Evaluation

To begin, the program meets all the requirements set out in the brief and there isn't anything I would change with the implementation of those requirements but there are some things I would change regarding the flow of the program, error handling and some nice to have things. The program is written in Python 3.10 and uses a text menu-based system. It follows most programming standards and is simple to use.

Program Flow

The program is menu based and the employees are loaded in from a CSV file called "employees.csv", this was done to allow the user of the program to easily add new employees and for I, the programmer, to easily handle where the employees would come from rather than inputting them at the start every time the program starts. This flow goes like this:

- The program starts and begins to set the required state.
- The program then loads the "employees.csv" file from the CWD.
- The employees are then cached in a list called "employees", this means that employees do not need to be loaded in every time they are required by the program, which can be a slow process depending on disk speed and how many employees are in the file.
- Once loaded the employee selector menu is printed to the screen.

Once the user selects an employee that employee will be set as the current employee object and the main menu is printed to the screen. This flow works rather well and I would change the way it works but one thing I would have liked to add is a way to load employees when the file changes at runtime because right now the user has to restart the program if a new employee is added to the file due to the way the cache works.

The main menu contains different options to interact with the current employees or all of them. For example, the "Properties Sold" option allows the user to view the current properties sold by the current employee and add more sales said employee but the "Calculate Commission" menu will rank the employees using bubble sort and calculate each employees commission and print the total. The main works well and there isn't much I would change to it.

The different menus are designed to be simple but one thing I would change is clearing the screen of pasted menus, the code is in the program but on testing, it would not work to save time I cut it from the program.

Overall, I think the program flows rather well and there isn't much I would change regarding it.

Error Handling

I wished I put more error handling and tested some more error cases in this program but due to time, I tried to catch all of the obvious ones. One of the main areas that could be improved is the properties sold and the employee selector which currently will not catch very obvious errors, the first being validating the data inputted by the user, for example, the user can input -1 for properties sold and the other being if the employee file does not exist the program will simply crash with a Python IOError. There are probably other errors lying around but time was tight and you can't catch everything.

Niceties

There are some things that I would have changed in this program and something I would add but time again to over. Some things I found while testing are the total sales just being printed out without any text explaining what it is, I would have changed and made it a little prettier. Another little thing is the employees are not numbered when ranked, this doesn't break anything but it would have been nice to have. The last thing regarding the output would be printing a message to the screen saying "Exiting..." or something similar when the program is exiting.

Regarding the code, there are some things I would improve or change. The first is documentation and some comments, currently the program doesn't have any documentation explaining what things do and how they do it. Furthermore, if time wasn't so tight I would have used OOP and less global state but this takes time to debug and get right, so I went with a more functional approach and used a class for the employee data structure.

WWW

Here is a list of things I feel went well in the making of this program:

- Meeting all of the requirements set in the brief.
- Managing time and cutting back on ideas to complete the program and supporting tasks.
- Creating an easy to use program that is simple.
- Implementing searching and sorting algorithms from scratch.
- Following programming standards and best practices.

Conclusion

To conclude, there are some things I would change to this program but overall I think it went well and was a success.