Proof from Lecture on 1/4

**Theorem 1** The memoized change-making algorithm calls the function \texttt{change()} at most $5n$ times, where $n$ is the input amount for which we want to make change.

**Proof:** A branching call to the function \texttt{change(a)} gets the answer by doing the recursive calls to \texttt{change(a-25)}, \texttt{change(a-10)} etc. Any other call, answered either by using \texttt{memoDictionary} or by returning one or infinity, we shall refer to as a memo call. There are at most $n$ branching calls, one for each $1 < a \leq n$. Each branching call makes at most four recursive calls. A memo call makes no recursive calls. So there are at most $4n$ recursive calls to \texttt{change()}. This makes $5n$ total, the $n$ branching calls plus their $4n$ recursive calls.