Ex04_p01

- To optimise loop (a) I simply removed the conditional statement and broke the loop into two parts like so:

Original loop:
p01_a.cpp
```
for (int i = 0; i < n - 2; i++) {
    if (i < 500) a[i] = 4.0 * b[i] + b[i+1];
    else a[i] = 4.0 * b[i+1] + b[i];
}
```

Optimised loop:
p01_a_optimised.cpp
```
for (int i = 0; i < 500; i++) {
    a[i] = 4.0 * b[i] + b[i+1];
}

for (int i = 500; i < n-2; i++) {
    a[i] = 4.0 * b[i+1] + b[i];
}
```

- To optimise loop (b) I changed from n stride to unit stride to optimise memory access like so:

Original loop:
p01_b.cpp
```
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        a[j][i] = b[j][i] / c[i];
    }
}
```

Optimised loop:
p01_b_optimised.cpp
```
for (int i = 0; i < n; i++) {
    for (int j = 0; j < n; j++) {
        a[i][j] = b[i][j] / c[i];
    }
}
```

- To measure CPU time I compiled four binaries p01_a p01_a_optimised p01_b p01_b_optimised and executed them each n = 50 times using the included Makefile and shell script “script.sh”. Finally I calculated the average CPU time for each binary using the python script “medians.py”. Below I have included the average CPU time for each

loop compiled with no optimisation, -O0 and -O3.

No optimisation:

```
Average of p01_a: 0.032825908 s
Average of p01_a_optimised: 0.029589669000000006 s
Average of p01_b: 0.12175619000000001 s
Average of p01_b_optimised: 0.034940662 s
```

-O0:

```
Average of p01_a: 0.03268100399999999 s
Average of p01_a_optimised: 0.029670404999999997 s
Average of p01_b: 0.12273508999999999 s
Average of p01_b_optimised: 0.035056594999999996 s
```

-O3:

```
Average of p01_a: 2.5370000000000005e-08 s
Average of p01_a_optimised: 2.4170000000000004e-08 s
Average of p01_b: 0.050251766999999996 s
Average of p01_b_optimised: 0.0019204685 s
```