Daft.ie scraper with algorithm to find a dream home.

Metaheuristic Optimisation & OOP

Mateusz Jakusz, CCT Dublin, sba21173 https://github.com/bloobsky/AffordableHousingFinder

2021

Table of Contents

[What is this application about? 2](#_Toc74501582)

[Design Decisions 2](#_Toc74501583)

[Structure of the code 2](#_Toc74501584)

[Algorithms 3](#_Toc74501585)

[Big-O-Notation (Huang, 2021) 4](#_Toc74501586)

[Screencast 5](#_Toc74501587)

[References 5](#_Toc74501588)

# What is this application about?

The application presented allows user to search through Daft.ie to find his perfect accommodation. It uses open-source module called daftlistings (Bloomer, 2021). It enables user to give the salary – and algorithm will find home crafted for his/her needs. User can also put information about location – where he/she wants to live. Currently available are 4 options: Cork City, Dublin, City, Galway City or Limerick. Options like rent or buy are also included (both using different algorithms). At the end he can choose between apartment or house. The results are presented in user-friendly GUI and also stores values in .csv (Comma Separated Values) file.

# Design Decisions

When application was in development process the design was crucial. The main target was – intuitive and simple so any user despite of age can use it without any complications.

The program is divided into 3 sections:

1. User enters the information that would be used towards search. His name, location that he’s willing to search, salary monthly, buy or rent and at the end property type(house or apartment).
2. Second window is only for double confirmation before program start to search for it.
3. Last window presents all the information corrected and adjusted for better UX (User Experience)

# Structure of the code

The code is divided into 3 files

1. Requirements.txt – has all the packages that are required for the program to work. To simply install it just type: **pip install -r requirements.txt.**
2. local\_settings.py – simple .py file that stores all CONSTANTS used for the application itself. It contains info like font type or size.
3. Main.py – main application file. It contains

* **Main Loop of the code that is divided into 3 steps (First Window for User Input, Second Window for informative purpose and eventually double confirmation, Last Window is presenting results to the user.**
* Class Person is an Object that takes 5 values
* Name = User Name
* Where – Desired Location
* Price – monthly salary
* Action – Buy or Rent
* Kind – House of Apartment

Class that contains two more function (default \_\_init\_\_ and \_\_str\_\_ are not counted in)

Function change\_price() – not used for the project it was written towards further development of the app.

Function Calc\_Price that captures all the logic and algorithms used towards finding dream accommodation.

Last one function that is important is to check that connection to daft.ie exists.

# Algorithms

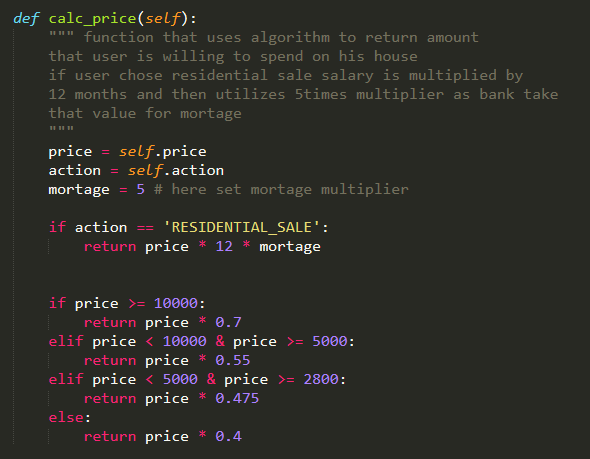
Application has many algorithms but for purpose of the project I selected 3 of them.

1. Check Connection Function



Simple function that takes no input it has 2 variables (one checks **status\_code** from website, the other return value ‘OK or ‘NOT OK’ thus the name on\_or\_404. It contains one if loop that returns that value depends of the status\_code acquired from website.

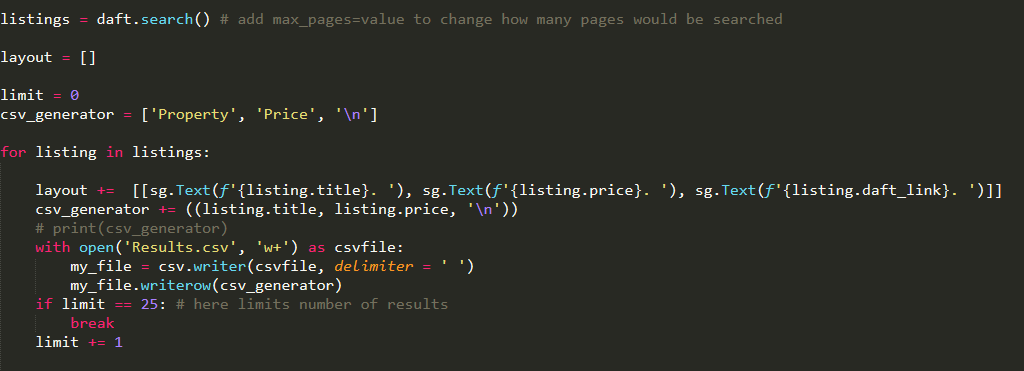
1. Calculate Price (function that calculates formula based on monthly salary given)



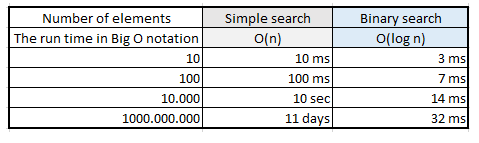
It take one arguments based on Class itself. The variables (price and action) -> where price is monthly salary and action means buy or rent.

Firstly, algorithm check is user is looking to buy or to rent. If user has choses buy algorithms returns monthly salary, multiplied by 12 months and mortgage parameter(in this case variable is set to calculate 5times of yearly salary). That can be easily change by modifying the mortgage variable.

1. Final algorithms that uses for loop to iterate through all properties founded. In this case I used limit (variable) to limit the results to 26 (if you wish to change simply modify if conditional at the end of the function .It also includes writing to file that will write any property found into the .csv file.



# Big-O-Notation (Huang, 2021)



**1st algorithms Big-O-Notation** -  if-then-else statements

if (condition) {

sequence of statements 1

}

else {

sequence of statements 2

}

Here, either sequence 1 will execute, or sequence 2 will execute. Therefore, the worst-case time is the slowest of the two possibilities: max(time(sequence 1), time(sequence 2)). For example, if sequence 1 is O(N) and sequence 2 is O(1) the worst-case time for the whole if-then-else statement would be O(N).

**2nd algorithm Big-O-Notation** is very similar to first one but have to conditionals but they are not nested so. I am also using mathematical equations – they are considered to be constant time O(1).

**3rd algorithm Big-O-Notation**  its much complexed. It has for loop that executes N times, so the sequence of statements also executes N times. Since we assume the statements are O(1), the total time for the for loop is N \* O(1) (if conditional), which is O(N) overall. In our case N is limit.

# Screencast

Screencast for proposed project is available in folder ‘screencast’ on GitHub (link in the title.

# References

Bloomer, A., 2021. *AnthonyBloomer/daftlistings*. [online] GitHub. Available at: <https://github.com/AnthonyBloomer/daftlistings> [Accessed 13 June 2021].

Huang, S., 2021. *What is Big O Notation Explained: Space and Time Complexity*. [online] freeCodeCamp.org. Available at: <https://www.freecodecamp.org/news/big-o-notation-why-it-matters-and-why-it-doesnt-1674cfa8a23c/> [Accessed 13 June 2021].