Hayden Miedema and Douglas Money

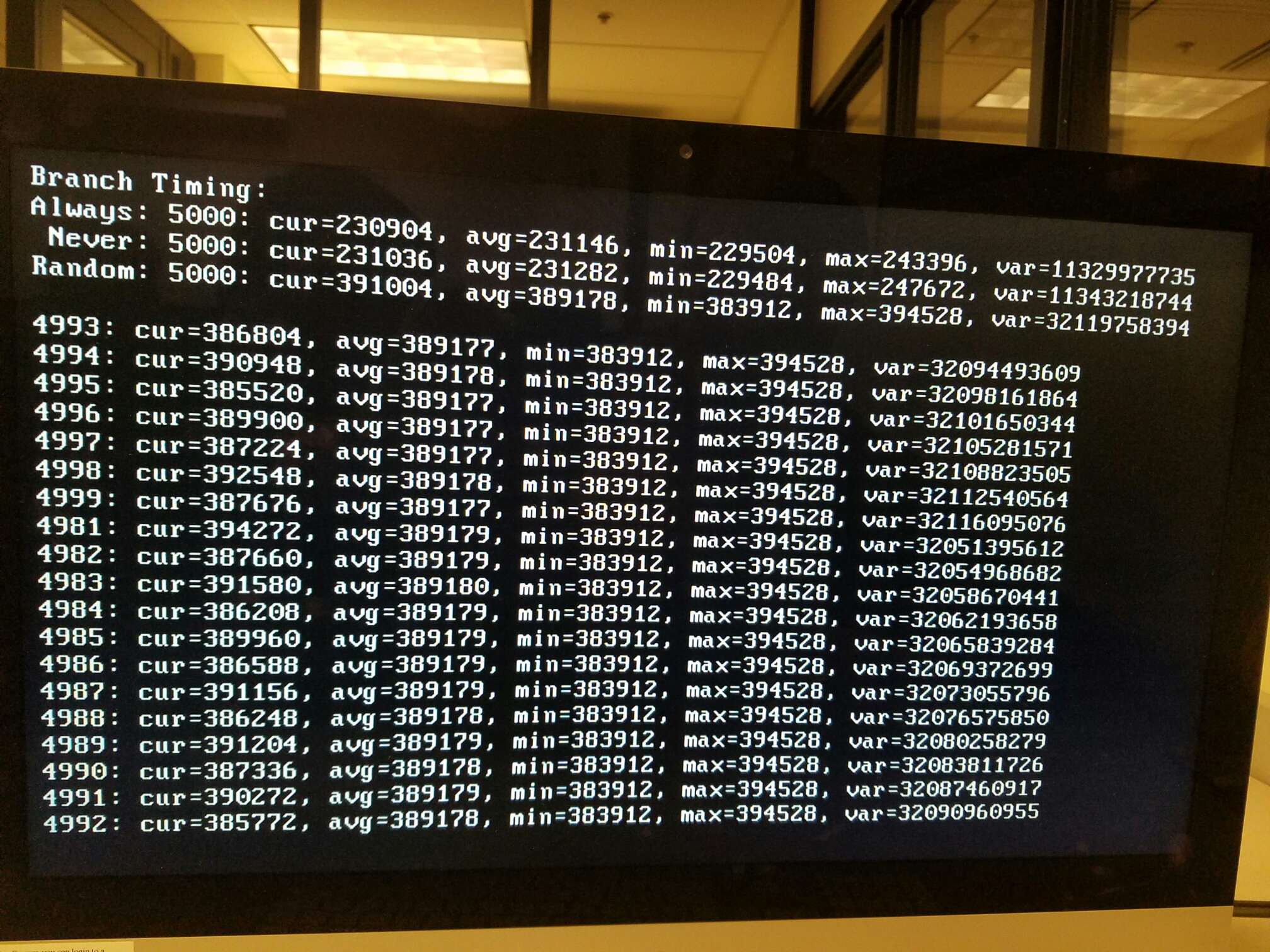
CIS 451

Professor Kurmas

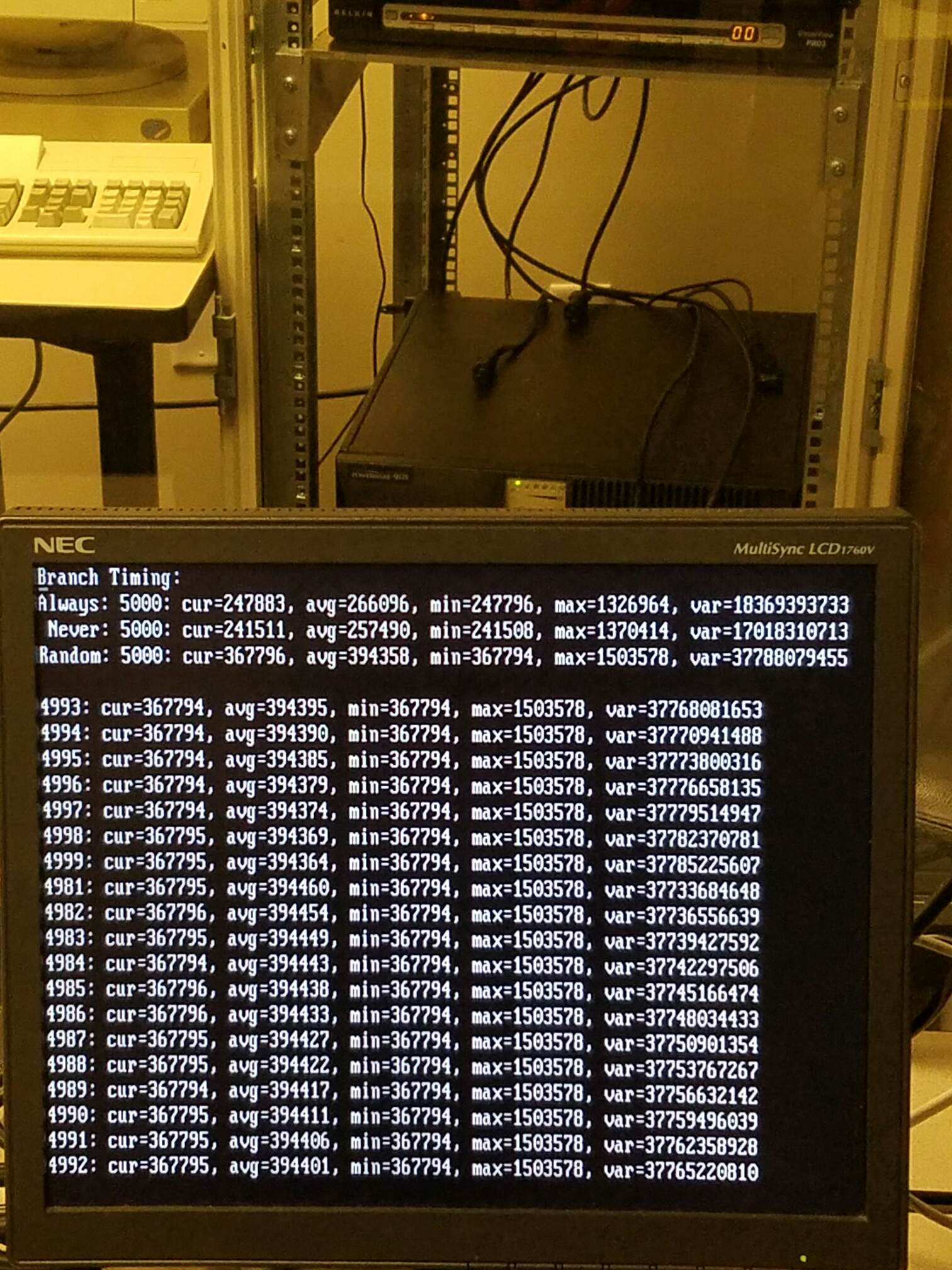
Lab 8: Branch Predictors

1. The sample method time\_never\_taken below won't generate a useful measurement. Explain why not.
   1. Because this loop never will take the else path. The if(true) statement will continue running forever. The timing will only tell us how long it took for that many ‘iterations’ (the variable in the for loop) of the if part of the loop to finish.
2. The sample method time\_never\_taken below won't generate a useful measurement. Explain why not.
   1. Again, we cannot be sure here that the if and the else part of the statements will get hit
3. The sample method time\_branch below makes two mistakes. Identify and fix them.
   1. sum \*= 5910; and sum += 19; should be doing the same operation. So one must switch, sum \*= 19; would be sufficient.
   2. Here the if part of the statement will once again dominate the action of the loop and run far more times than an else statement. All this does is check
4. Edit the repeat\_measurements function so that it
   1. creates a branch pattern based on pattern\_id, and
   2. collects the data to be reported.
5. Your code that generates the branch pattern should always call random(), even when generating an "always taken" or "never taken" pattern. Why? (Hint, the answer is given earlier in this document.
   1. So that the timing remains consistent throughout each of the RANDOM,ALWAYS, or NEVER sequences. The random() call doesn’t take a substantial amount of time but is a complicated function that we should take into account.
6. Build this new instance of TimingOS, run it (1) using ./timingos.debug, (2) using a virtual machine, (3) as an actual OS on the Mac hardware (i.e., by copying the ISO to a USB drive and booting from it), and (4) as an actual OS on the AMD machine in the back of the Arch lab. Describe and explain the differences do you see between the "user space", "virtual machine", and "real" results.
7. Explain how the results provide evidence that the machine has a branch predictor. (Remember, the results are in cycles, not seconds, so the absolute numbers are not directly comparable between machines.)
   1. The results provide evidence that the AMD has a branch predictor because the results are wildly different regarding the ALWAYS/NEVER sequences with the RANDOM sequence. A RANDOM sequence is known to cause problems for a branch predictor, therefore seeing much higher running times in the RANDOM column is expected.
8. How does the branch predictor in the Intel i7 (the Mac Hardware) differ from the AMD's branch predictor?
   1. The intel i7 has similar results to the AMD, but is faster in all 3 of the categories.

Intel:



AMD:



We can see that the intel even had a better average when random sequences were apparent.