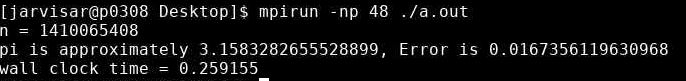
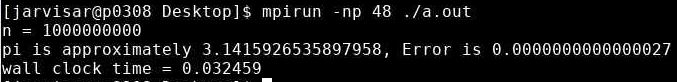
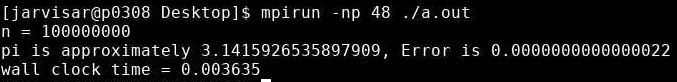
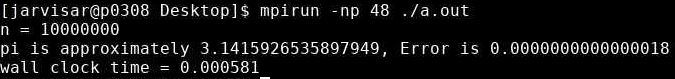
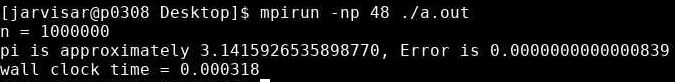
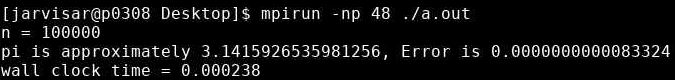
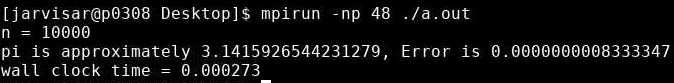
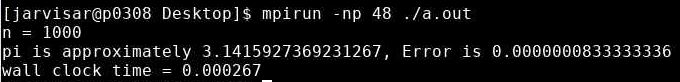
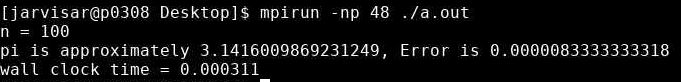
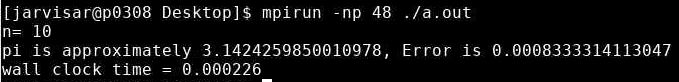
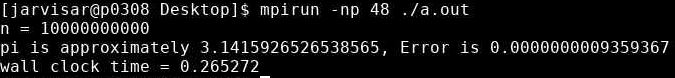
Adam Jarvis Assignment 7

Below is the output of picalc.c after modifying the number of intervals from 10^1 to 10^10:



Unfortunately, due to truncation, n = 10,000,000,000 was truncated to 1,410,065,408. However, after changing the code to accept a long integer, I was able to have the program run for n = 10,000,000,000:



Below is a table listing all the error and wall clock times for the given number of intervals:

|  |  |  |
| --- | --- | --- |
| n | Error | Wall Clock Time |
| 10 | 0.0008333314113047 | 0.000226 |
| 100 | 0.0000083333333318 | 0.000311 |
| 1000 | 0.0000000833333336 | 0.000267 |
| 10000 | 0.0000000008333347 | 0.000273 |
| 100000 | 0.0000000000083324 | 0.000238 |
| 1000000 | 0.0000000000000839 | 0.000318 |
| 10000000 | 0.0000000000000018 | 0.000581 |
| 100000000 | 0.0000000000000022 | 0.003635 |
| 1000000000 | 0.0000000000000027 | 0.032459 |
| 10000000000 | 0.0000000009359367 | 0.265272 |

As we can see from the charts above, error is drastically reduced while wall clock time is increased as we increase the number of intervals from 10 to 10,000,000,000.