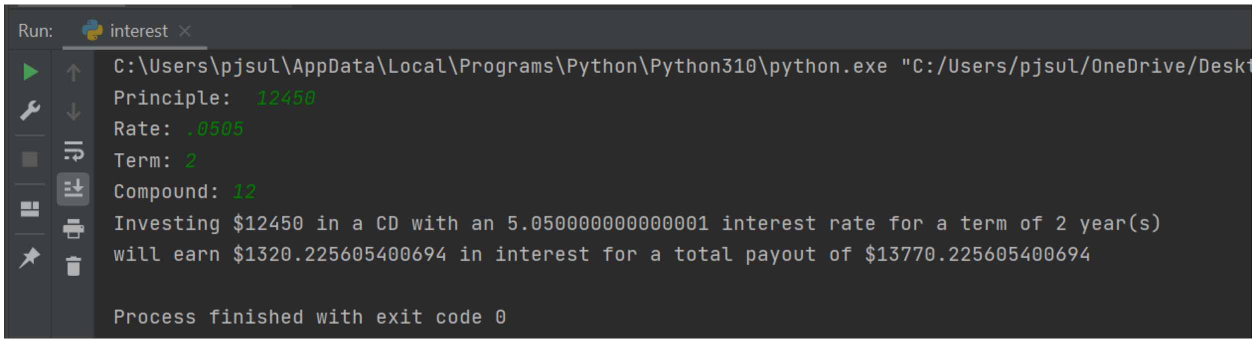
Interest.py Results

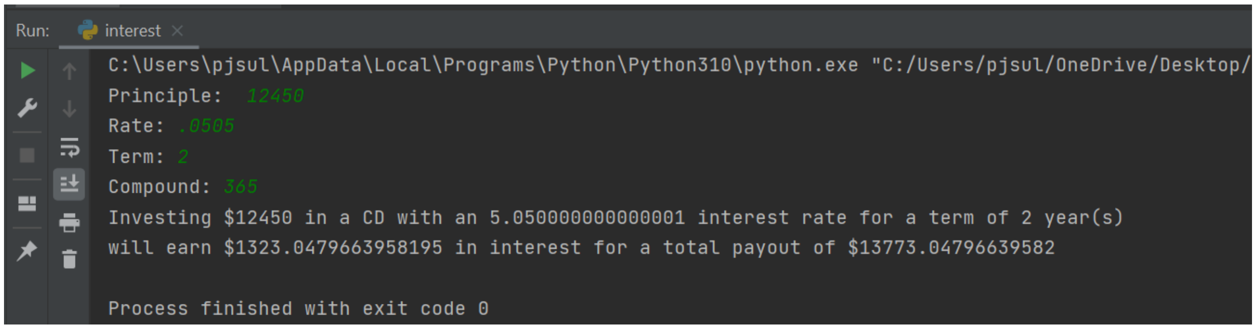
Scenario 1:



```
C:\Users\pjsul\AppData\Local\Programs\Python\Python310\python.exe "C:/Users/pjsul/OneDrive/Deskt
Principle:  12450
Rate: .0505
Term: 2
Compound: 12
Investing $12450 in a CD with an 5.050000000000001 interest rate for a term of 2 year(s)
will earn $1320.225605400694 in interest for a total payout of $13770.225605400694

Process finished with exit code 0
```

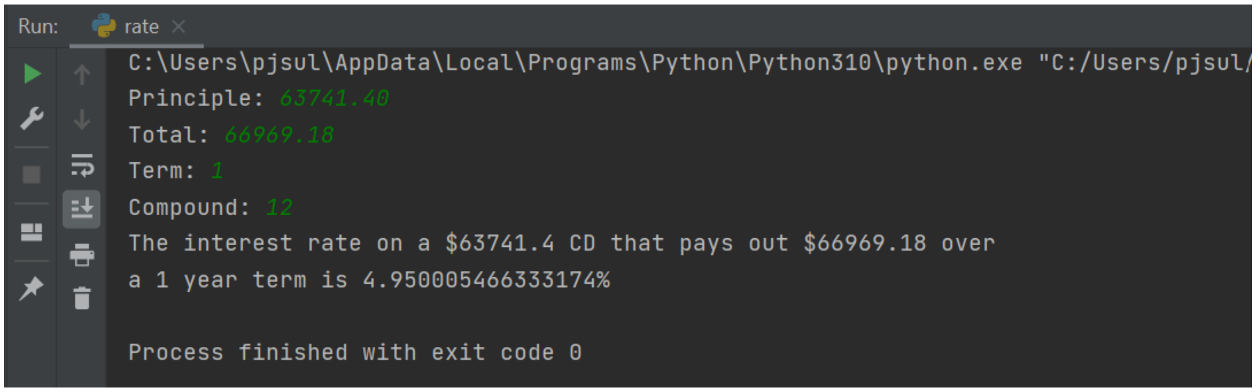
Scenario 2:



```
C:\Users\pjsul\AppData\Local\Programs\Python\Python310\python.exe "C:/Users/pjsul/OneDrive/Desktop/
Principle:  12450
Rate: .0505
Term: 2
Compound: 365
Investing $12450 in a CD with an 5.050000000000001 interest rate for a term of 2 year(s)
will earn $1323.0479663958195 in interest for a total payout of $13773.04796639582

Process finished with exit code 0
```

Rate.py Results



```
C:\Users\pjsul\AppData\Local\Programs\Python\Python310\python.exe "C:/Users/pjsul/
Principle: 63741.40
Total: 66969.18
Term: 1
Compound: 12
The interest rate on a $63741.4 CD that pays out $66969.18 over
a 1 year term is 4.950005466333174%

Process finished with exit code 0
```

Questions:

1. The error in accuracy is coming from the internal representation that python uses for floats. Floats use a 64 bit double precision representation based on the standard IEEE 754. Float type numbers are represented in a binary format as bae-2 fractions. This allows for quick computation but can cause rounding errors when computing between floating numbers. We had dived the paid out by the principle which is causing the rounding error displayed for the rate.
2. The issue of dropping trailing fractional zeros is also due to the IEEE 754 standard of representing float values as a 64 bit double precision value. The reason that 63741.40 is being displayed as

63741.4 is due to the number being converted to a float. Floats are not decisioned to hold precision.