalSeatsPerCoach \* 6 - passengersDown[time];

cout << "Updated Screen Display:" << endl;

cout << "Journey\t\tAvailable Seats\t Money Collected" << endl;

cout << time << "\t\t" << ((journeyType == "up") ? availableSeatsUp : availableSeatsDown)

<< "\t\t\t$" << fixed << setprecision(2) << ((journeyType == "up") ? moneyUp[time] : moneyDown[time]) << endl;

}

void purchaseTickets(string journeyType, string time, int numPassengers) {

int availableSeats = (journeyType == "up") ? totalSeatsPerCoach \* 6 : totalSeatsPerCoach \* 8;

if (passengersUp[time] + passengersDown[time] + numPassengers <= availableSeats) {

double totalPrice = calculateTotalPrice(numPassengers, journeyType, time);

if (journeyType == "up") {

passengersUp[time] += numPassengers;

} else {

passengersDown[time] += numPassengers;

}

cout << "Tickets purchased for the " << time << " " << journeyType << " journey for "

<< numPassengers << " passengers. Total Price: $" << fixed << setprecision(2) << totalPrice << endl;

updateScreenDisplay(journeyType, time);

} else {

cout << "Tickets not available for the " << time << " " << journeyType << " journey." << endl;

}

}

void displayDailyTotals() {

cout << "\nDaily Totals:" << endl;

cout << "Total Passengers: " << passengersUp["09:00"] + passengersUp["11:00"] + passengersUp["13:00"] +

passengersUp["15:00"] + passengersDown["10:00"] + passengersDown["12:00"] +

passengersDown["14:00"] + passengersDown["16:00"] << endl;

cout << "Total Money Collected: $" << fixed << setprecision(2) <<

moneyUp["09:00"] + moneyUp["11:00"] + moneyUp["13:00"] + moneyUp["15:00"] +

moneyDown["10:00"] + moneyDown["12:00"] + moneyDown["14:00"] + moneyDown["16:00"] << endl;

string maxPassengersJourney;

int maxPassengers = 0;

for (const auto& entry : passengersUp) {

if (entry.second > maxPassengers) {

maxPassengers = entry.second;

maxPassengersJourney = entry.first + " up";

}

}

for (const auto& entry : passengersDown) {

if (entry.second > maxPassengers) {

maxPassengers = entry.second;

maxPassengersJourney = entry.first + " down";

}

}

cout << "Train journey with the most passengers: " << maxPassengersJourney << " (" << maxPassengers << " passengers)" << endl;

}

int main() {

setupScreen();

cout << "Enter the total number of tickets for the up journey: ";

int numPassengersUp;

cin >> numPassengersUp;

purchaseTickets("up", "09:00", numPassengersUp);

cout << "Enter the total number of tickets for the down journey: ";

int numPassengersDown;

cin >> numPassengersDown;

purchaseTickets("down", "10:00", numPassengersDown);

displayDailyTotals();

return 0;

}

OUTPUT:

