Carolyn Kao (chkao831@stanford.edu)

CME 241, Winter 2021

HW11

1.

Here, we consider a simple case of Monte-Carlo Prediction where the MRP consists of a finite state space with the non-terminal states $\mathcal{N} = \{s_1, s_2, \ldots, s_m\}$. In this case, we represent the Value Function of the MRP in dictionary of (state, expected return) pairs, the relevant code is sketched out as the following,

```
S = TypeVar('S')

def tb_mc_pred(traces: Iterable[Iterable[mp.TransitionStep[S]]]
               gamma: float,
               tol: float=1e-08,
               vf: Dict[S,float])
               ->List[Mapping[S,float]]:

    episodes=[returns(trace,gamma,tol) for trace in traces]

    vf:List[Mapping[S,float]]=[]
    count:Mapping[S,int]=defaultdict(lambda:0)

    for episode in episodes:
        if(len(vf) == 0):
            vn: Mapping[S,float] = defaultdict(lambda:0)
        else:
            vn: Mapping[S,float] = {key:val for (key,val) in vf[-1].items()}
        for epi in episode:
            count[epi.state]+=1
            vn[epi.state] = vn[epi.state] + 1.0/(count[epi.state])*(epi.return_-vn[epi.state])
        vf.append(vn)
    return vf
```

In the code, we make update to the value function for state s_i by the weight for its latest trace experience return times the difference between the latest trace experience return and the current Value Function estimate. The formula is mathematically presented by:
$V_n\left(s_i\right) = \frac{n-1}{n} \cdot V_{n-1}\left(s_i\right) + \frac{1}{n} \cdot Y_i^{(n)} = V_{n-1}\left(s_i\right) + \frac{1}{n} \cdot \left(Y_i^{(n)} - V_{n-1}\left(s_i\right)\right)$