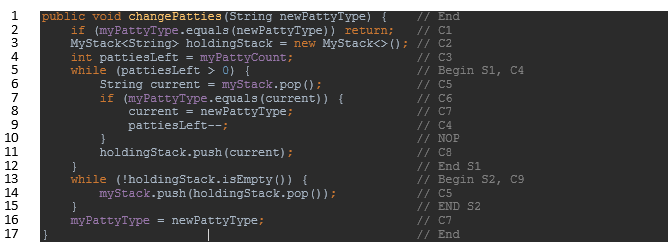
Grant Toepfer

TCSS 342

Assignment 1 – Burger Baron

Analysis



In the code above the comments show the line-by-line analysis of the costs. Line 2 checks to see if the new patty type is the same as the previous patty type and returns if it is. This transforms the cost of the function to a short O(1) for that case. For the rest of the analysis, however, we will pretend this if always evaluates to false and therefore will always have a cost of C1.

C2, C3, and C7 will always be ran once and therefore be combined into C10.

S1 will be run until the last patty is changed to the new patty type.

Inside S1: C5, C6 and C8 will be run on every iteration and C7 and C4 will be run when we find a patty. Lets pretend they are ran on every iteration as it will not effect our final answer. n is the number of ingredients between the top of the stack and the final patty, but for simplicity we’ll consider n to be the size of the entire stack since it won’t affect our final answer.

Inside S2: C5 and C9 will be ran on every iteration. The loop will end once all the ingredients are moved back to the other stack. This means that n on this loop is the same n from the S1, which again we will pretend to be the size of the entire stack.

We can get the runtime of the method by adding up all our sums and constants.