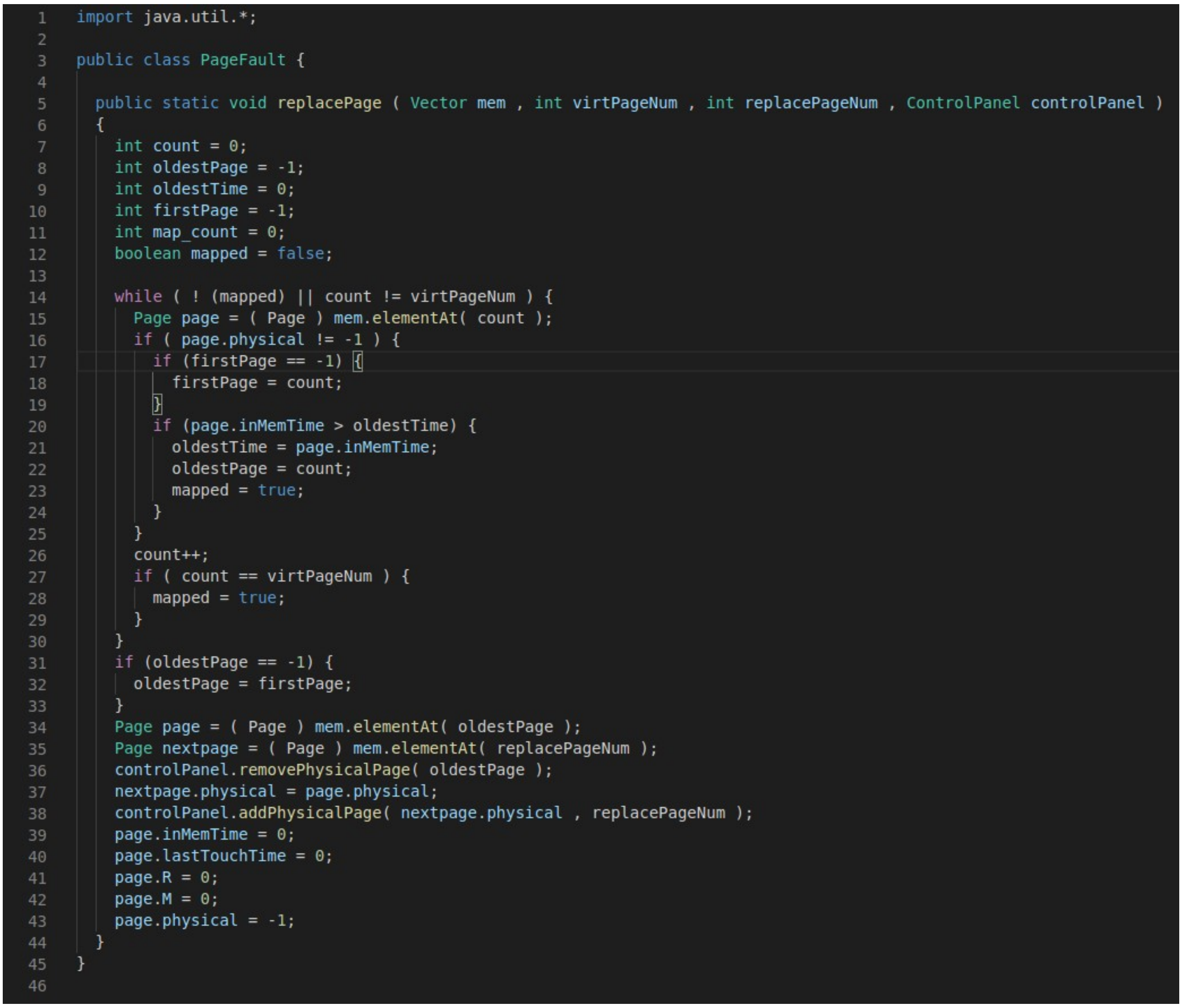
EOPSY TASK 4

Once again to begin the project we have to download the file, extract it and use the command **run setup** in a linux terminal. When it’s done we can find the file **PageFault.java** in the folder **work**. The file **PageFault.java** contain the following code as you can see on the picture.
All the comments line have been deleted.



```
 1  import java.util.*;
 2
 3  public class PageFault {
 4
 5    public static void replacePage ( Vector mem , int virtPageNum , int replacePageNum , ControlPanel controlPanel )
 6    {
 7      int count = 0;
 8      int oldestPage = -1;
 9      int oldestTime = 0;
10      int firstPage = -1;
11      int map_count = 0;
12      boolean mapped = false;
13
14      while ( ! (mapped) || count != virtPageNum ) {
15        Page page = ( Page ) mem.elementAt( count );
16        if ( page.physical != -1 ) {
17          if (firstPage == -1) {
18            firstPage = count;
19          }
20          if (page.inMemTime > oldestTime) {
21            oldestTime = page.inMemTime;
22            oldestPage = count;
23            mapped = true;
24          }
25        }
26        count++;
27        if ( count == virtPageNum ) {
28          mapped = true;
29        }
30      }
31      if (oldestPage == -1) {
32        oldestPage = firstPage;
33      }
34      Page page = ( Page ) mem.elementAt( oldestPage );
35      Page nextpage = ( Page ) mem.elementAt( replacePageNum );
36      controlPanel.removePhysicalPage( oldestPage );
37      nextpage.physical = page.physical;
38      controlPanel.addPhysicalPage( nextpage.physical , replacePageNum );
39      page.inMemTime = 0;
40      page.lastTouchTime = 0;
41      page.R = 0;
42      page.M = 0;
43      page.physical = -1;
44    }
45  }
46
```

We can see that the firsts line are made to define all the variables such as count, **oldestPage**, **oldestTime**, **firstPage**, **map_count** and **mapped**.

We can see some loops. The main one is the while loop. In this loop there are several if loop made to identify if the tested page is the first or the last created page.
The program stay in the loop while all the pages (stocked in the vector **mem**) are not all tested or while the Boolean variable called **mapped** is not set at true.

The algorithm test first if the current page is linked or not to a physical page (line 16).
Then if it’s true (the virtual page is link to a physical one) the line 17 check if the page is the first one . Just after at the line 20 the algorithm is looking for the oldest page. To find the oldest one the algorithm try **if(page.inMemTime > oldestTime).**
To make it easier we can take a case where we create four virtual pages which are all linked to a physical page. In this case we will have 40, 30, 20 and 10 of in memory time (in µs).

In this case the algorithm will not do anything really interesting but if some modification occurred and then the in memory times are looking like 20, 10, 40 and 30 µs so the algorithm will find that the oldest page is the third one (40 µs) because at the **inMemTime** test the third one is greater than the two first one (20 and 10 µs).
So, after the detection of that, the algorithm will keep in memory the position of the third page in the vector **mem** in the **oldestPage** variable and then the Boolean variable **mapped** takes the value true to quit the loop.

In the case were the algorithm can’t find any modification after checked all the pages the Boolean variable **mapped** takes the value true to quit the loop.

When the algorithm is out of the loop it know which page is the oldest one. With that it can retrieve the information and store it in a new virtual page which result a page fault.
The line 34 allow the algorithm to get the information of the page to replace (the page fault).

Then in the GUI, the address of the physical page from the virtual page is deleted and after the address is put in the field physical page of the page which need to be modified.

Line 39 to 43 are made to reset all the value by default.

So, finally this algorithm is made to update virtual pages which don’t have physical page linked.