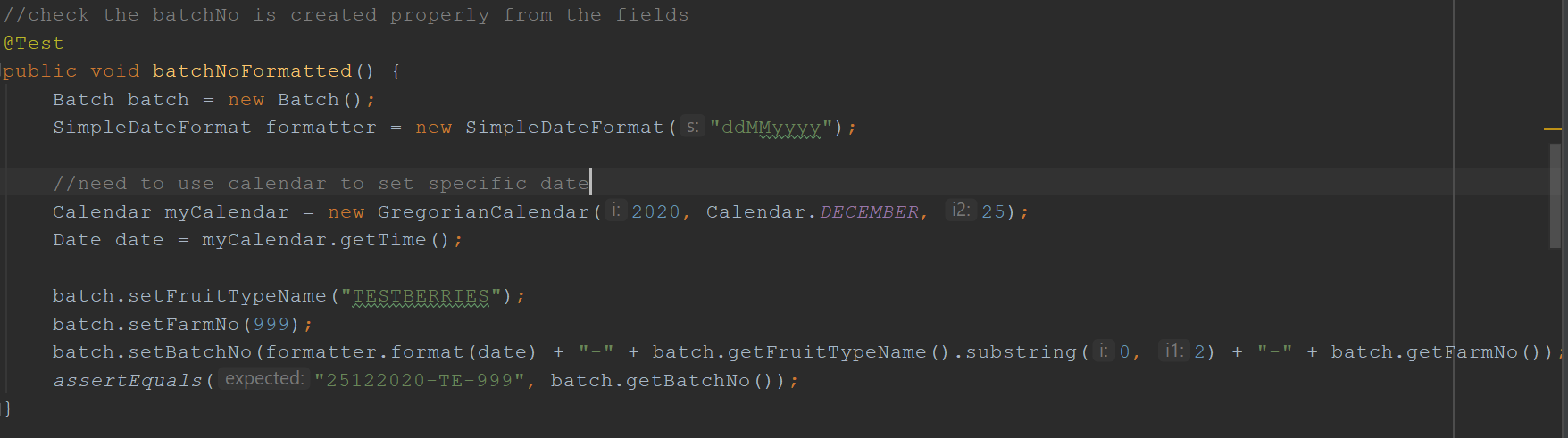
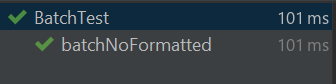
# Step 4

## Junit

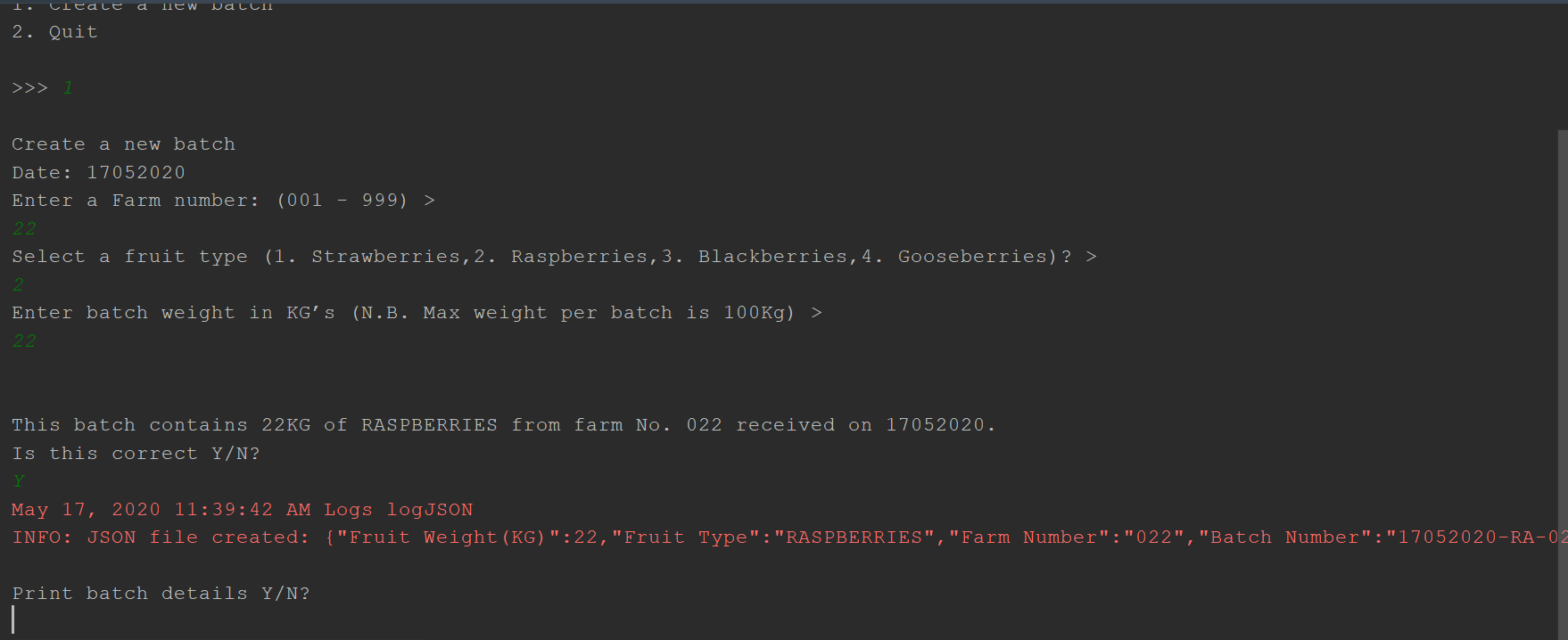
I tried to implement Junit tests, however I ran into some difficulty testing for Junit with a command line interface. I researched a variety of solutions which will automatically input set testing parameters into the console when a prompt is made. This has so far been unsuccessful and I have spent far too much time on it to the point it is no longer making sense to continue.





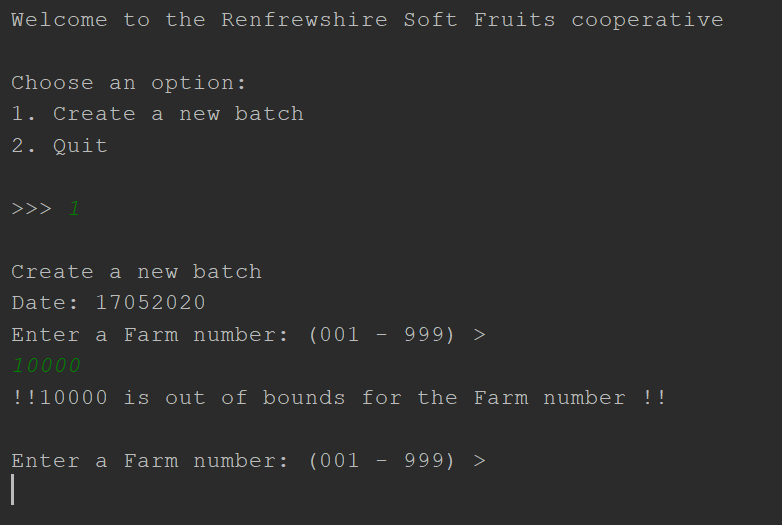
## Testing

Below is an example of 0 issues. This is running without error.



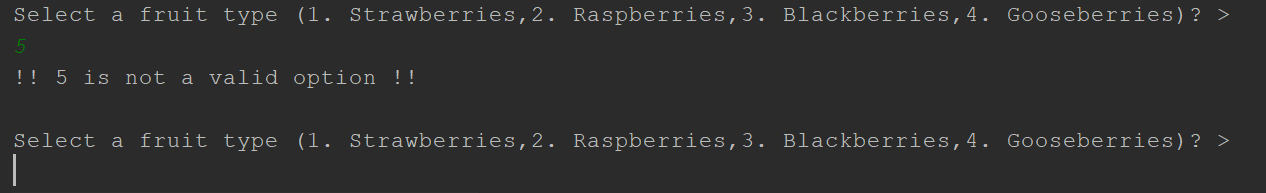
Below is an example of incompatible information being entered regarding the Farm Number.

As you can see the user is prompted to input correct information.



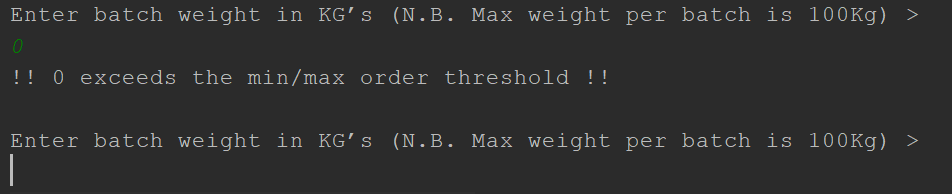
Below is an example of incompatible information being entered regarding the Fruit type.

As you can see the user is prompted to input correct information.



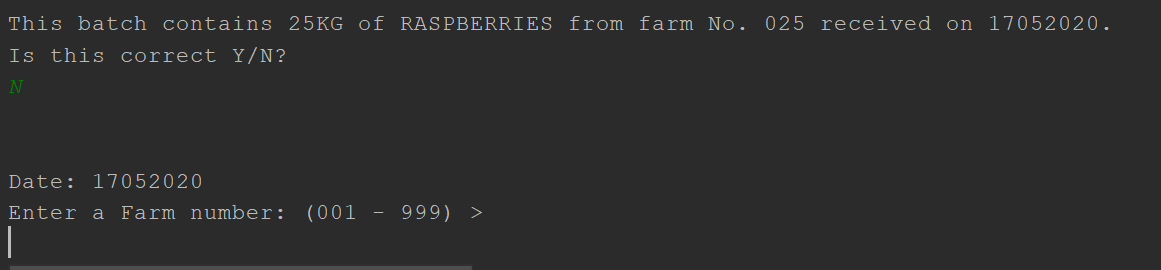
Below is an example of incompatible information being entered regarding the Fruit weight.

As you can see the user is prompted to input correct information.



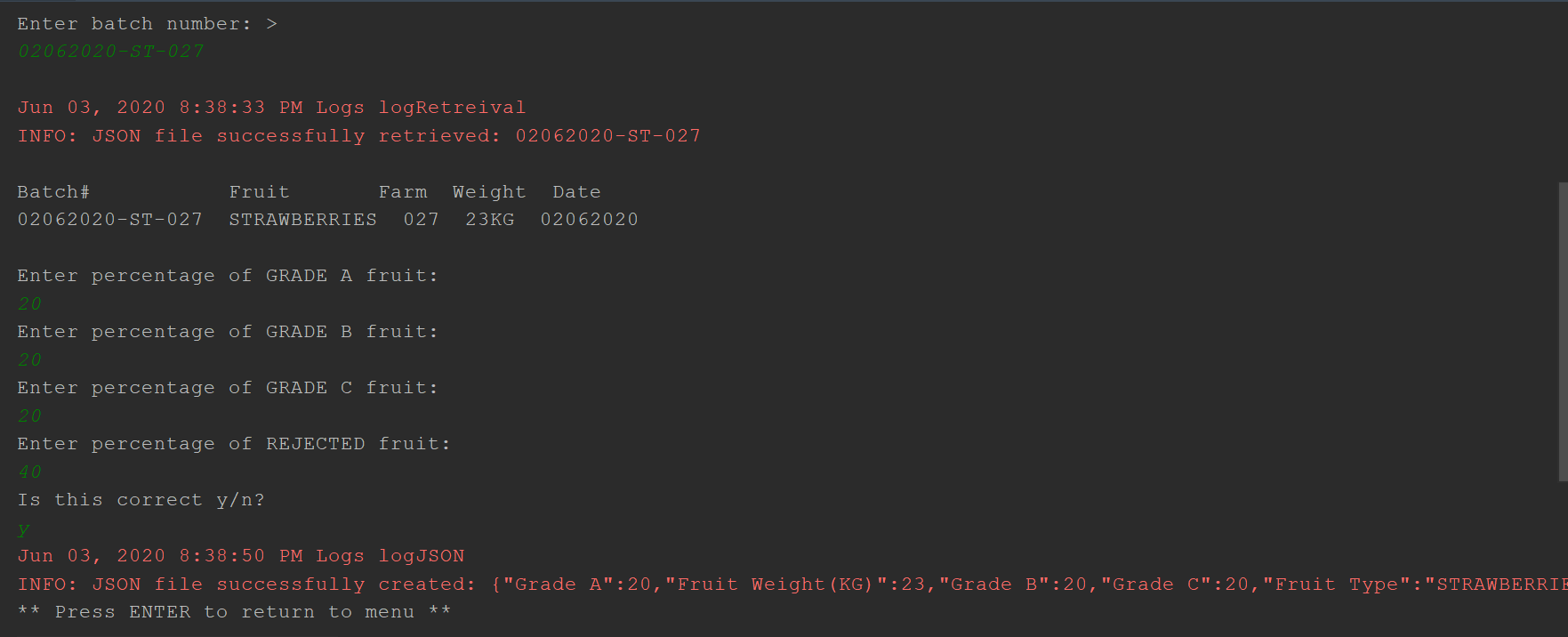
Below is an example of the user not being satisfied with their order information.

As you can see the user is prompted to input correct information.



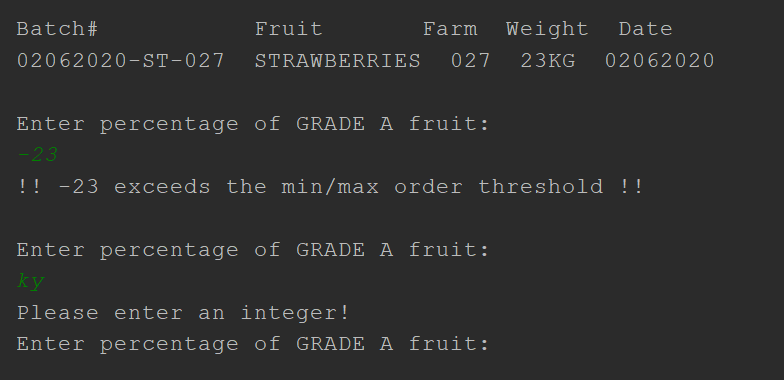
Below is an example of entering grading info and it being correct

As you can see the user is told that info has been stored successfully



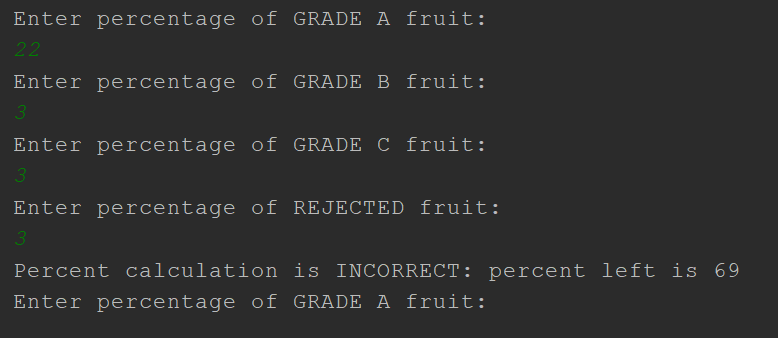
Below is an example of entering grading info and it being incorrect

As you can see the user is told that info has to be re-input

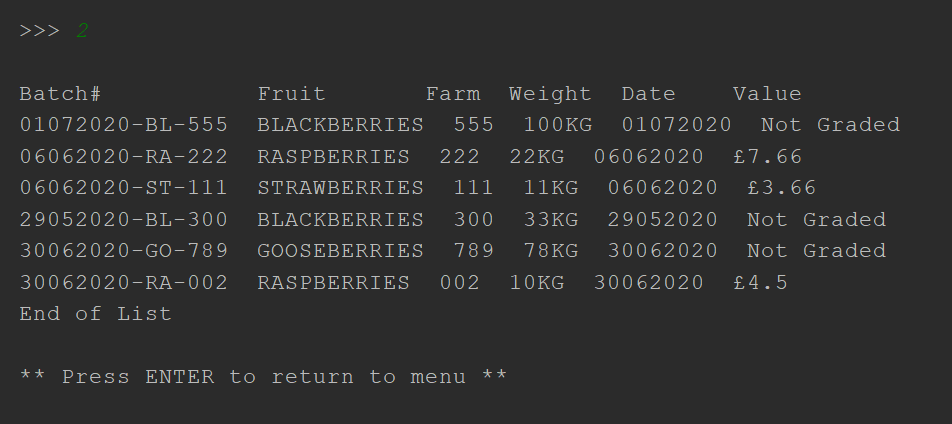


Below is an example entering grading info and it being incorrect due to wrong percentage

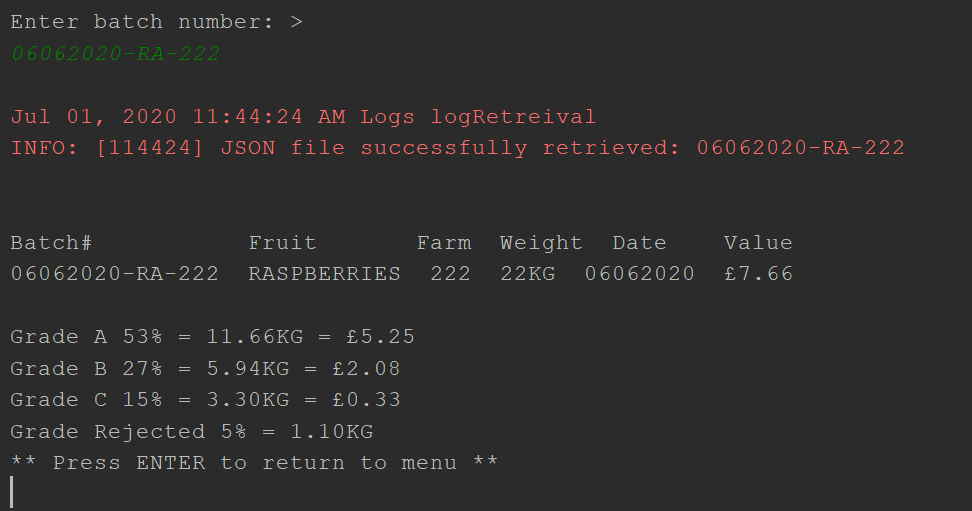
As you can see the user is told that info has to be re-input



2. List all batches

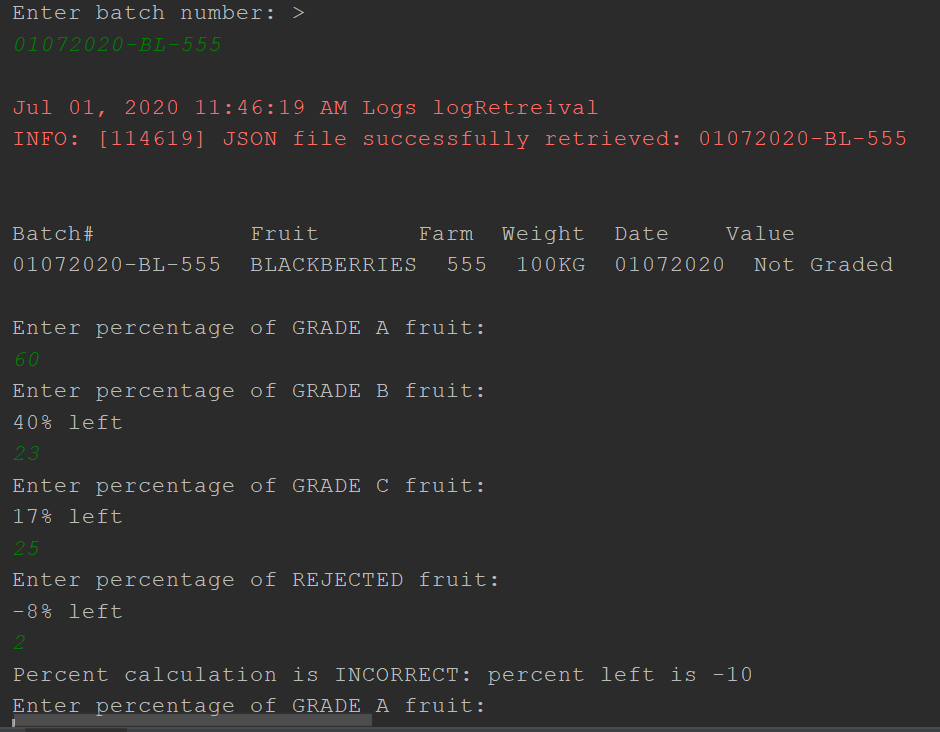


3. View details of a batch

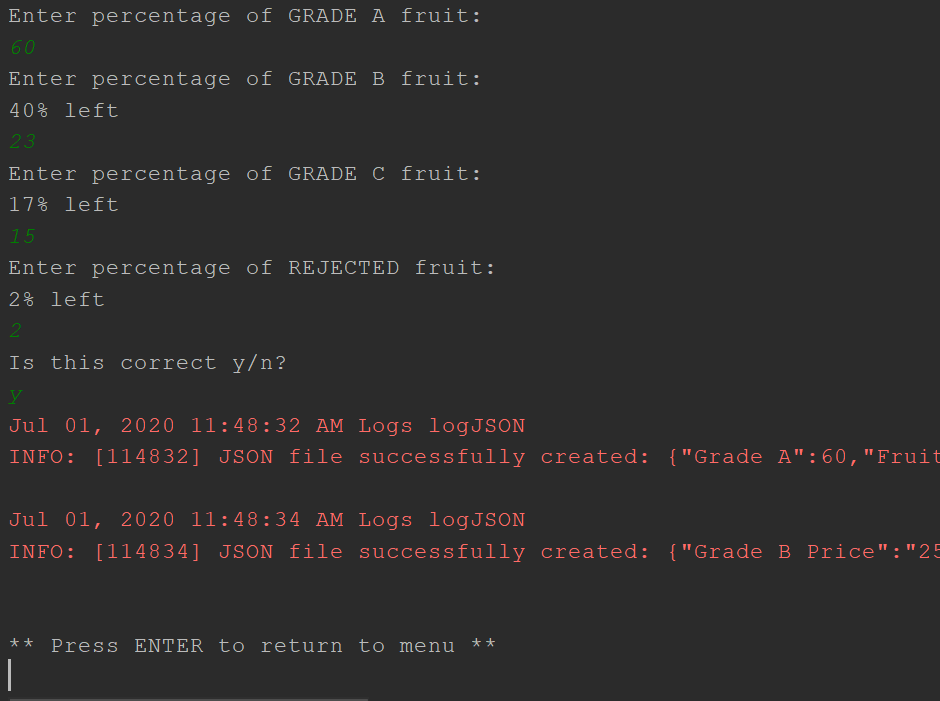


4. Sort/Grade a batch

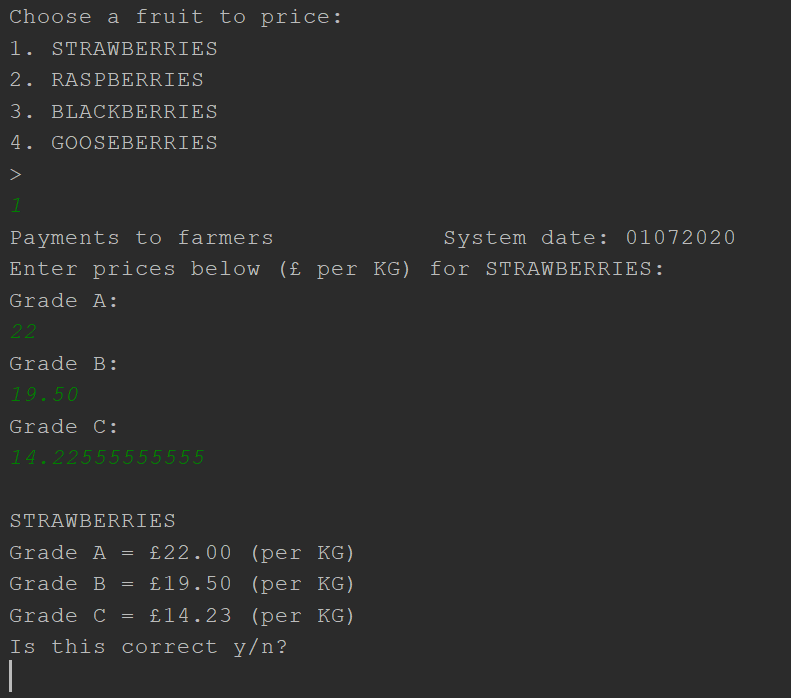
*Invalid input*

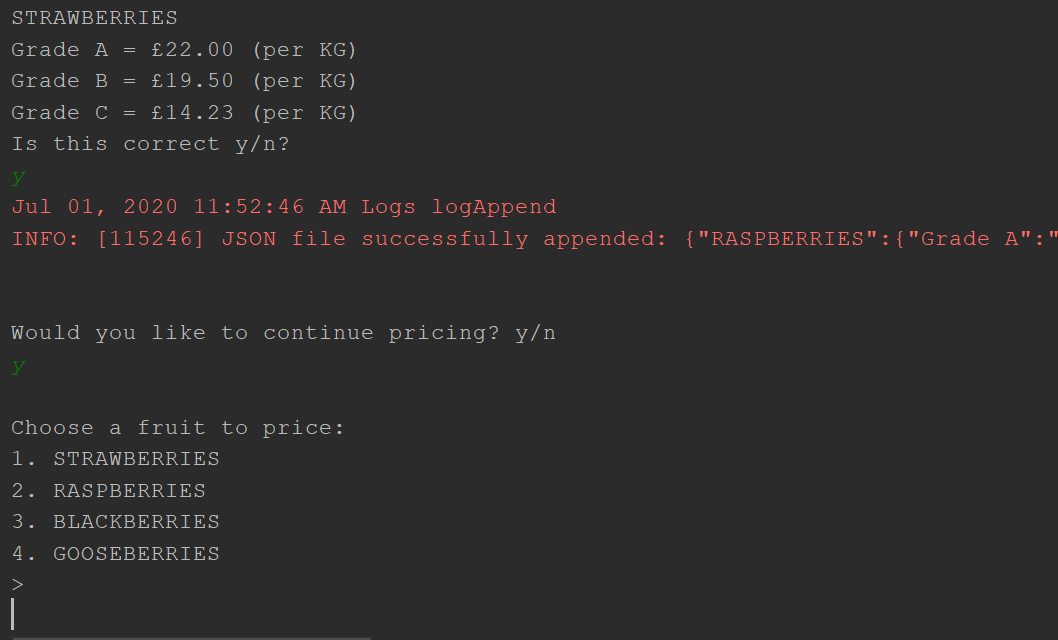


*Valid Input*

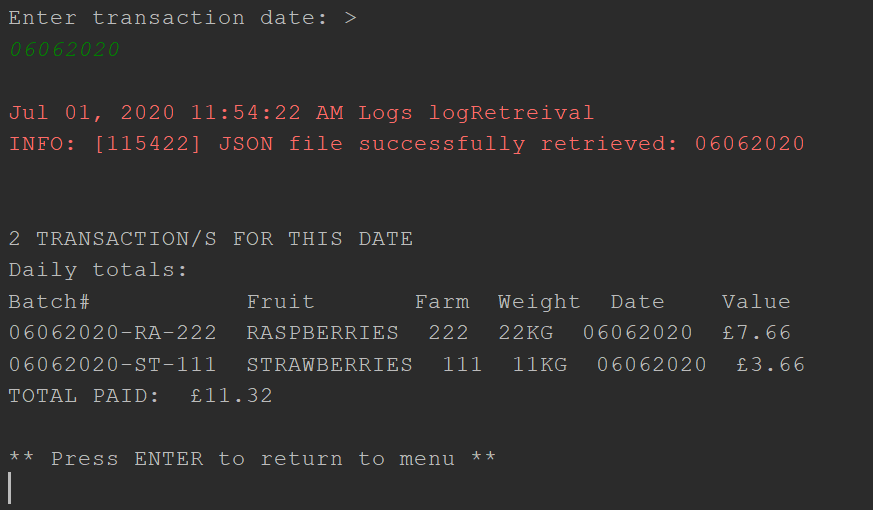


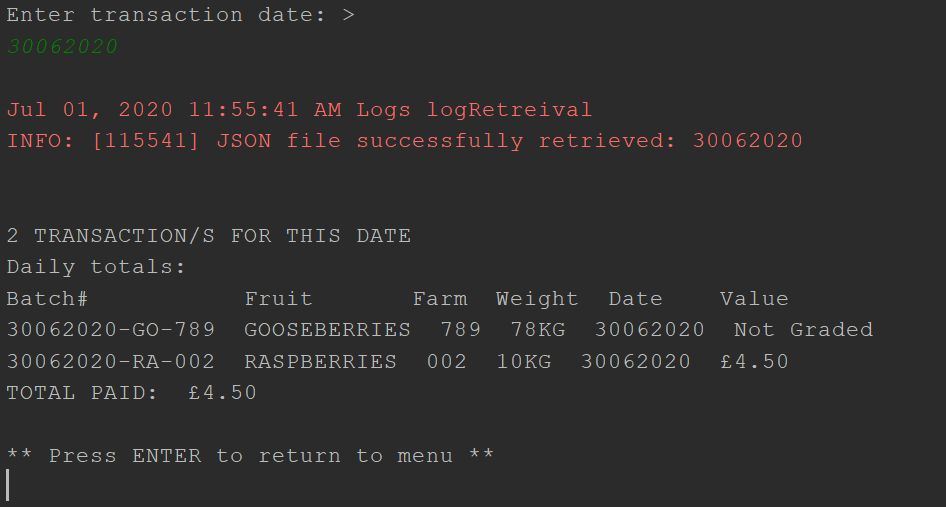
5. Payments

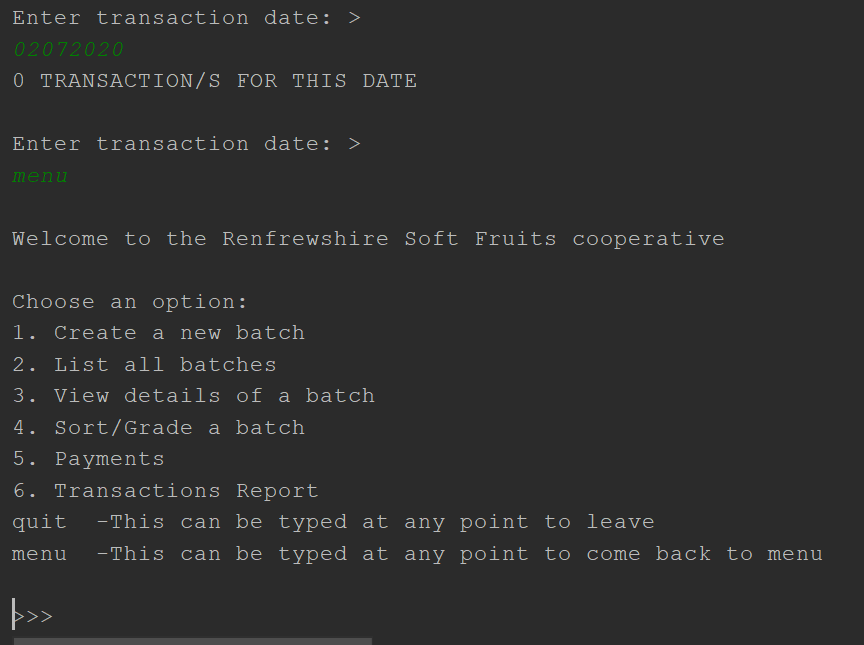




6. Transactions Report





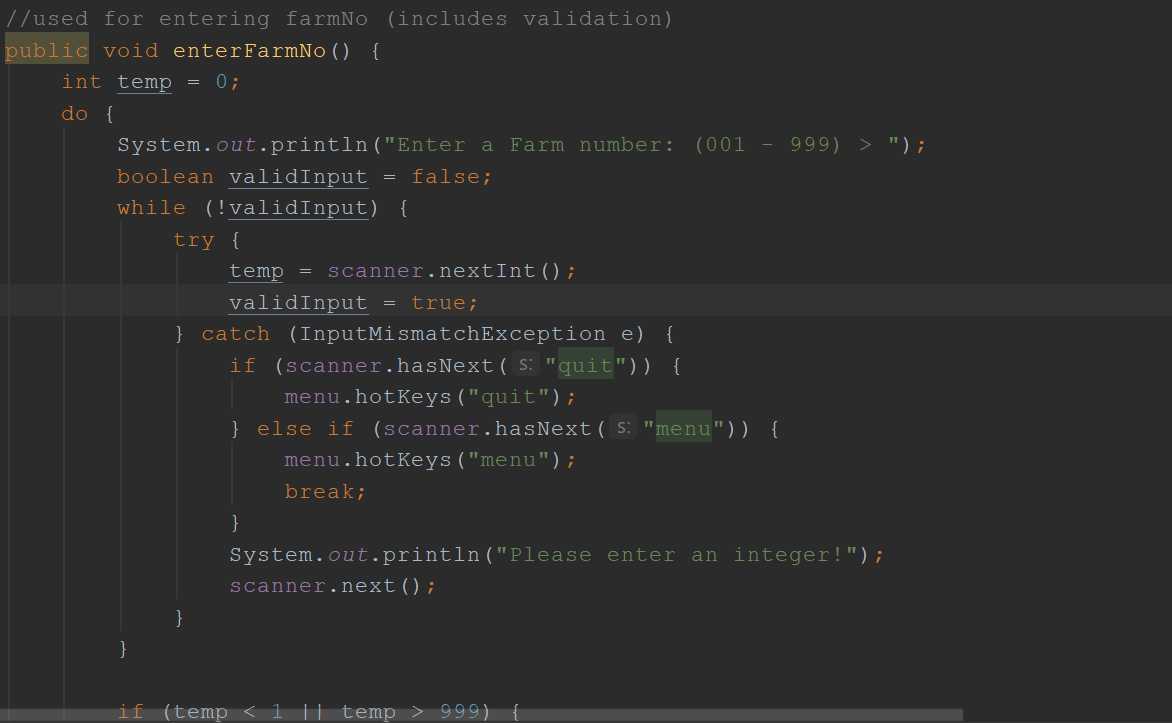


# Additional Functionality

In my first step I had added the functionality to loop around when inputs were invalid. This loop prompted the user to re-input their values. I managed this using a do-while.

In my second step I realised that this posed a problem. If you realised you had the wrong details once you arrived at this console input then you couldn’t back out as an app user. So, I implemented methods which allowed the user to always be able to jump back to menu and quit. These can be put in during most console prompts, it is not appropriate for y/n choices I thought.

HERE is enterFarmNo() for example



LOGS

I created a logging tool so that I could track issues and keep documentation of file changes. I have never implemented logs before so was interesting to research and learn. There is a folder included in my project where you can see logs going back to step2 as this is where I implemented it. It shows how my storing of details has progressed.

