There were several problems I detected and fixed:

1. I included the file size as a variable.
  `self.filesize` stores the size of the file, and it is initialized with the length of the file when the `LPFile` is created.
This is to track the offset problem, where EOF problems (SeekPastEndOfFileError) should be handled.

```
self.filesize = len(self.LPfile.readat(None,0)) # add file size
self.pending_data = None
self.pending_offset = None
self.pending_file_size = self.filesize #pending file size equals self.filesize
```

2. Writeat
  In the same context, parts like `if self.file_closed or offset < 0 or offset > self.pending_file_size: ` is added to ensure that the needed specifications are met.
Instead of raising an error I decided to include more cases that regulates the writeat method, and doesn't update if it doesn't match the case.

```
if offset > self.pending_file_size or self.file_closed or offset < 0:
       #safety check!
       self.LPfile.writeat(data,offset)
if self.pending_data is not None:
       self.LPfile.writeat(self.pending_data, self.pending_offset)
       self.filesize = self.pending_file_size
```

Also the file size and `pending_file_size` are updated together to reflect the maximum offset written to, ensuring that the file size is correctly tracked.

```
self.pending_file_size = max(self.pending_file_size,self.pending_offset +
len(self.pending_data))
```

These updates in writeat helped me to update the testfiles correctly as well as handle the EOF problems correctly.

3. Also the undo method needed some change,
 Now the `undo` method also resets the `pending_file_size` to match the actual file size.
This also handles the problem of fixing the offset and the filesize to correctly handle the created file.
However, controlling the attack cases that don't erase the existing <filename.txt> wasn't able to be done since I couldn't find a way to delete it only at the beginning of each attack file. I would further handle this via the way I run it on the bash, but couldn't find for the security layer.