

# CP1890 – Assignment 3

**Handed Out:** 31 Mar 2024

**Due Date:** 14 Apr 2024

## Objectives

This problem set will build on your development, Github usage, proper versioning and code contribution skills in a professional environment while building up on what you learnt in this semester with the introduction of inheritance, polymorphism, handling databases, UI elements and handling objects.

## Collaboration

You will be working with your assigned group. Each group will designate a leader and the members will make commits to the leader's fork of the class by submitting a pull request and going through with the review process. You are free to divide the assignment objectives among your group members. Please note that each pull request must be reviewed with the two other members of your group.

Once the assignment is complete, **the leader of the group** will raise a pull request to merge the code with the main class repository and will assign the **other group members as reviewers for that pull request**.

## Submission

You will create one folder for your group under assignments with your group name. In this folder, you will have one python file for each question. You need to ensure that suitable comments are present in the code and the code has gone through the pull request and review process with your group. Once done, the leader should ensure the assignment link is posted to the drop box for the group that is provided on Brightspace **before the due date (no commits after the due date will be considered for grading)**.

### Late submissions

Penalties for late submissions is as follows:

- Up to 24 hours after the due date: Flat 25% penalty
- Beyond 24 hours up to 48 hours: Flat 50% penalty
- Beyond 48 hours: No points for assignment

If for some reason you are unable to submit the assignment on time, please reach out to me to make alternate arrangements – which will be handled based on the merits of the case. **No extensions will be provided.** Please ensure you stay on top of the requirements and manage your time well.

## Question 1

Create a program that imports customer data from a CSV file into a database table.

### Console

```
Customer Data Importer

CSV file:  customers.csv
DB file:   customers.sqlite
Table name: Customer

All old rows deleted from Customer table.
500 row(s) inserted into Customer table.
```

### Specifications

- Your instructor should provide you with the CSV and database files shown above (customers.csv and customers.sqlite). The SQLite database file should contain a table named Customer.
- The program should begin by deleting any old data from the Customer table. Then, it should insert all data from the customers.csv file into the Customer table of the SQLite database.

- The CSV file should be in this format:

```
first_name,last_name,company_name,address,city,state,zip
James,Butler,,6649 N Blue Gum St,New Orleans,LA,70116
Josephine,Darakjy, ,4 B Blue Ridge Blvd,Brighton,MI,48116
Art,Venere,,8 W Cerritos Ave #54,Bridgeport,NJ,08014
Lenna,Paprocki,Feltz Printing,639 Main St,Anchorage,AK,99501
```

- The Customer table should have the following columns and data types:

customerID	INTEGER PRIMARY KEY
firstName	TEXT
lastName	TEXT
companyName	TEXT
address	TEXT
city	TEXT
state	TEXT
zip	TEXT

- This program must complete within a few seconds. If it takes longer than that, you need to figure out how to improve its speed.
- Use DB Browser for SQLite to view the data and make sure that it has been added to the database correctly. In particular, check to make sure the database automatically generates the customer IDs.

## Question 2

Create a program that allows you to store the data for the players of a game.

### Console

```
Player Manager

COMMAND MENU
view - View players
add  - Add a player
del  - Delete a player
exit - Exit program

Command: view
Name           Wins    Losses    Ties    Games
-----
Mike           4       3        7       14
Joel           3       7       10       20

Command: add
Name: anne
Wins: 9
Losses: 5
Ties: 3
Anne was added to database.

Command: view
Name           Wins    Losses    Ties    Games
-----
Anne           9       5         3       17
Mike           4       3         7       14
Joel           3       7       10       20

Command: del
Name: anne
Anne was deleted from database.

Command: exit
Bye!
```

### Specifications

- Your instructor should provide you with a database file (players\_db.sqlite) that contains a Player table that stores the data for each player.
- Use the three-tier architecture (presentation, business, database) for this program, and store the code for each tier in a separate file.
- Display the players in order by wins, starting with the player with the most wins.
- Assume that the name for each player is unique.
- Make sure the user enters valid integers for wins, losses, and ties. In addition, make sure those integers aren't less than zero.

### Possible enhancement

- Add an "update" command. This command should prompt the user to enter the name of a player. Then, it should let the user update the wins, losses, and ties for the player.

### Question 3

Create a program that allows you to manage a task list that's stored in a database.

#### Console

```
Task List

COMMAND MENU
view      - View pending tasks
history   - View completed tasks
add       - Add a task
complete  - Complete a task
delete    - Delete a task
exit      - Exit program

Command: view
1. Buy toothbrush
2. Do homework

Command: complete
Number: 2

Command: add
Description: Pay bills

Command: view
1. Buy toothbrush
2. Pay bills

Command: history
1. Get bike fixed (DONE!)
2. Call your mom (DONE!)
3. Do homework (DONE!)

Command: exit
Bye!
```

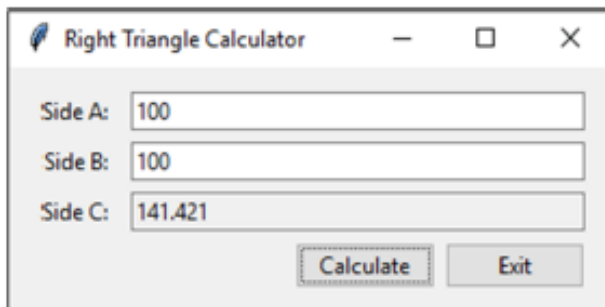
#### Specifications

- Your instructor should provide you with a database file (task\_list\_db.sqlite) that contains a Task table that stores the tasks.
- Use the three-tier architecture (presentation, business, database) for this program, and store the code for each tier in a separate file.
- The view command should only display tasks that have not been completed.
- The complete command should only mark a task as completed, not delete it from the database.
- The history command should allow you to view tasks that have been completed, but not deleted.
- The program should make sure the user enters a number that's a valid integer and also in the task list.

#### Question 4

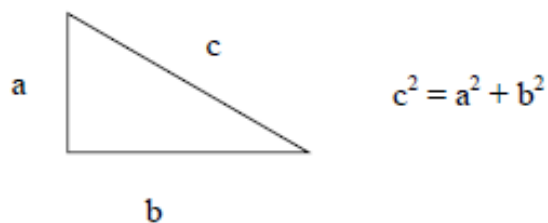
Create a GUI program that calculates the hypotenuse of a right triangle after the user enters the lengths of the two short sides and clicks the Calculate button.

#### GUI



#### Specifications

- Use the Pythagorean Theorem to calculate the length of the third side. The Pythagorean Theorem states that the square of the hypotenuse of a right-triangle is equal to the sum of the squares of the opposite sides:

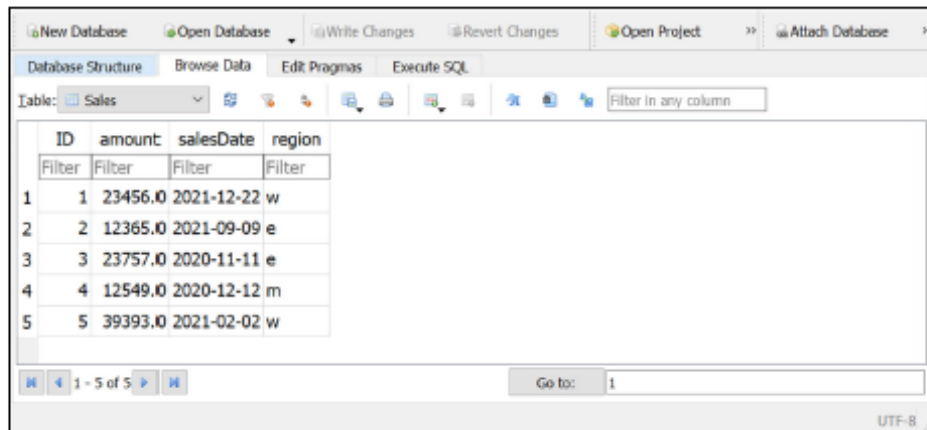


- As a result, you can calculate side C like this:  
`c = square_root(a2 + b2)`
- Side C should be rounded to a maximum of 3 decimal places.
- Make sure the user enters valid int values, and display any data validation messages in a message box.

## Question 5

Use a database to store sales, region, and imported file data for the program.

### DB Browser with the Sales table displayed



The screenshot shows the DB Browser for SQLite interface. The 'Database Structure' tab is active, and the 'Sales' table is selected. The table has four columns: ID, amount, salesDate, and region. The data is as follows:

	ID	amount	salesDate	region
1	1	23456.0	2021-12-22	w
2	2	12365.0	2021-09-09	e
3	3	23757.0	2020-11-11	e
4	4	12549.0	2020-12-12	m
5	5	39393.0	2021-02-02	w

The interface also shows a 'Filter in any column' box and a 'Go to:' field with the value '1'. The status bar at the bottom indicates 'UTF-8'.

### Specifications

- Use DB Browser for SQLite to create a new database for the program. To do that, click the New Database button, enter the database name including the extension, and then close the dialog box that's displayed.
- Use a table named Sales to store sales data, a table named Region to store region data, and a table named ImportedFiles to store imported file names. Here are the SQL statements that define the columns and their data types for these tables:

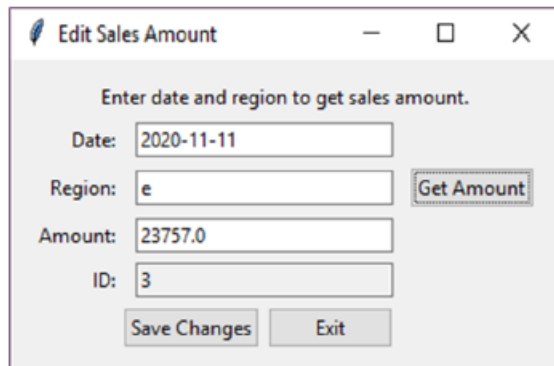
```
CREATE TABLE "Sales" (  
    "ID" INTEGER NOT NULL,  
    "amount" REAL NOT NULL,  
    "salesDate" TEXT NOT NULL,  
    "region" TEXT NOT NULL,  
    PRIMARY KEY ("ID" AUTOINCREMENT)  
);  
CREATE TABLE "Region" (  
    "code" TEXT NOT NULL,  
    "name" TEXT NOT NULL,  
    PRIMARY KEY ("code")  
);  
CREATE TABLE "ImportedFiles" (  
    "fileName" TEXT NOT NULL,  
    PRIMARY KEY ("fileName")  
);
```

- Use DB Browser's Execute SQL tab to run these SQL statements. To add the valid regions to the Regions table, use DB Browser's Browse Data tab.
- Modify the DailySales class to include an id attribute for the primary key and a fromDb() method. Modify the Regions class to start with an empty list and use an add() method to add regions to the list.
- Modify the db module so the functions work with the database rather than files. The exception is the import sales function, which should still import data from a CSV file.
- Note: SQLite doesn't have a date data type, so the sales date is stored as text in the YYYY-MM-DD format. You can use the SQLite date() function in the ORDER BY clause of the SELECT statement in the get\_all\_sales() function to sort by sales date.

## Question 6

Use a GUI to allow the user to change the sales amount for a specific date and region that's already in the database.

### GUI



Enter date and region to get sales amount.

Date: 2020-11-11

Region: e

Amount: 23757.0

ID: 3

Get Amount

Save Changes Exit

### Specifications

- The user must enter a date (in YYYY-MM-DD format) and a region code to retrieve and display the sales amount.
- The ID for the record is also retrieved and displayed in a read-only text box. This ID value is used to update the sales amount. That way, it won't matter if the user changes the date or region code after retrieving the sales amount.
- The program should notify the user by displaying a message box if there's no sales amount for the date and region entered.
- The program should notify the user by displaying a message box when the amount is updated successfully.