CSCI 3901: Software Development Concepts

Final Project

Name:Abhishek Latawa **Date:15 December,2023**

Overview:

The CSCI 3901 final project involves creating a Java-based system to manage information related to homeless shelters and camps in Halifax, addressing the increased homelessness issue. The system aims to provide insights into occupancy, funding, and inspection schedules. The project requires implementing various classes, including "HomelessSupport," along with support classes like "Point." The system manages shelters and camps, their services, staff, donors, and funding programs. It records occupancy, tracks donations, and offers functionality to answer specific queries. The key tasks include defining services, shelters, and camps, recording occupancy, managing staff, donors, and donations, and generating reports on occupancy variance, shelter capacities, donor activities, underfunded shelters, and inspection schedules. The system handles data persistently and demonstrates proficiency in software development techniques, database design, and algorithmic problem-solving.

Files and external data

The Implementation contains the one HomelessSupport class:

HomelessSupport:In this class we have implemented the 13 methods stated below:

boolean defineService(String serviceName, int inspectionFrequency):

The defineShelter method is part of a system managing homeless shelters and camps. It inserts or updates shelter information in a database, including the shelter's name, location, maximum capacity, and staff in charge. It checks if the shelter exists, handles exceptions, and utilizes helper methods for database interactions, such as resetting auto-increment and checking existing shelters. The method ensures data integrity and returns false in case of failure.

boolean defineShelter(String name, Point location, int maxCapacity, String staffInCharge)

The defineShelter method adds a shelter or camp to the system with the provided name, location, maximum capacity, and staff in charge. The location is specified in UTM coordinates (x, y) and measured in meters from a fixed origin. If a shelter with the same name already exists, a second call to the method updates the information for that shelter.

Purpose: Add a shelter or camp to the system or update existing shelter information.

Parameters:

name: Name of the shelter or camp.

location: Coordinates (x, y) in UTM format representing the location.

maxCapacity: Maximum occupancy capacity of the shelter or camp.

staffInCharge: Staff member responsible for the shelter or camp.

Functionality:

- Checks if the shelter already exists by calling the shelterExists method.
- If the shelter exists, updates its information using the updateShelter method.
- If the shelter does not exist, inserts a new record using the insertShelter method.
- Ensures data integrity by interacting with a database.

- Handles exceptions, logging and returning false in case of errors.
- Supports a second call to update information for an existing shelter.

Helper Methods:

Utilizes helper methods (**resetAutoIncrement**, **shelterExists**, **insertShelter**, and **shelterExistsAtLocation**) for specific tasks such as resetting auto-increment, checking existing shelters, inserting shelter information, and verifying shelter location.

Manages database interactions through prepared statements and SQL queries.

boolean serviceForShelter(String shelterName, String serviceName)

The serviceForShelter method is designed to identify and declare that a specific service is available at a given shelter or camp. Each service available at the shelter or camp can be registered by making a call to this method, effectively adding the service to the list of supported services.

Purpose: Identify and declare that a particular service is available at a shelter or camp.

Parameters:

- shelterName: Name of the shelter or camp where the service is being added.
- serviceName: Name of the service being declared as available.

Functionality:

- Registers the availability of a service at a specified shelter or camp.
- Each service available at a shelter or camp requires a separate call to this method.
- Supports the addition of multiple services to a shelter or camp over multiple calls.
- The method may interact with the underlying data structure or database to update service information.

Helper Methods

- Assumes that services are specific to individual shelters or camps.
- The method likely updates internal data structures or a database to track the availability of services at the specified shelter or camp.
- No return value is mentioned, but success or failure might be signaled through exceptions or a boolean return value in the actual implementation.

boolean declareShelterOccupancy(String name, String date, int occupancy)

The declareShelterOccupancy method is responsible for recording the daily occupancy or usage of a shelter or camp. It takes parameters such as the name of the shelter or camp, the specific date for which the occupancy is recorded, and the number of occupants on that day.

Purpose: Record the daily occupancy or usage of a shelter or camp.

Parameters:

- name: Name of the shelter or camp for which occupancy is being recorded.
- date: Specific date on which the occupancy is being recorded.
- occupancy: Number of occupants on the specified date.

Functionality:

- Records the daily occupancy of the given shelter or camp on the specified date.
- The occupancy data is recorded for each day of operation across a non-leap year (365 occupancy figures).
- The method may interact with the underlying data structure or database to store the occupancy information.

boolean addStaff(String name, Set<String> services, boolean volunteer, String manager)

The addStaff method is designed to declare the addition of a new staff member to the organization. It includes information such as the staff member's name, the set of services they are qualified to inspect, whether they are a volunteer or paid staff, and the name of their manager.

Purpose: Declare the addition of a new staff member to the organization**.**

Parameters:

- name: Name of the new staff member.
- services: Set of services that the staff member is qualified to inspect.
- volunteer: A boolean indicating whether the staff member is a volunteer (true) or paid staff (false).
- manager: Name of the manager overseeing the new staff member.

Functionality:

- Registers the details of a new staff member, including their name, qualified services, employment type (volunteer or paid), and manager.
- The set of services represents the specialized areas the staff member can inspect.
- The method may interact with the underlying data structure or database to store staff information.

boolean defineDonor(String name, Point centralOffice, String contact, Set<String> fundingPrograms)

The defineDonor method is responsible for identifying and defining a potential donor who will provide funding to support shelters or camps within the organization. Donors are characterized by their name, location of their central office, contact information, and a set of funding programs they offer.

Purpose: Identify and define a potential donor for funding shelters or camps.

Parameters:

- name: Name of the donor.

- centralOffice: Location (Point object) of the donor's central office in the city.
- contact: Contact information for the donor.
- fundingPrograms: Set of funding programs that the donor offers for supporting shelters or camps.

Functionality:

- Registers the details of a potential donor, including their name, central office location, contact information, and the set of funding programs they offer.
- Individuals, as donors, might provide a single cash donation, treated as a single funding program of philanthropy.
- The method may interact with the underlying data structure or database to store donor information.

boolean receiveDonation(String donor, String fundingProgram, String date, int donation)

The receiveDonation method is designed to record and manage the receipt of funding from a donor under a specific funding program on a particular date

Purpose: Record that a donor has provided funding under a given program on a specific date to the organization.

Parameters:

- donor: Name of the donor providing the funding.
- fundingProgram: Specific program under which the donation is made.
- date: Date on which the donation is received.
- donation: Amount of the donation.

Functionality:

- Records the details of a donation, including the donor's name, the specific funding program, the date of the donation, and the amount donated.
- The method may interact with the underlying data structure or database to store donation information.

boolean disburseFunds(String shelterReceiving, String date, int funds)

The disburseFunds method is responsible for recording and managing the disbursement of funds in support of the operations of a specific shelter or camp on a given date.

Purpose: Record the disbursement of funds to support the operations of a shelter or camp on a specific date.

Parameters:

shelterReceiving: Name of the shelter or camp receiving the funds.

date: Date on which the funds are disbursed.

funds: Amount of funds disbursed.

Functionality:

- Records the details of fund disbursement, including the name of the shelter or camp, the date of disbursement, and the amount of funds used.
- The method may interact with the underlying data structure or database to store disbursement information.

Set<String> shelterAtCapacity(int threshold)

The shelterAtCapacity method aims to report the names of all shelters or camps whose most recent occupancy report indicates that they are operating at or above a certain percentage of their capacity.

Here's a brief overview:

Purpose: Report shelters or camps operating at or above a specified percentage of their capacity based on the most recent occupancy report**.**

Parameters:

threshold: The threshold percentage, represented as an integer (e.g., 90 for 90%).

Return Type:

Set<String>: A set containing the names of shelters or camps meeting the specified condition.

Functionality:

- Analyzes the most recent occupancy reports for all shelters or camps.
- Identifies and collects the names of shelters or camps where the occupancy is at or above the specified threshold percentage of their capacity.
- The method may interact with the underlying data structure or database to retrieve and analyze occupancy data.

Set<String> occupancyVariance(String startDate, String endDate, int threshold)

The occupancyVariance method is designed to calculate and report the names of all shelters or camps whose occupancy reports, within a given date range (inclusive of both ends), show a variance equal to or greater than a specified threshold percentage.

Here's a brief overview:

Purpose: Report shelters or camps with occupancy variance meeting or exceeding a specified threshold.

Parameters:

startDate: Start date of the range for analyzing occupancy reports.

endDate: End date of the range for analyzing occupancy reports.

threshold: The threshold variance percentage.

Return Type:

Set<String>: A set containing the names of shelters or camps meeting the specified variance condition.

Functionality:

- Retrieves and analyzes occupancy reports for all shelters or camps within the specified date range.
- Calculates the occupancy variance for each shelter or camp within the date range.
- Identifies and collects the names of shelters or camps with an occupancy variance equal to or greater than the specified threshold.
- The method may interact with the underlying data structure or database to retrieve and analyze occupancy data.

void donorReport(String startDate, String endDate, PrintWriter outstream)

The donorReport method is responsible for generating and sending a report about the activity of each donor within a specified time range (inclusive of both endpoints). The report is sent to a PrintWriter outstream parameter.

Purpose: Generate and send a report about the activity of each donor within a given time range.

Parameters:

- startDate: Start date of the time range for the report.
- endDate: End date of the time range for the report.
- outstream: PrintWriter to which the report is sent.

Functionality:

- Analyzes donation activities of each donor within the specified time range.
- Generates a report for each donor, including their name and details about each funding program they participated in.
- For each donor, prints their name on a line by itself.
- For each funding program, prints a tab, the funding program's name, a tab, and the total funding provided under that program in the specified time range.
- Finishes the reporting block for a donor with a blank line.

- The method may interact with the underlying data structure or database to retrieve donation information.

Set<String> underfundedShelter(String startDate, String endDate, int distance, int threshold)

The underfundedShelter method is designed to identify and report the names of shelters or camps that are considered underfunded based on a set of criteria within a specified reporting period.

Purpose: Identify and report the names of underfunded shelters or camps.

Parameters:

- startDate: Start date of the reporting period.
- endDate: End date of the reporting period.
- distance: Maximum distance (in meters) from the donor's home office for funds allocation.
- threshold: The threshold percentage used to identify underfunded shelters or camps.

Return Type:

Set<String>: A set containing the names of underfunded shelters or camps.

Functionality:

- Determines the eligible shelters or camps based on the assumption that funds should go to those within a certain distance from the donor's home office.
- Calculates the funds allocation for each eligible shelter or camp based on the specified criteria (distance, proportion to capacities).
- Identifies the per-occupant capacities funding for each shelter or camp.

- Reports the names of shelters or camps with the lowest per-occupant capacities funding, meeting or exceeding the specified threshold, within the given reporting period.
- The method may interact with the underlying data structure or database to retrieve and analyze shelter, camp, and donation information.

Map<String, List<String>> inspectionSchedule(int scheduleDays, int inspectLimit)

The inspectionSchedule method is designed to generate an inspection schedule for staff members, considering services that need periodic inspection at shelters and camps. The schedule minimizes the number of inspections to be done within a specified timeframe and adheres to certain scheduling constraints.

Purpose: Generate a service inspection schedule for staff members inspecting services at shelters and camps.

Parameters:

- **scheduleDays**: The number of days for which the schedule is generated.
- **inspectLimit**: The maximum number of services a staff member can inspect in one day.

Return Type:

Map<String, List<String>>: A map where each key is a staff member's name, and the corresponding value is a list of inspection entries for each day in the schedule.

Functionality:

- Considers all services that need inspection at shelters and camps.
- Considers all staff members who can inspect these services.
- Assumes that all services have been inspected on the day the method is called.

- Defines a service inspection schedule for each staff member for the specified number of days.
- Each entry in the schedule represents the inspections to be done on that day, and the entries are formatted as strings containing shelter/service pairs.
- Staff members are limited to inspecting a maximum of inspectLimit services in one day.
- If a staff member has more than inspectLimit services to inspect on one day, the method shifts the inspection of some services to earlier days in the schedule.
- Aims to minimize the number of inspections to be done within the given timeframe under the specified scheduling constraints.

Assumptions:

- I have assumed that managers will not be considered as staff rather they will be treated as a separate staff. They will not be present in any DB table as Manager data will be provided at the time of the funcntion call.

Choices:

- I have used the helper methods to insert the data in the database tables rather than storing them in the different classes. Initially approach was to segregate the classes but it became complicated for me to work that way.
- I have made the names as the primary key in the DB Design rather then taking the Id's. I found it as an easy approach to retrieve the data from the database by taking the name as the primary key.
- I am making the db connection for every method I have created and closing the connection after every method

Key algorithms and design elements:

Database Design:

- Storing Persistent Information:
- Design Element: Ensured that information survives between program executions.

- Implementation: Utilized databases, files, or other persistent storage methods.

Exception Handling:

- **Design** Element: Handled errors and exceptional cases gracefully.
- **Implementation**: Using try-catch blocks to handle potential exceptions, such as invalid input or data inconsistencies.

Limitations:

- The assumption that each service requiring periodic inspection has at most one staff member might not reflect real-world scenarios. In reality, multiple staff members might be capable of inspecting the same service.
- The assumption that funds should go to shelters or camps within a certain distance might oversimplify donor preferences. In reality, donors may have more complex criteria for fund allocation.
- The calculation of occupancy variance as a percentage of capacity might oversimplify the assessment of shelters or camps' operational variability. Real-world factors influencing occupancy might not be fully captured.

References:

- ***[1]OpenAI, "ChatGPT," chat.openai.com, Oct. 25, 2023. https://chat.openai.com***
- **[2]"GeeksforGeeks | A computer science portal for geeks," *GeeksforGeeks*, 2019. https://www.geeksforgeeks.org**
- ***Youtube***
- ***Stack Overflow***