1.

a.

s	a	s'	p(s'|s,a)	r(s,a,s')	States indexes indicate the following: Coin 1 Value, Coin 2 Value, Where Arm is Over
H,H,1	Call Fairy	H,H,1	1	20	
H,H,2	Call Fairy	H,H,2	1	20	
H,H,1	Move	H,H,2	1	-2	
H,H,2	Move	H,H,1	1	-2	
H,H,1	Flip	H,H,1	0.5	-1	
H,H,1	Flip	T,H,1	0.5	-1	
H,H,2	Flip	H,T,2	0.5	-1	
H,H,2	Flip	H,H,2	0.5	-1	
T,T,1	Call Fairy	T,T,1	1	10	
T,T,2	Call Fairy	T,T,2	1	10	
T,T,1	Move	T,T,2	1	-2	
T,T,2	Move	T,T,1	1	-2	
T,T,1	Flip	T,T,1	0.5	-1	
T,T,1	Flip	H,T,1	0.5	-1	
T,T,2	Flip	T,T,2	0.5	-1	
T,T,2	Flip	T,H,2	0.5	-1	
T,H,1	Call Fairy	T,H,1	1	-5	
T,H,2	Call Fairy	T,H,2	1	-5	
T,H,1	Move	T,H,2	1	-2	
T,H,2	Move	T,H,1	1	-2	
T,H,1	Flip	T,H,1	0.5	-1	
T,H,1	Flip	H,H,1	0.5	-1	
T,H,2	Flip	T,H,2	0.5	-1	
T,H,2	Flip	T,T,2	0.5	-1	
H,T,1	Call Fairy	H,T,1	1	-5	
H,T,2	Call Fairy	H,T,2	1	-5	
H,T,1	Move	H,T,2	1	-2	
H,T,2	Move	H,T,1	1	-2	
H,T,1	Flip	H,T,1	0.5	-1	
H,T,1	Flip	T,T,1	0.5	-1	
H,T,2	Flip	H,H,2	0.5	-1	
H,T,2	Flip	H,T,2	0.5	-1	

b. This is continuing as even when the battery runs out, the robot will continue to operate and as such will not end

c. Make it such that when the robot runs out of charge, it will stop operating

2.

a. $q_\pi(s,a) = E_\pi[R_{t+1} + \gamma V_\pi(S_{t+1}) | S_t = s \,, A_t = a]$

b. $\sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')]$ as $q_\pi(s,a) \sum_{,a} \pi(a|s) = v_\pi(s)$ and $v_\pi(s) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] \sum_{,a} \pi(a|s)$

3.

a. V(s)=$\sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] \sum_{,a} \pi(a|s)$

Starting with s=Z

Therefore $\sum_{s',r} p(s',r|Z,a)[r + \gamma v_\pi(s')] \sum_{,a} \pi(a|Z)$=V(Z)

= 0.5* 0.8V(Z) + 0.5* 0.8V(Z)

V(Z)=0.8V(Z)

V(Z)=0

s=Y

$\sum_{,a} \pi(a|Y) \sum_{s',r} p(s',r|Y,a)[r + \gamma v_\pi(s')]$=V(Y)

= 0.5*1+0.5*5
V(Y)=3

s=X
$\sum_{,a} \pi(a|X) \sum_{s',r} p(s', r|X, a)[r + \gamma v_\pi(s')]$=V(X)
= 0.5(0.2 (25))
=2.5

s=W
$\sum_{,a} \pi(a|W) \sum_{s',r} p(s', r|W, a)[r + \gamma v_\pi(s')]$=V(W)
= 0.5*(3)+ 0.5(0.5*(4*0.8 *2.5) + 0.5(2+0.8*3))
= 4.1

b. We can change the policy such that $\pi(a|s) = \pi(b|s) = 0.85$. This is better than the above policy because if we redo the calculations above, we can get the following values:

V(Z) will remain zero not matter the policy
V(Y) = 0.85*1+0.85*5= 5.1 which is greater than 3
V(X)= 0.85(0.2 (25)) = 4.25 which is greater than 2.5
V(W) = 0.85*(5.1)+ 0.85(0.5*(4+0.8 *4.25) + 0.5(2+0.8*5.1)) = 10.064

This shows that for all values of s, the new policy gives us state values greater than the older policy

4.

V(W)

Episode 1 Val: 0 + 10+0 =10
Episode 2 Val: -10 + 0 = -10
Episode 3 Val: 6+2 =8
Episode 4 Value: 12+0=12
Final Val= (10 + -10 +8 +12)/ 4 = 5

V(Y)
Episode 1 Val: 0
Episode 2 Val: Non-existent
Episode 3 Val: 6
Episode 4 Value: 12
Final Val= (12 +6 + 0)/ 3 = 6

V(X)
Episode 1 Val: 0 +10 = 10
Episode 2 Val: 0

Episode 3 Val: Non-existent
Episode 4 Value: Nonexistent
Final Val= (10 + 0)/ 2 = 5

V(Z) = 0 as it is the terminal state

5.

a. $Q(S, A) += \alpha [R + \Upsilon \max_a Q(S', A) - Q(S , A)$
=0.5[4 + 1*16-0]
= 0.5*20
Q(W,a)=10

b. $Q(S, A) += \alpha [R + \Upsilon Q(S', A') - Q(S , A)$
=0.5[4 + 1*8-0]
= 0.5*12
Q(W,a)=6

6.

a. We use a greedy policy instead of an ε greedy policy as we want to display the episode with the maximum possible value. We are also not going to be exploring anymore hence we use a greedy rather than epsilon greedy. It will work with Sarsa as Sarsa only selects the action based off the policy passed in so it can work with a standard greedy policy.
b. This would work with Monte Carlo control(off policy only) because it will converge to the optimal policy π^* as the policy we learn is deterministic and as long as we estimate Q values for it, we learn a policy greedy to the previous estimated target policy we learn and eventually will converge. This however, would not work with On-Policy Monte Carlo control as that is not necessarily guaranteed to give us an optimal policy.