**MOD 3 C++ OOPS PROGRAMMING ASSESSMENT**

**EVENT MANAGEMENT SYSTEM**

#include <iostream>

#include <string>

#include <cmath>

#include <iomanip>

using namespace std;

/\*

Event Organiser Project

Implements cost estimation for an event using OOP concepts, functions, loops and conditionals.

\*/

class Event {

private:

string eventName;

string firstName;

string lastName;

int numberOfGuests;

int numberOfMinutes;

// Rate card (constants)

const double CostPerHour = 18.50;

const double CostPerMinute = 0.40;

const double CostOfDinner = 20.70;

public:

// Constructor

Event() : eventName(""), firstName(""), lastName(""), numberOfGuests(0), numberOfMinutes(0) {}

// Input functions

void readEventDetails() {

cout << "Enter the NAME OF THE EVENT (example: Wedding): ";

getline(cin, eventName);

cout << "Enter FIRST NAME: ";

getline(cin, firstName);

cout << "Enter LAST NAME: ";

getline(cin, lastName);

// Read and validate number of guests

while (true) {

cout << "Enter Number of Guests: ";

if (!(cin >> numberOfGuests) || numberOfGuests <= 0) {

cout << "Please enter a positive integer for number of guests.\n";

cin.clear();

cin.ignore(10000, '\n');

continue;

}

break;

}

// Read and validate number of minutes

while (true) {

cout << "Enter Number of Minutes for the event (e.g. 150): ";

if (!(cin >> numberOfMinutes) || numberOfMinutes < 0) {

cout << "Please enter a non-negative integer for number of minutes.\n";

cin.clear();

cin.ignore(10000, '\n');

continue;

}

break;

}

// Clear newline left in buffer

cin.ignore(10000, '\n');

}

// Compute number of servers required: 1 server per 20 guests (round up)

int computeNumberOfServers() const {

return static\_cast<int>(ceil(numberOfGuests / 20.0));

}

// Compute cost for one server based on minutes

double computeCostForOneServer() const {

int hours = numberOfMinutes / 60;

int minutes = numberOfMinutes % 60;

double cost1 = hours \* CostPerHour;

double cost2 = minutes \* CostPerMinute;

return cost1 + cost2;

}

// Compute total food cost

double computeTotalFoodCost() const {

return numberOfGuests \* CostOfDinner;

}

// Compute average cost per person

double computeAverageCostPerPerson() const {

if (numberOfGuests == 0) return 0.0;

return computeTotalFoodCost() / numberOfGuests;

}

// Compute total cost (food + servers)

double computeTotalCost() const {

int servers = computeNumberOfServers();

double costForOneServer = computeCostForOneServer();

return computeTotalFoodCost() + (costForOneServer \* servers);

}

// Compute deposit amount (25% of total cost)

double computeDepositAmount() const {

return computeTotalCost() \* 0.25;

}

// Display a formatted cost estimation

void displayEstimate() const {

cout << fixed << setprecision(2);

cout << "\n----- EVENT COST ESTIMATION -----\n";

cout << left << setw(25) << "Event Name" << ": " << eventName << '\n';

cout << left << setw(25) << "Client Name" << ": " << firstName << ' ' << lastName << '\n';

cout << left << setw(25) << "Number of Guests" << ": " << numberOfGuests << '\n';

cout << left << setw(25) << "Duration (minutes)" << ": " << numberOfMinutes << '\n';

int servers = computeNumberOfServers();

double costOneServer = computeCostForOneServer();

double totalFood = computeTotalFoodCost();

double avgCostPerPerson = computeAverageCostPerPerson();

double totalCost = computeTotalCost();

double deposit = computeDepositAmount();

cout << left << setw(25) << "Servers Required" << ": " << servers << '\n';

cout << left << setw(25) << "Cost for 1 Server" << ": $" << costOneServer << '\n';

cout << left << setw(25) << "Total Food Cost" << ": $" << totalFood << '\n';

cout << left << setw(25) << "Average Cost/Person" << ": $" << avgCostPerPerson << '\n';

cout << left << setw(25) << "TOTAL COST" << ": $" << totalCost << '\n';

cout << left << setw(25) << "DEPOSIT (25%)" << ": $" << deposit << '\n';

cout << "---------------------------------\n";

}

};

int main() {

cout << "=== Event Organiser Cost Estimator ===\n\n";

Event ev;

ev.readEventDetails();

ev.displayEstimate();

cout << "\nThank you! Please review the estimation above.\n";

return 0;

}