Cristian Panaro
2 November 2023

Assignment 2.3

There were three main areas of concern in my reference monitor. The first was to account for negative offset. I needed to add a part to my program that raised a RepyArgumentError when confronted with a negative offset during a write operation, which is invalid. I did this like so in my write function:

```
if offset<0:
	raise RepyArgumentError()
```

Secondly, I needed to account for when people tried to write outside the EOF. To do so, I needed to keep track of where the EOF file was. I did this through initializing the size of the eof when opened to the size of the current file:

```
initial_eof = len(self.LPfile.readat(None, 0))
self.eof = initial_eof
self.prev_eof = initial_eof
```

This was done in the __init__ function

After this, I added this to my write function:

```
if offset>self.eof:
	raise SeekPastEndOfFileError()
```

This ensured that if an offset out of the EOF was attempted, this would result in the proper SeekPastEndOfFileError.

Another aspect of this was adding the following to the undo function. This ensured that if an undo was completed, that the current EOF reverted to the old EOF marker:

```
self.eof = self.prev_eof
```

Finally, I had to make sure that a FileClosed Error was thrown when operations were attempted after the file had been closed:

```
if self.closed:
	raise FileClosedError("File is closed")
```

I added the following to the __init__ to keep track of whether the file was closed or not. When the file was first opened, it starts off as “False”.

```
self.closed = False
```