Matthew Wyman  
Matthew James  
Connor Smith  
10/20/15

**Design Problem #2: Pancake Sort**

|  |  |
| --- | --- |
| Stack Size | Time ) |
| 5 | 20 |
| 10 | 12 |
| 15 | 16 |
| 20 | 20 |
| 25 | 25 |
| 30 | 31 |
| 40 | 44 |
| 50 | 55 |
| 60 | 72 |
| 70 | 89 |
| 80 | 109 |
| 90 | 129 |
| 100 | 151 |

For this design problem we implemented a pancake sort using a simple algorithm. We find the biggest pancake that is not in the right order and we flip all the pancakes above it so that it is at the top. After we do that we flip the stack so that it goes to the right place. We then find the next largest pancake that is out of order and repeat the process until this is done. This will take time. While not being the most efficient algorithm it is very simple. It is based off selection sort.

The recorded time increases as expected with an anomaly with a small stack of 5. We believe that it is due to some startup factors like having a cold cache and probably having a page fault that we miss in later simulations. Other than that the time does increase as expected.