Programming and Design Principles

Skills Demonstration

Contents

[Flowchart 1](#_Toc59046929)

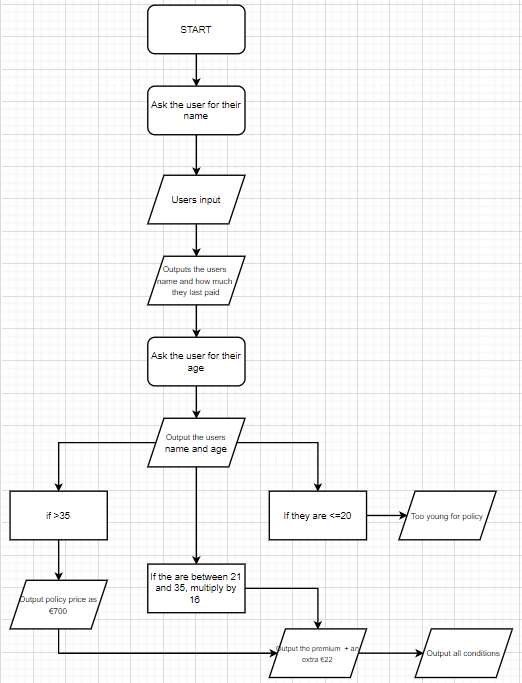
[Data Dictionary 2](#_Toc59046930)

[Test Plan 2](#_Toc59046931)

[The code 2](#_Toc59046932)

[Test Scenarios 4](#_Toc59046933)

# Flowchart

The first step of the development process was to get a basic idea of what I wanted the code to do, to do this I designed a flowchart. A flowchart is a diagram that showcases the process in a step by step format allowing for a clear understanding of what needs to happen and how it will happen.

The first step was to ask the user to enter their name and once they had it inputted, the code would output their name and how much they paid for their last policy.

Next was to ask for the users name and once inputted, the code would output a sentence using both the users name and age. If the user was over 35 it would output their policy price as 700 + an administrative charge of €22. If the user was between 21 and 35, their age would be multiplied by 16 and outputted with the extra €22 admin charge on top of that. If they were 20 or younger, the code would simply output a sentence explaining that they were too young to receive a policy.

Once the premiums had been outputted, a list of all 6 conditions would then be outputted. Terminating the process.

# Data Dictionary

The next step was to make a data dictionary, a data dictionary is a method used for analysing the data flows and the data stores. They collect, coordinate and confirm what a specific data term means to different people. Data dictionaries provide documentation, eliminate redundancy, they validate the data flow from the flow chart, they provide a starting point and they develop the logic for DFD processes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Data Item | Data Type | Data Format | Data Description | Example |
| name | String |  | Name of current user | Alex |
| age | Int | xnn | Age of Current User | 17 |
| policy | Int | Xnnnn | Policy Price | 602 |
| charge | int | xn | Admin Charge | 22 |
| Duration | Boolean | True or False | If under a certain, false | T |

# Test Plan

The next step was to make a test plan. This is a way in which I can have an idea of what my code should be outputting with specific inputs, this is extremely useful as it shows me what I should expect when writing my code.

Below is my test plan.

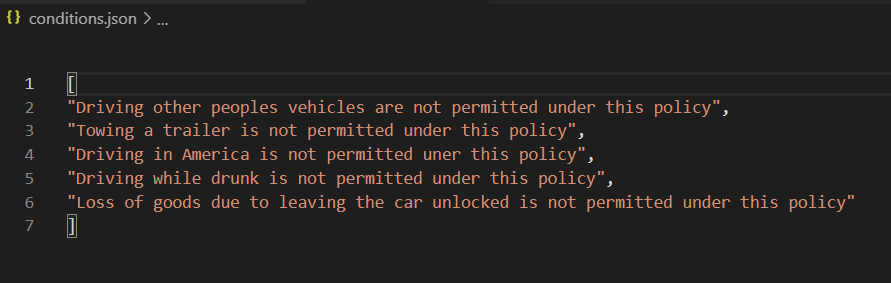
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Plan** | Name input | Age input | Output |
| Example 1 | Alex | 17 | Too young for a policy + the amount of years of until they can get one  (Hello Alex, as you’re only 17 you do not meet our requirements for a policy. Please come back in 4 years) |
| Example 2 | John | 24 | Should output €406 (24\*16+22) and the conditions  (Hello John, as you’re 24 your premium will be 384 plus an admin charge of 22) |
| Example 3 | Sophie | 40 | Should output €722 (700+22) and the conditions  (Hello Sophie, as you’re 40, your premium is a standard charge of 700 plus a 22 admin charge) |

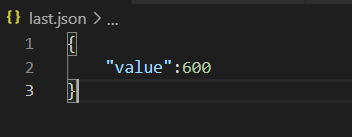
# The code

Here is my JavaScript code, I made sure to follow the briefs requirements and I followed my flowchart to help me keep track of what I was doing.

Here is my main file, Assignment.js. This the part that does everything important whereas the other files only store information.



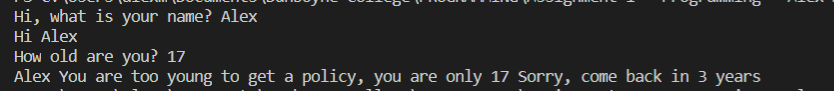
Here is my first json file, conditions.json. This is the file that stores all the policy conditions. 

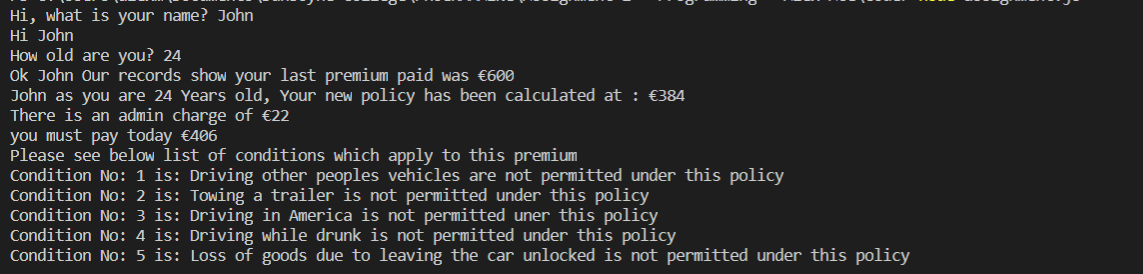
Here is my final file, last.json. This is the file that is used to store the previous premimum paid. 

# Test Scenarios

Now that I have my code ready and running, I decided to use my test plan to test the code to ensure it is working accordingly.

Here is example one, this test is testing to see if my age requirements are working properly. It was successful.



Here is example two, this test is testing to make sure my code works for people aged between 21 and 35. It was successful. 

Here is my final example. This test was testing to make sure my code worked for people over 35. 