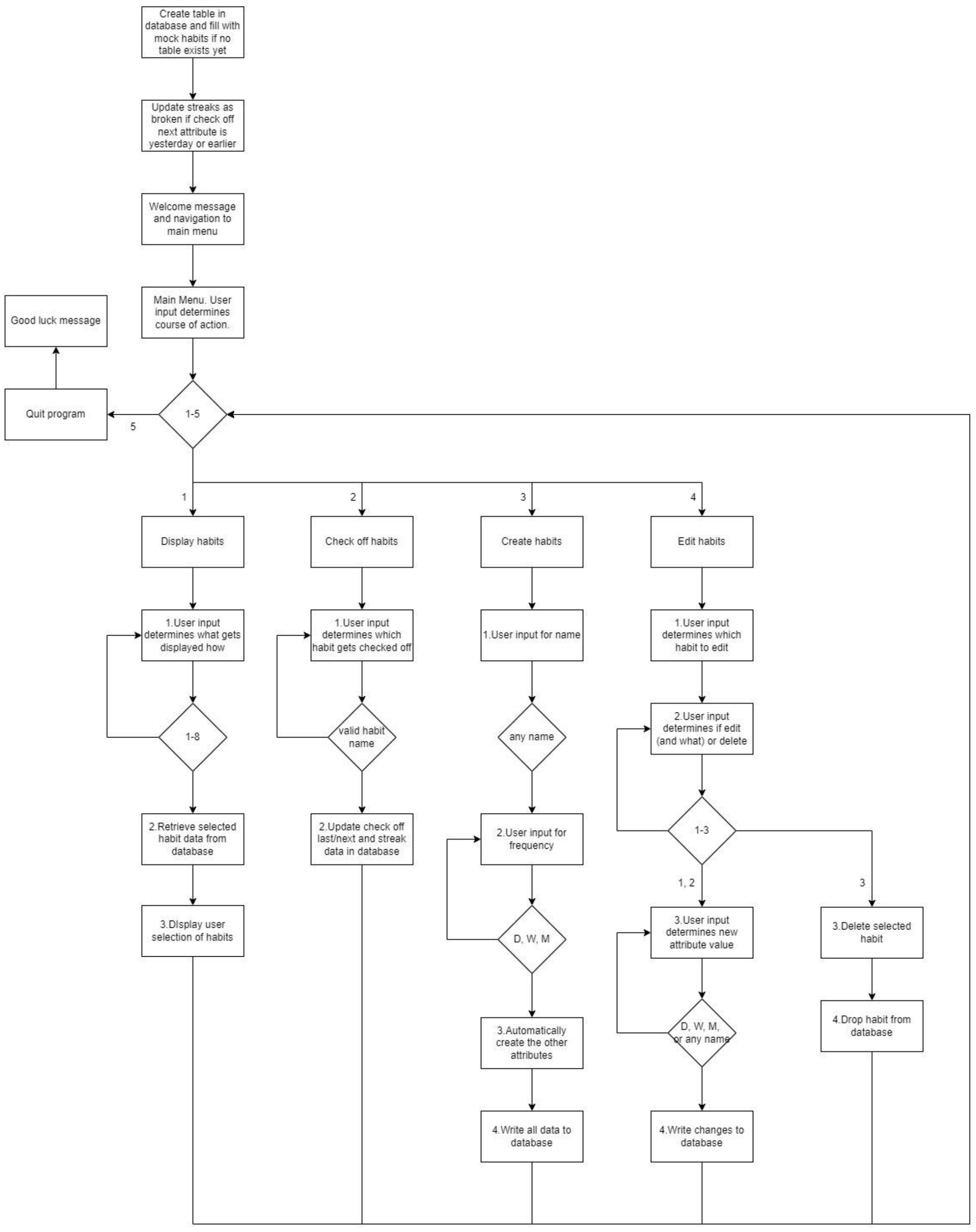
Habit tracking app – Conception Phase

Project scope and requirements:

The project consists of creating a habit tracker backend for my university course. The habit class should be implemented with the object-oriented paradigm in mind, while the processes will be implemented with the functional paradigm. Habit data needs to be stored outside of the application in order to persist in between app usage.

Requirements include the ability for the user to return a list of all currently tracked habits, to return a list of all habits with the same periodicity, to return the longest run streak of all tracked habits and to return the longest run streak for a given habit.

Flowchart of how the app is supposed to run:


Create table in database and fill with mock habits if no table exists yet
Update streaks as broken if check off next attribute is yesterday or earlier
Welcome message and navigation to main menu
Main Menu. User input determines course of action.
Good luck message
Quit program
5
1-5
1
2
3
4
Display habits
Check off habits
Create habits
Edit habits
1.User input determines what gets displayed how
1.User input determines which habit gets checked off
1.User input for name
1.User input determines which habit to edit
1-8
valid habit name
any name
2.User input determines if edit (and what) or delete
2.Retrieve selected habit data from database
2.Update check off last/next and streak data in database
2.User input for frequency
1-3
1, 2
3
3.DIsplay user selection of habits
D, W, M
3.User input determines new attribute value
3.Delete selected habit
D, W, M, or any name
4.Drop habit from database
3.Automatically create the other attributes
4.Write all data to database
4.Write changes to database

Tools:

I'll be using python with the class 'habit' and the attributes 'name', 'frequency', 'start_date', 'check_off_next', 'check_off_last', 'current_streak' and 'longest_streak'.

For libraries I'll be using sqlite3 for my database, datetime for the dates, dateutil.relativedelta for the date increase, re for user input validating and os for a clean terminal.

Things to implement:

On first startup. I need to create the database (if not existing) and add mock habits (if database empty). I'm using sqlite3 to use SQL queries in my python code to communicate with the database and will continuously write the data to the database whenever changes occur. On every startup. I need to import libraries, update streak values according to check_off_last and today's date.

1. Navigation/Interaction with the app

The client needs to be able to navigate through and interact with the app. For that I am planning on implementing a navigation function which will print text into the terminal and show the user his options regarding navigating the app. The user input is verified and used to determine what functions should be called to display their habits, check off habits, create new habits, edit or delete habits and eventually quit the app again. Once a function has been called and executed the program will call a circle_back function in order to see if the user wants to go back to the main menu to perform another task or if they want to quit the app. If the app is quit it will print a goodbye message.

2. Displaying habits

In order to display habits and save them in between uses of the app, I'm implementing a sqlite3 database that will store the habits and their attributes. To display them we send a request to the database file and print them out in the terminal. The user can choose to select attributes to order the data by, such as frequency of check off, longest streak, name, etc.

3. Check off habits

I will store an attribute last checked off with a date which gets updated whenever the user chooses to check off a habit. Checking off the habit also adds +1 to the current streak attribute and if the current streak attribute is higher than the longest streak attribute due to that change it will update the longest streak as well. Checking off a habit will also update the next check off attribute which will gain +1 or +7 days or +1 month, depending on the chosen frequency.

4. Creating habits

I'm implementing a function that lets the user create a new habit with a name (up to 50 characters) and a frequency (restricted to daily, weekly or monthly) of checking off the habit. The start date of the habit will automatically be implemented as today's date, along with the check off and streak attributes which start empty/at 0. Once the habit data has been collected inside the innit function I will write it to the database to save it long term.

5. Edit or delete habits

If the user chooses to edit or delete habits, they first need to select the habit by naming it and then select to either delete or edit. If they choose to delete the habit gets dropped from the database. If they choose edit, they are given the choice between editing the name or the frequency of check off and can then adjust the attribute they want to change.