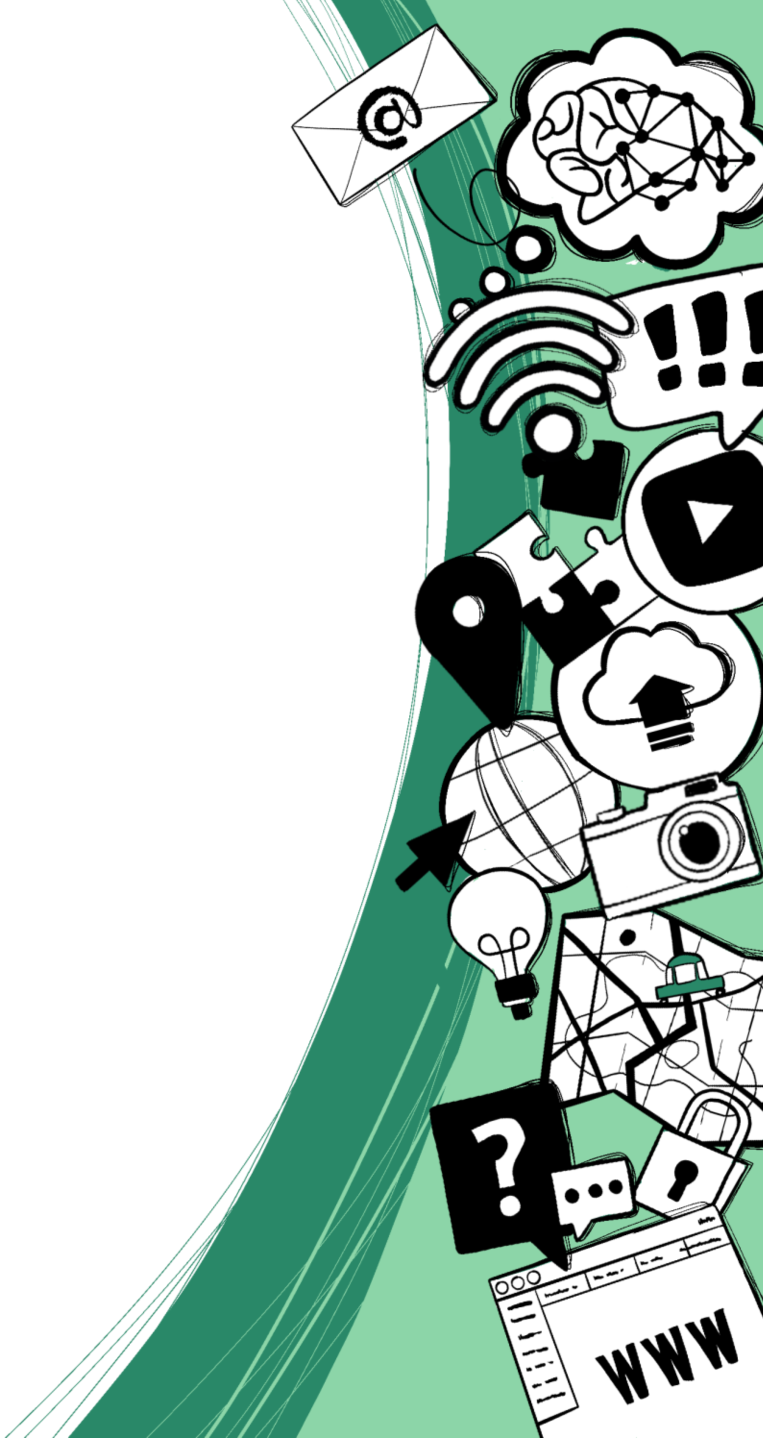
ENARIS

Coin Game



mobilis
interaktív kiállítási központ

TU Graz

OESTERREICHISCHE COMPUTER GESELLSCHAFT®
AUSTRIAN COMPUTER SOCIETY

Interreg
Austria-Hungary 2014-2020
European Union – European Regional Development Fund

Core Rules

Rules

- **2 players** and **5 coins**
- The players take turns either **taking one** or **taking two** coins
- The player taking the **last coin loses**

Example

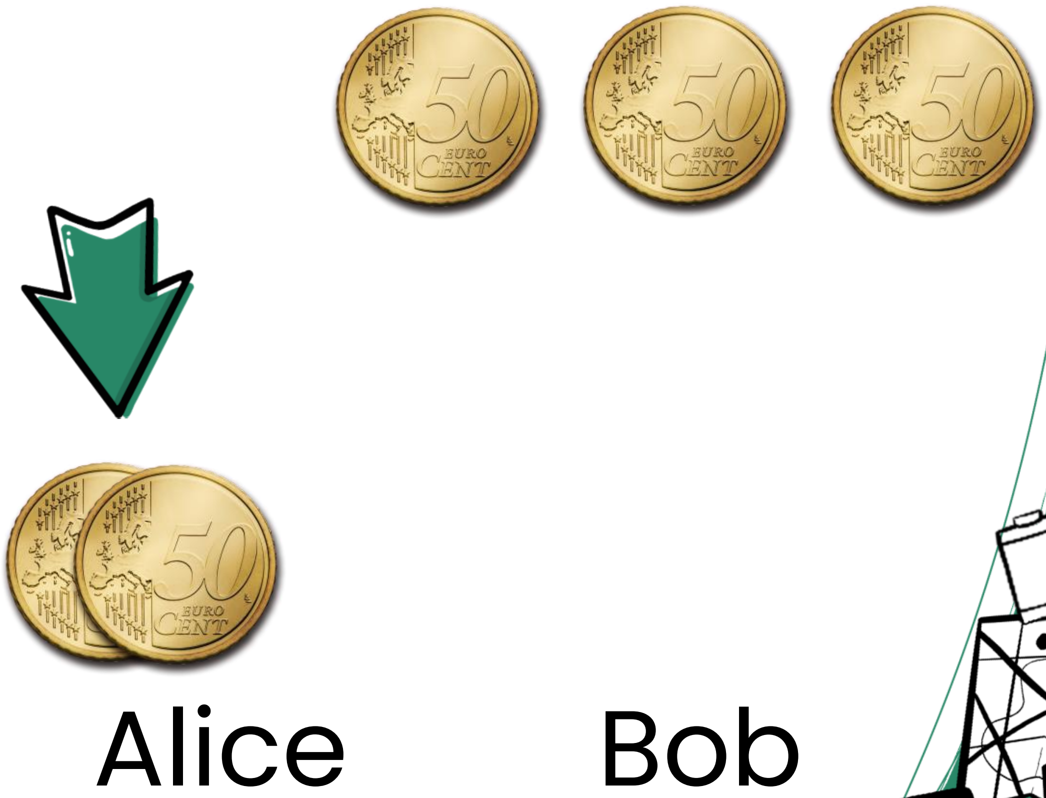
1. The game starts with **five coins**

Alice

Bob

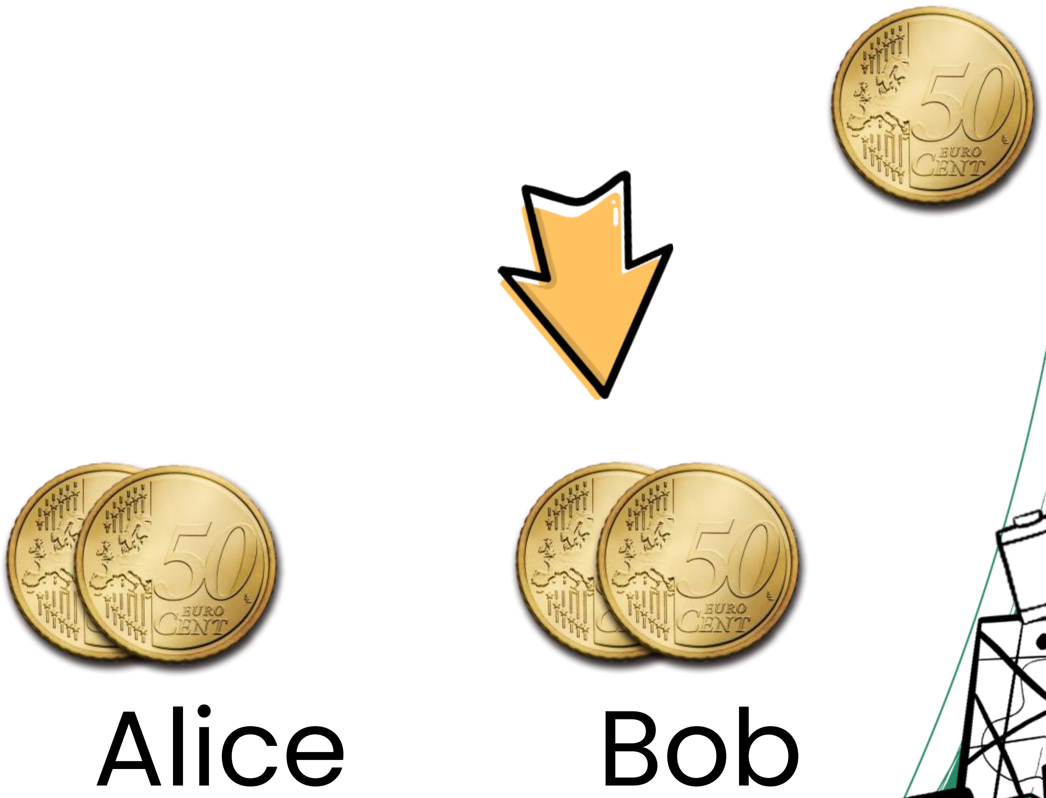
Example

1. The game starts with five coins
2. **Alice takes two coins**



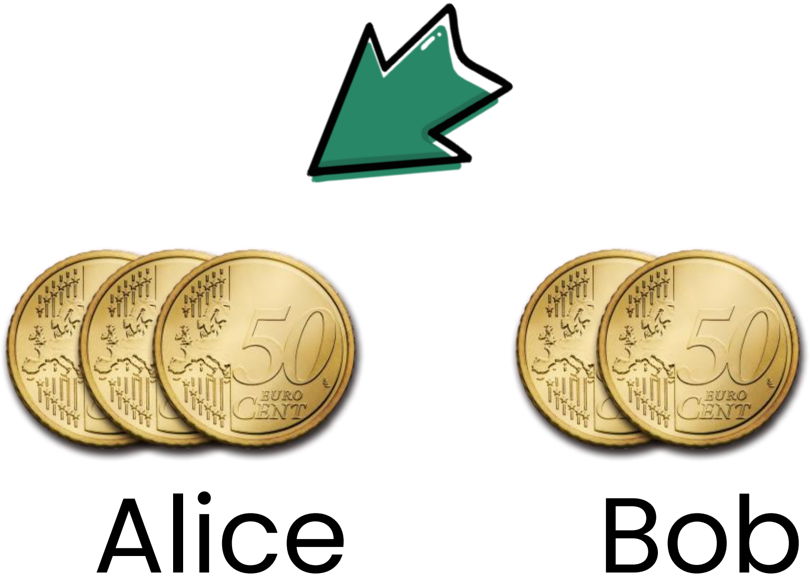
Example

1. The game starts with five coins
2. Alice takes two coins
3. **Bob takes two** coins



Example

1. The game starts with five coins
2. Alice takes two coins
3. Bob takes two coins
4. **Alice** **takes one** coin

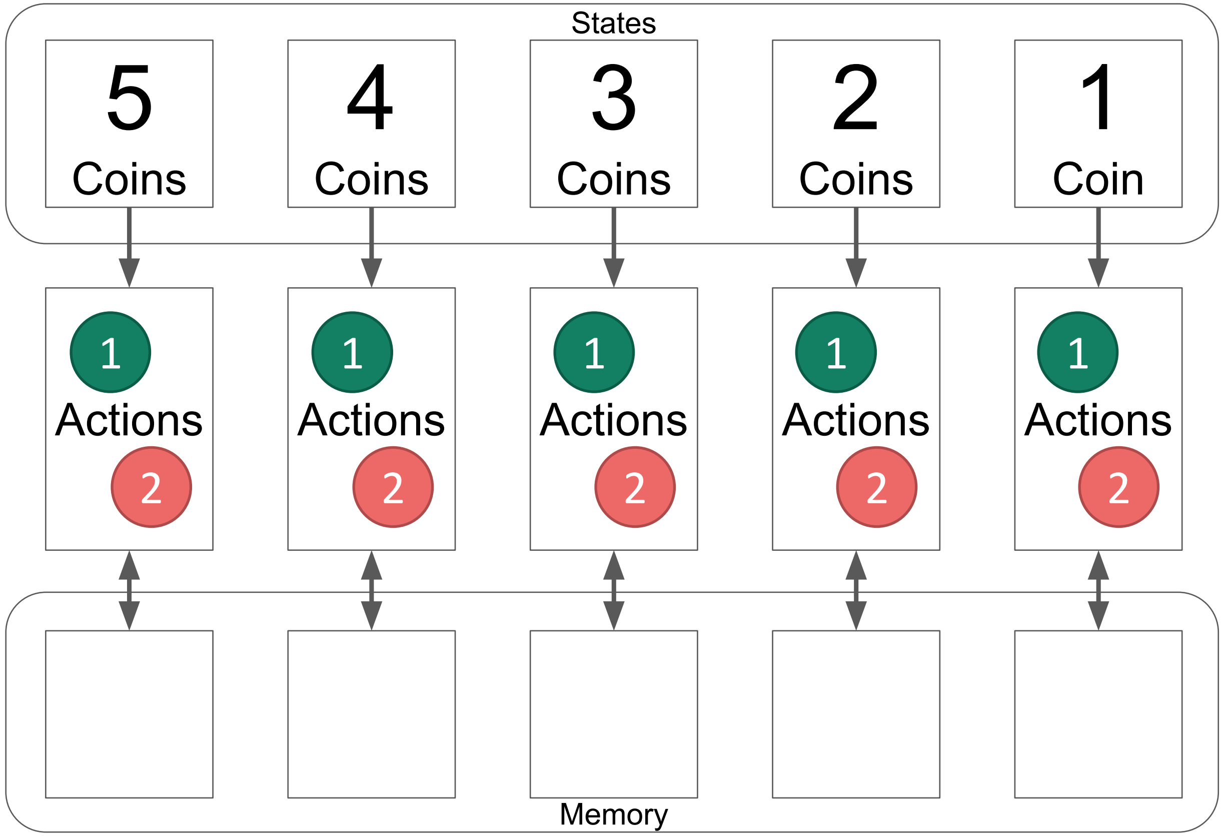


Example

1. The game starts with five coins
2. Alice takes two coins
3. Bob takes two coins
4. Alice takes one coin
5. **Alice loses**, she got the **last coin**!

Meet the AI

AI Board



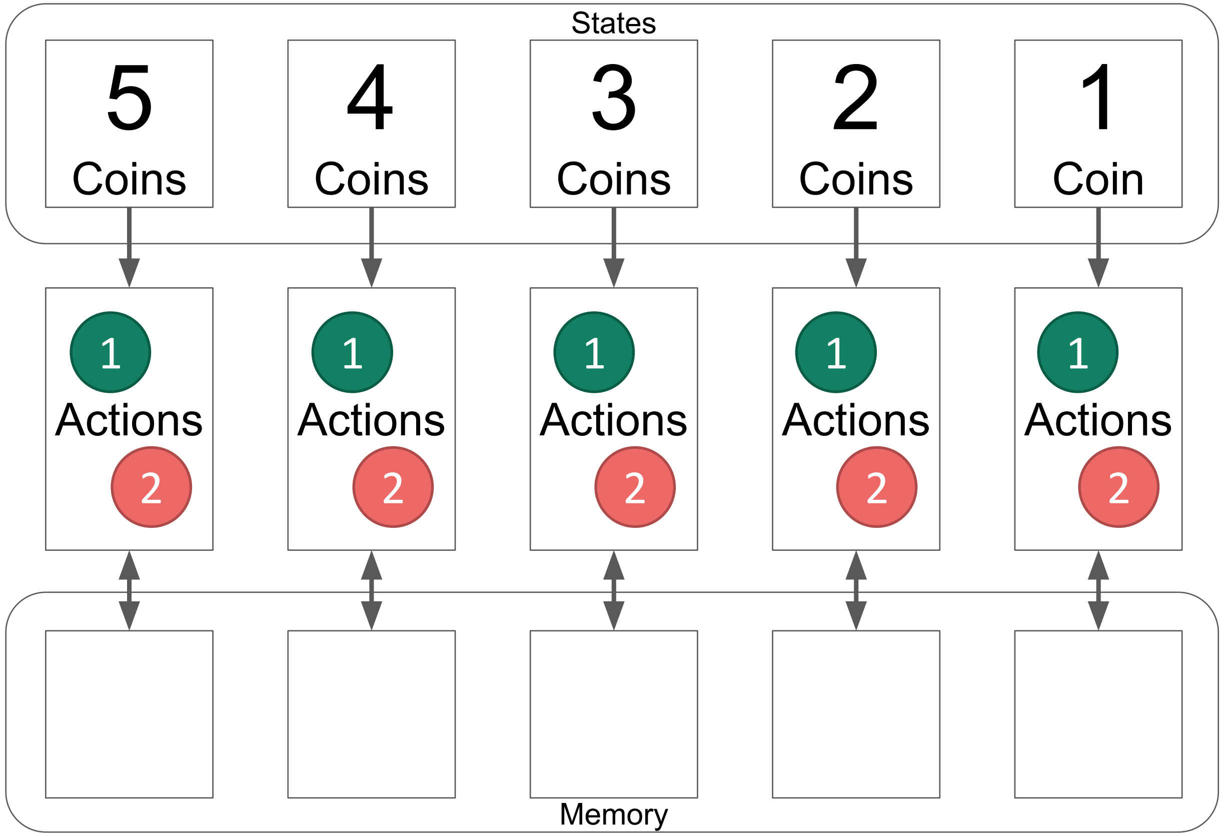
States

5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
1 Actions 2	1 Actions 2	1 Actions 2	1 Actions 2	1 Actions 2

Memory

current **state**
(coins on the table)

AI Board



States

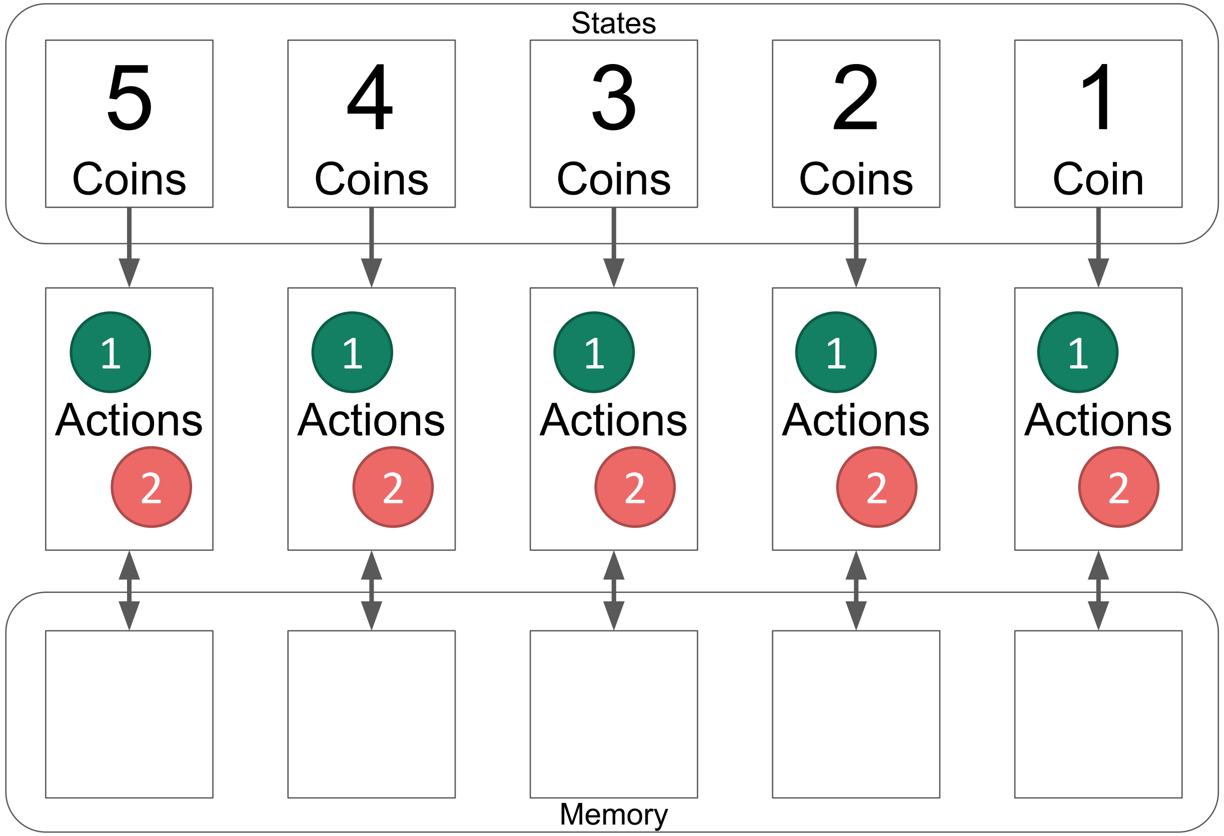
5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
1 Actions 2	1 Actions 2	1 Actions 2	1 Actions 2	1 Actions 2

Memory

current **state**
(coins on the table)

available **actions**
(**take 1** or **2** coins)

AI Board



States

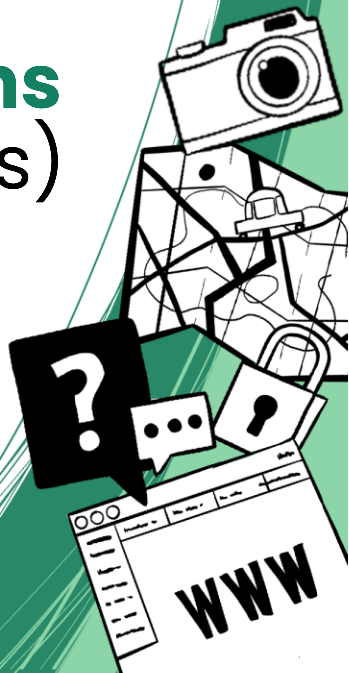
5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
1 Actions 2	1 Actions 2	1 Actions 2	1 Actions 2	1 Actions 2

Memory

current **state**
(coins on the table)

available **actions**
(**take 1** or **2** coins)

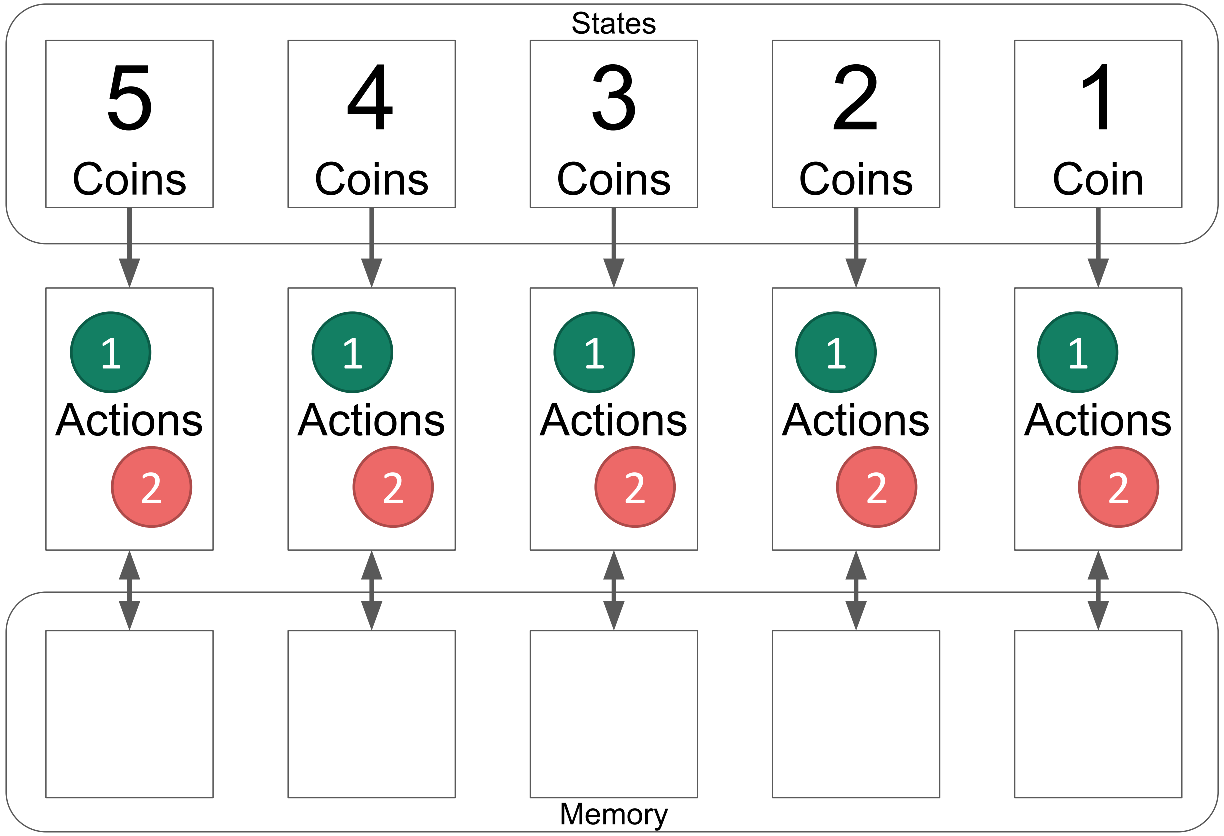
Memory
(**remember**
last **action**)



Rewards

- **Remove last action**, whenever the AI…
    - has **lost**
    - has chosen an **invalid action**
    - **can’t find** an **action**
- **Change nothing**, on all other occurrences

Game 1



States

5 Coins

4 Coins

3 Coins

2 Coins

1 Coin

1 Actions 2

1 Actions 2

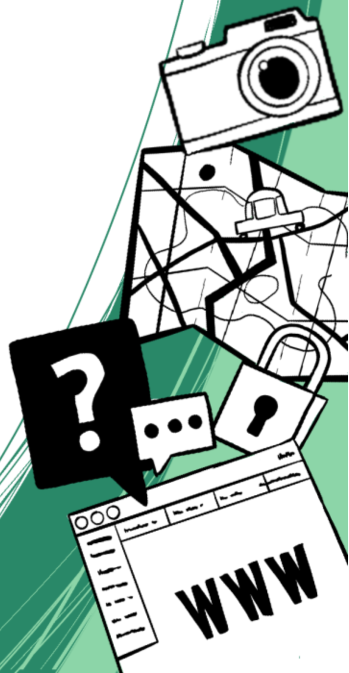
1 Actions 2

1 Actions 2

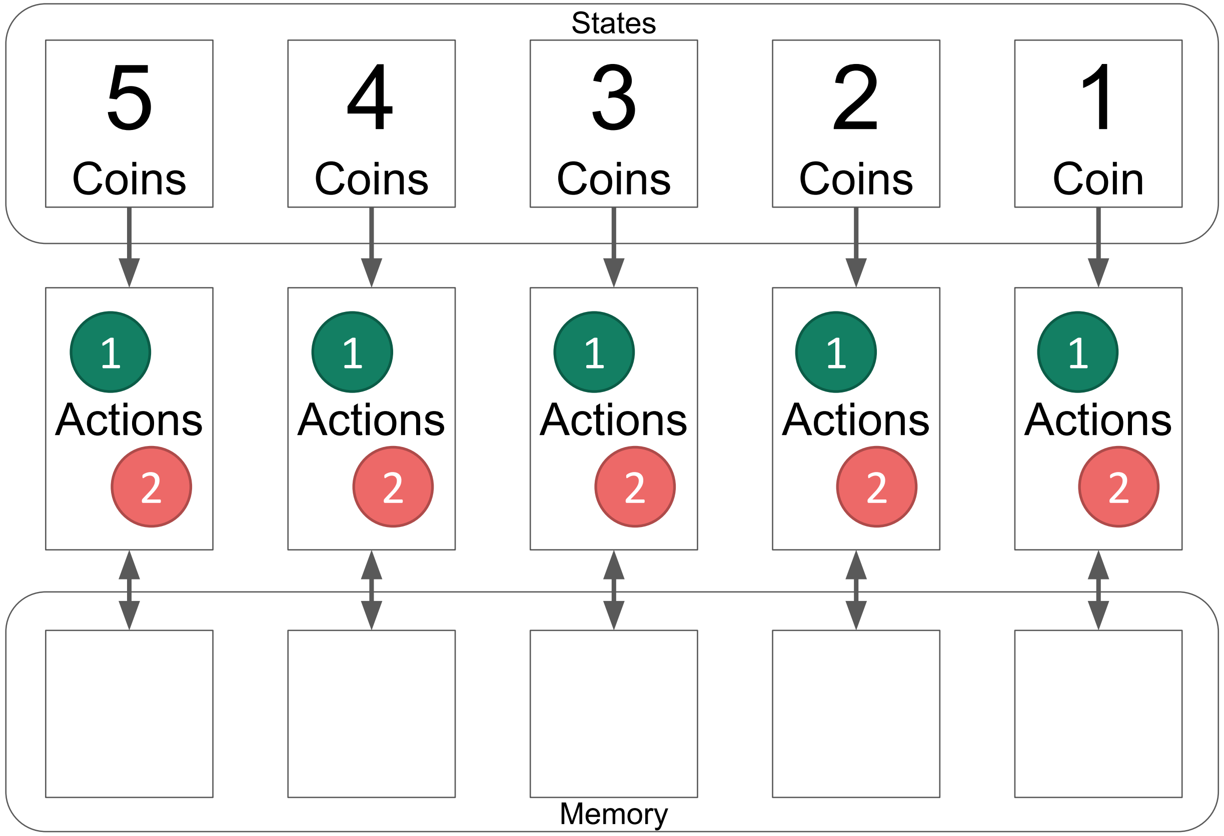
1 Actions 2

Memory

1. The game starts with five coins, **Alice** plays like **normal** and **Bob** plays as the **AI**



Game 1



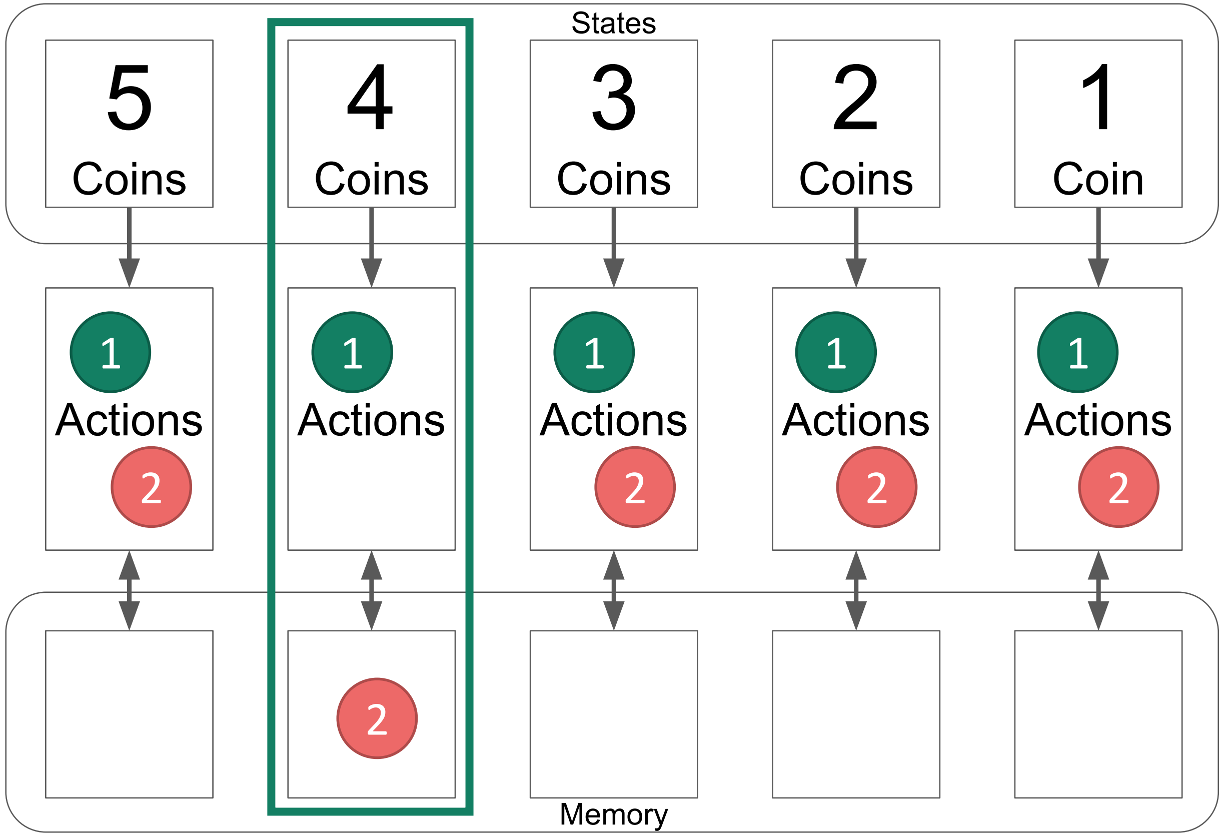
States

5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
1 Actions 2	1 Actions 2	1 Actions 2	1 Actions 2	1 Actions 2

Memory

1. The game starts with five coins, **Alice** plays like **normal** and **Bob** plays as the **AI**
2. **Alice takes one** coin

Game 1



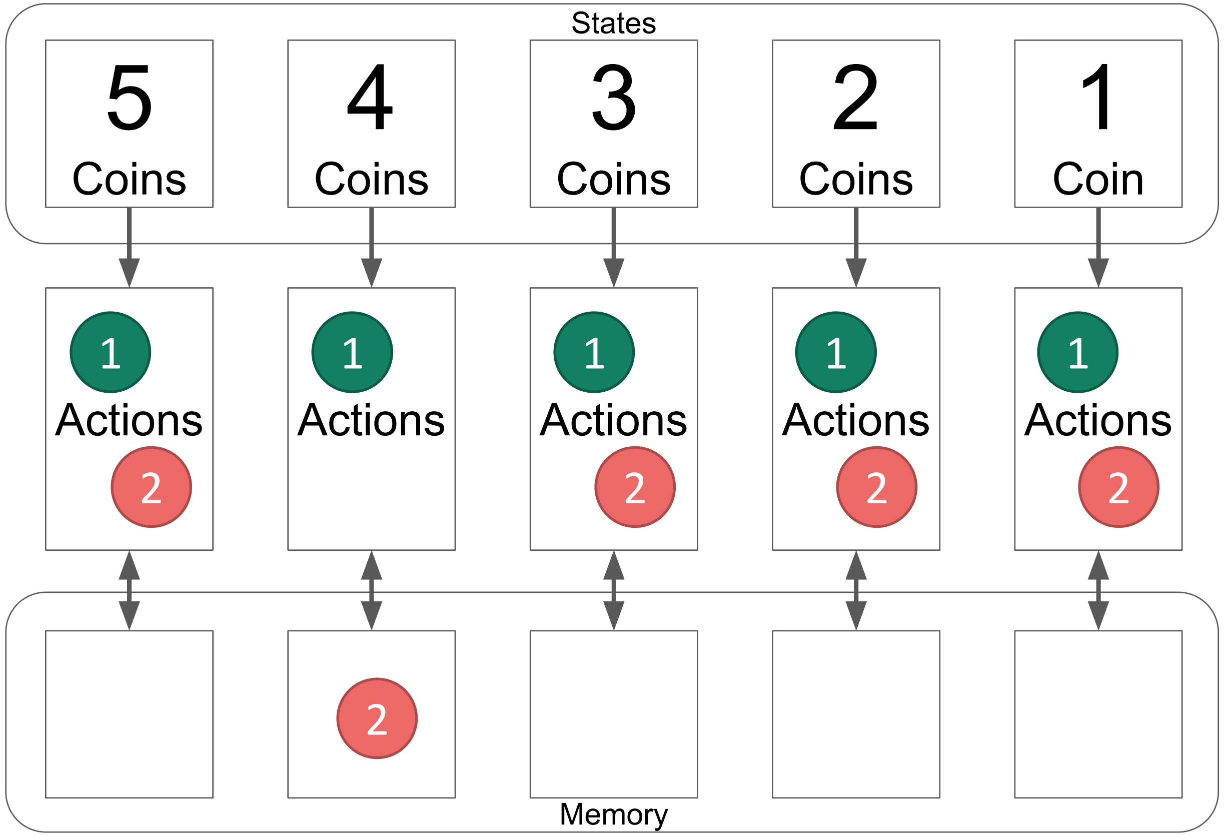
States

5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
1 Actions 2	1 Actions	1 Actions 2	1 Actions 2	1 Actions 2
	2			

Memory

3. There are **four coins** left, therefore **Bob** takes a **random action stone** from **state 4** and puts it into **memory**

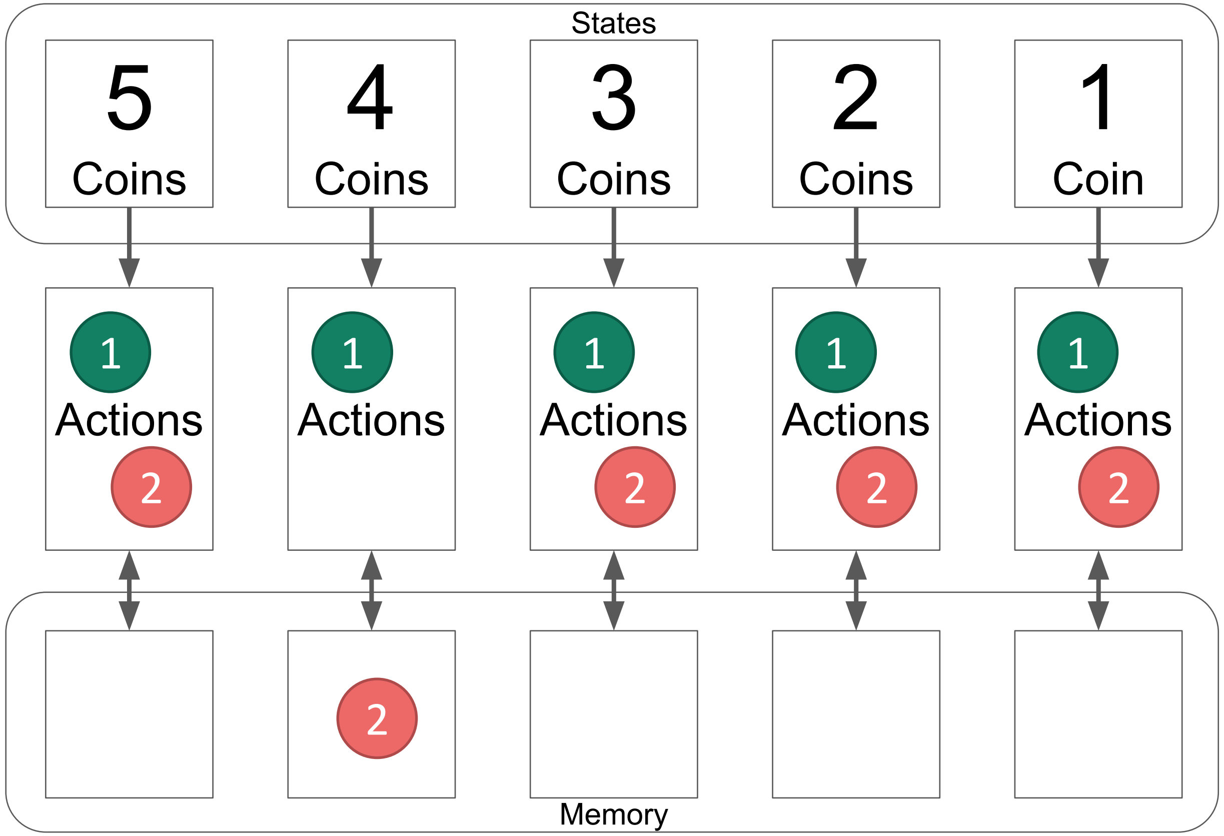
Game 1



3. There are **four coins** left, therefore **Bob** takes a **random action stone** from **state 4** and puts it into **memory**
4. **Bob** then performs the action and **takes two** coins

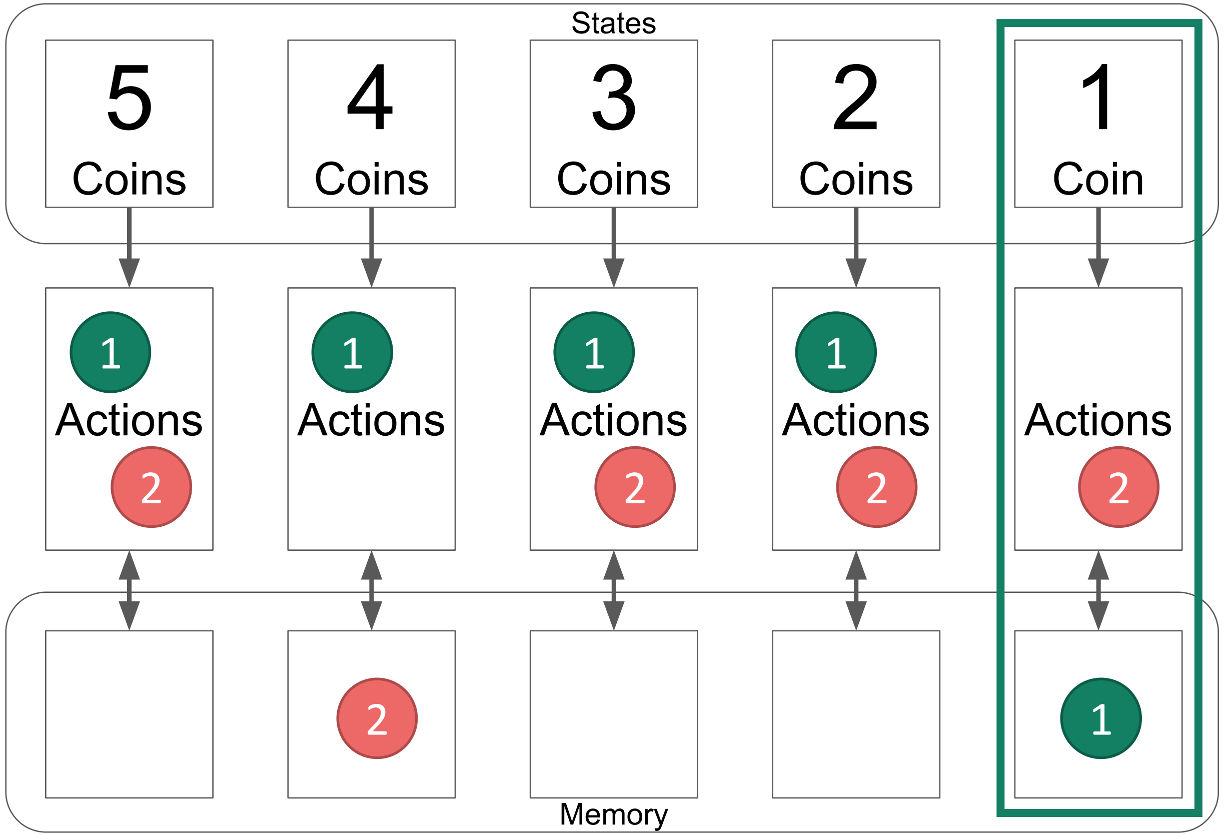
Game 1

5. **Alice takes one coin**



ENARIS

Game 1



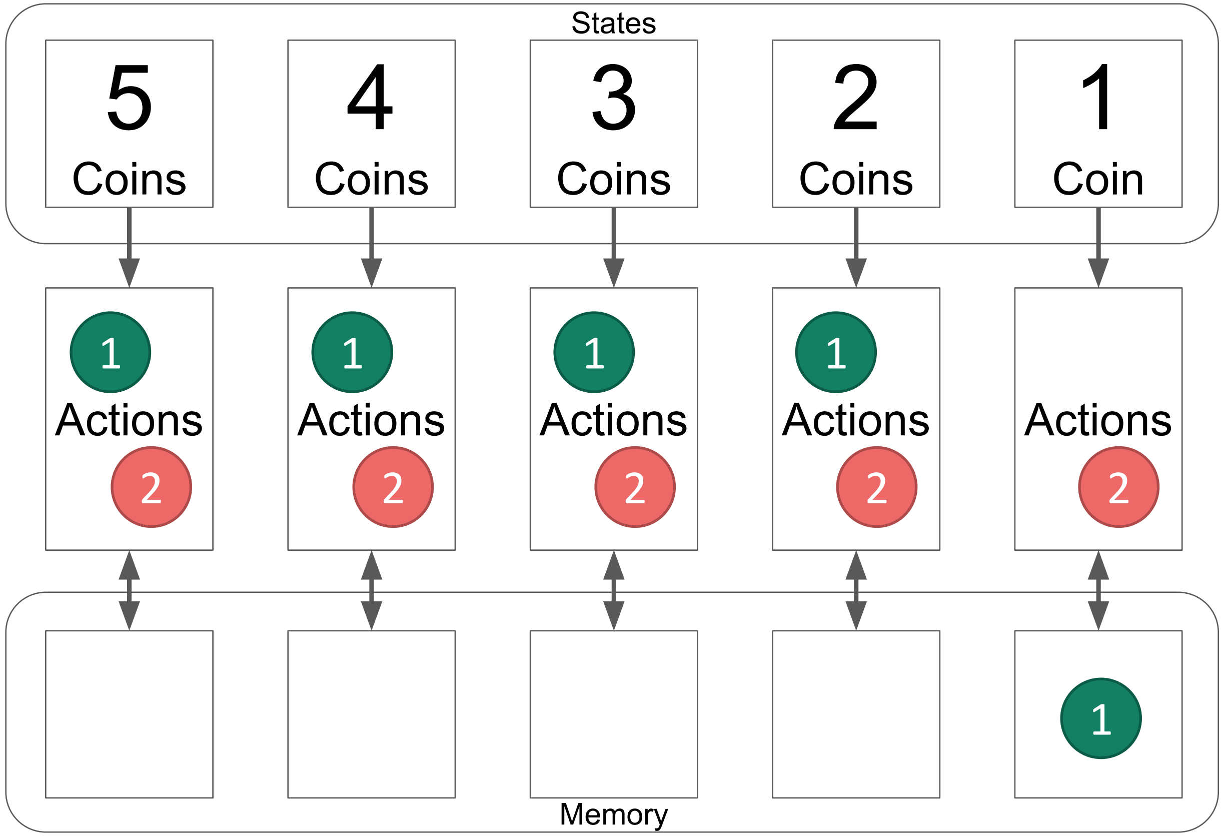
States

5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
1 Actions 2	1 Actions	1 Actions 2	1 Actions 2	Actions 2
	2			1

Memory

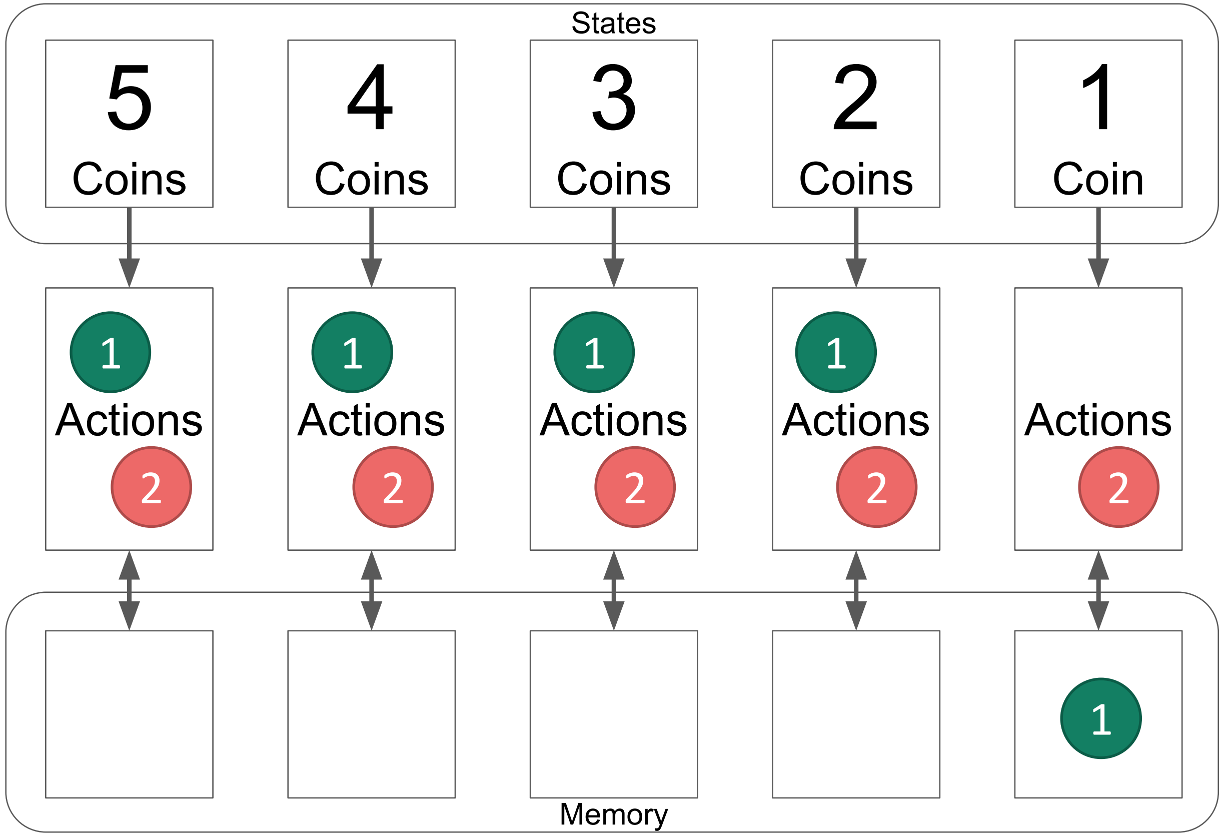
5. **Alice takes one** coin
6. There is **one coin** left, therefore **Bob** takes a **random action stone** from **state 1** and puts it into **memory**

Game 1



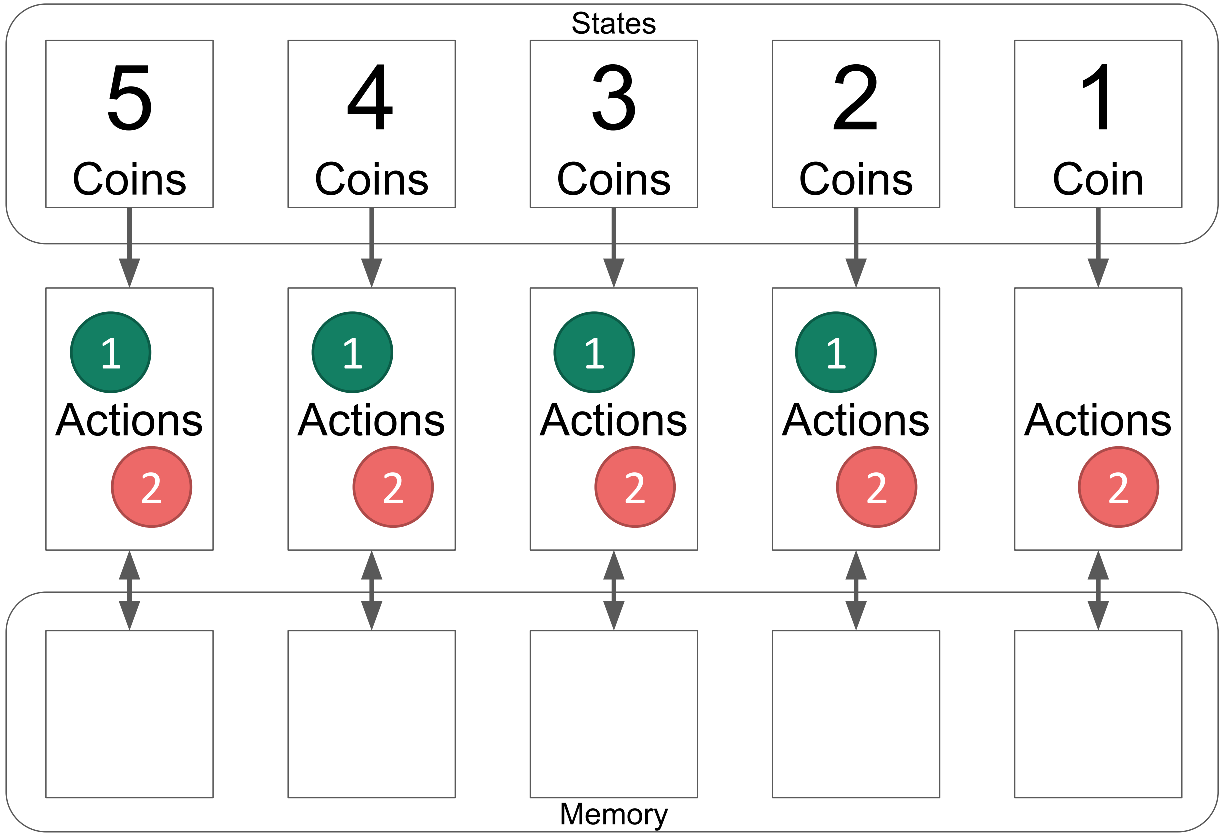
5. **Alice takes one** coin
6. There is **one coin** left, therefore **Bob** takes a **random action stone** from **state 1** and puts it into **memory**
7. **Bob** also returns the **stone** which was **previously in memory** back to its actions field

Game 1



8. **Bob** then performs the action and **takes one** coin

Game 1



States

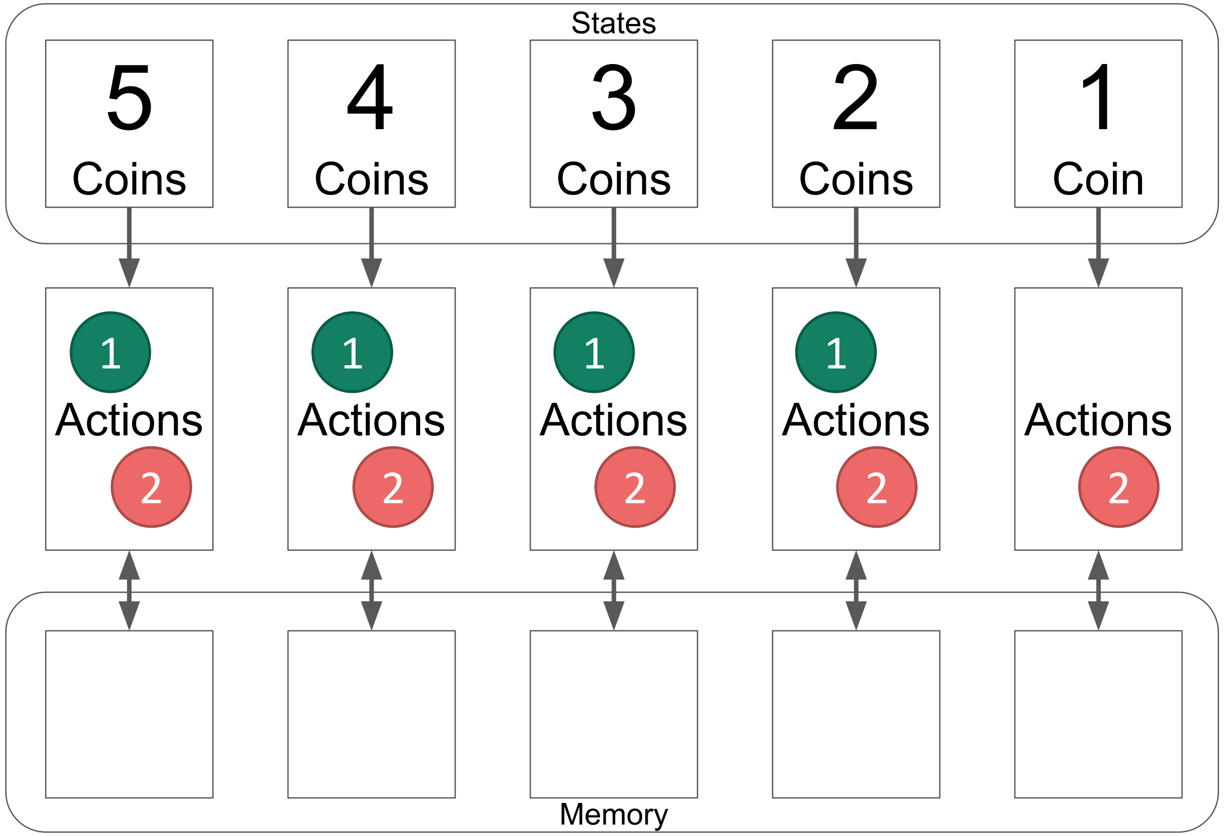
5 Coins | 4 Coins | 3 Coins | 2 Coins | 1 Coin

1 Actions 2 | 1 Actions 2 | 1 Actions 2 | 1 Actions 2 | Actions 2

Memory

8. **Bob** then performs the action and **takes one** coin
9. **Bob** has **lost**, therefore he **removes** the **stone in memory** from the game

Game 1



States

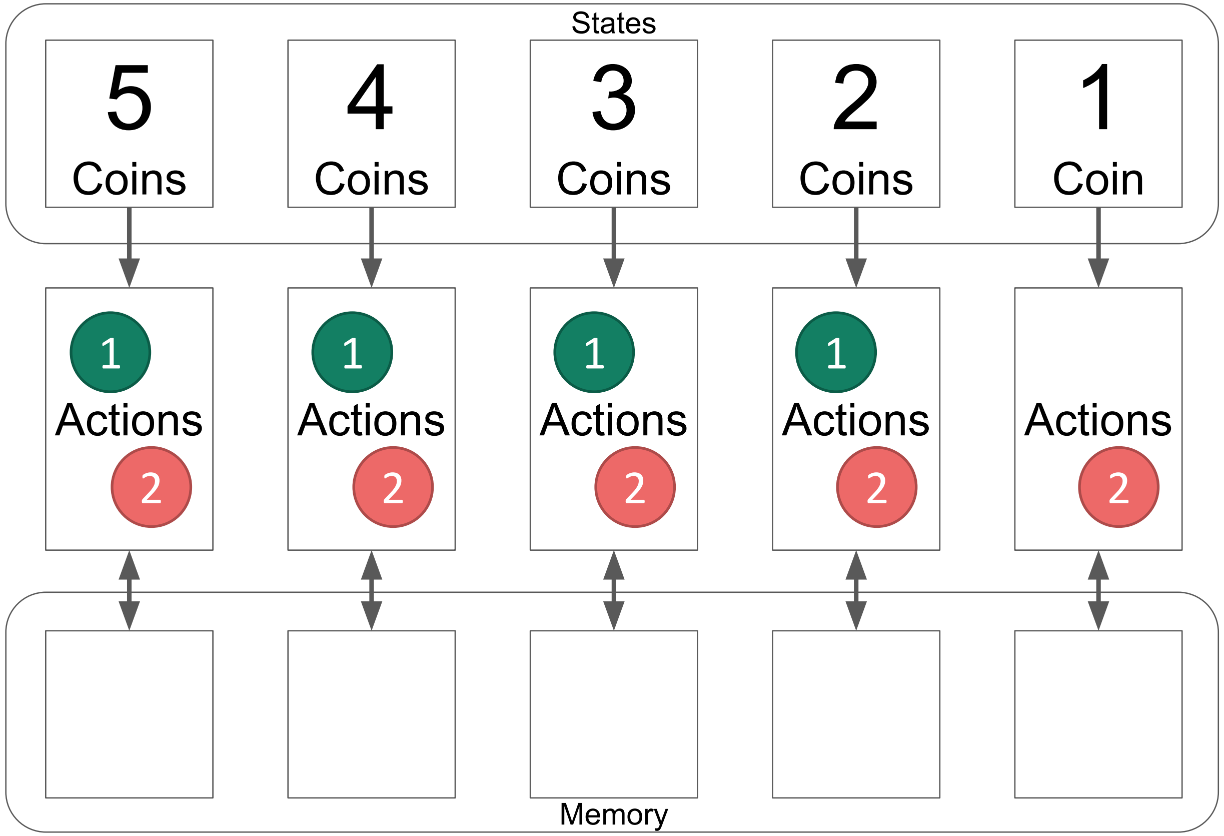
5 Coins | 4 Coins | 3 Coins | 2 Coins | 1 Coin

1 Actions 2 | 1 Actions 2 | 1 Actions 2 | 1 Actions 2 | Actions 2

Memory

8. **Bob** then performs the action and **takes one** coin
9. **Bob** has **lost**, therefore he **removes** the **stone in memory** from the game
10. Removed stones will **not return** for the following games!

Game 2



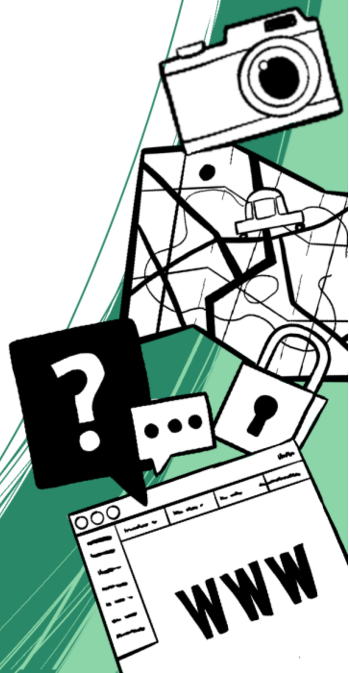
States

5 Coins | 4 Coins | 3 Coins | 2 Coins | 1 Coin

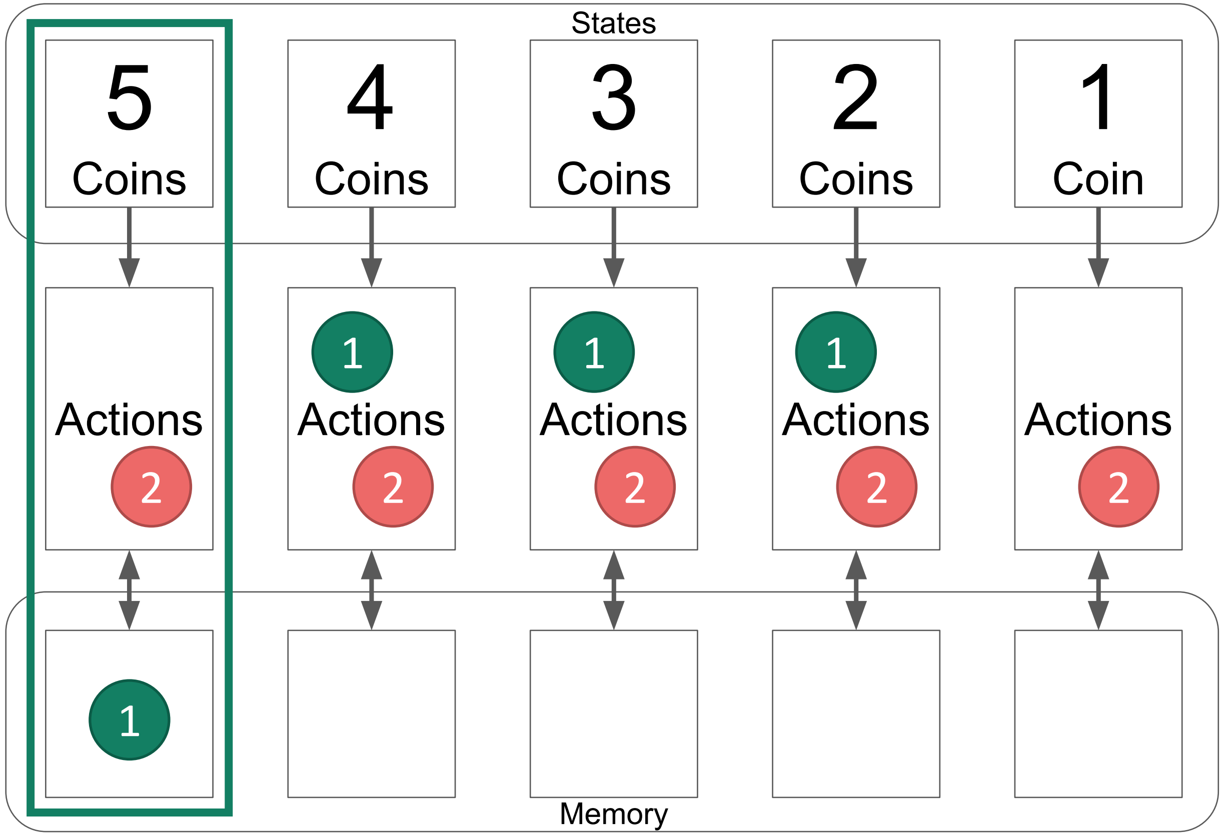
1 Actions 2 | 1 Actions 2 | 1 Actions 2 | 1 Actions 2 | Actions 2

Memory

1. The game starts again with five coins, this time **Bob** begins



Game 2



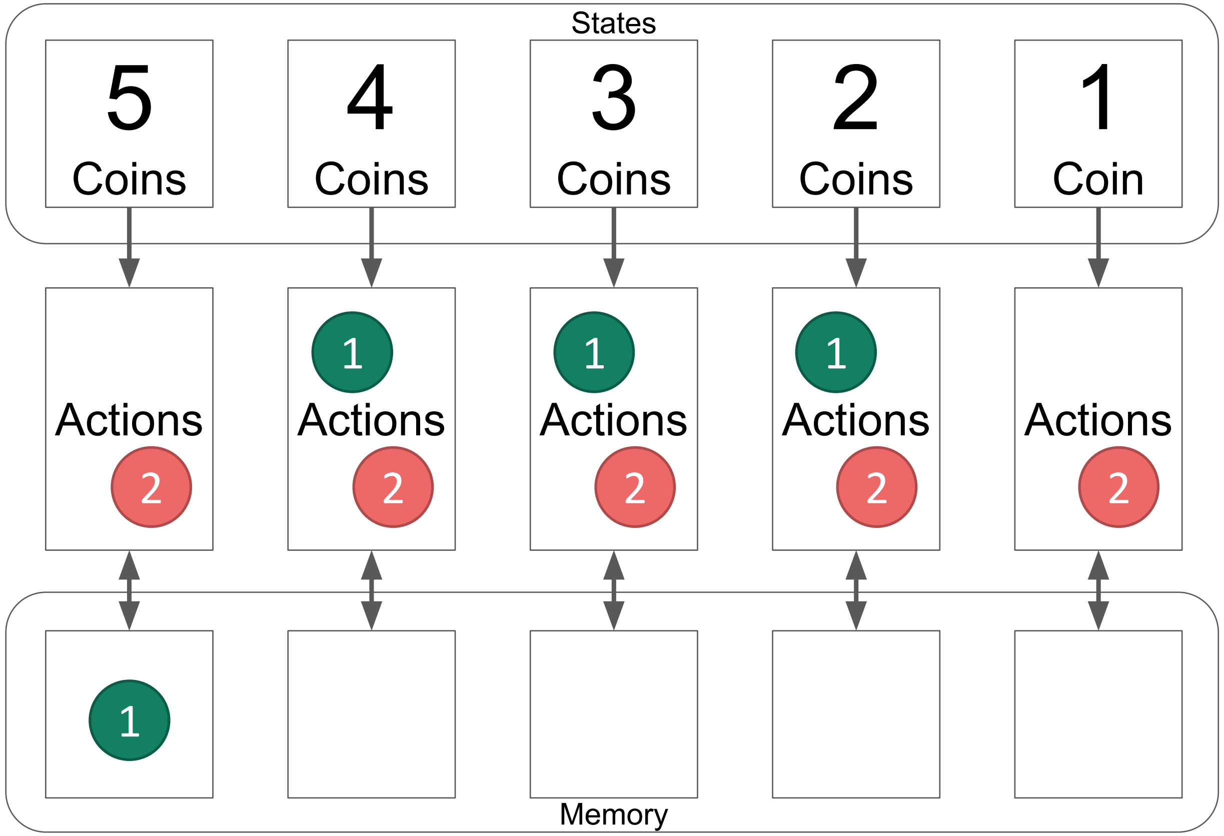
States

5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
Actions 2	1 Actions 2	1 Actions 2	1 Actions 2	Actions 2
1				

Memory

1. The game starts again with five coins, this time **Bob** begins
2. There are **five coins** left, therefore **Bob** takes a **random action stone** from **state 5** and puts it into **memory**

Game 2



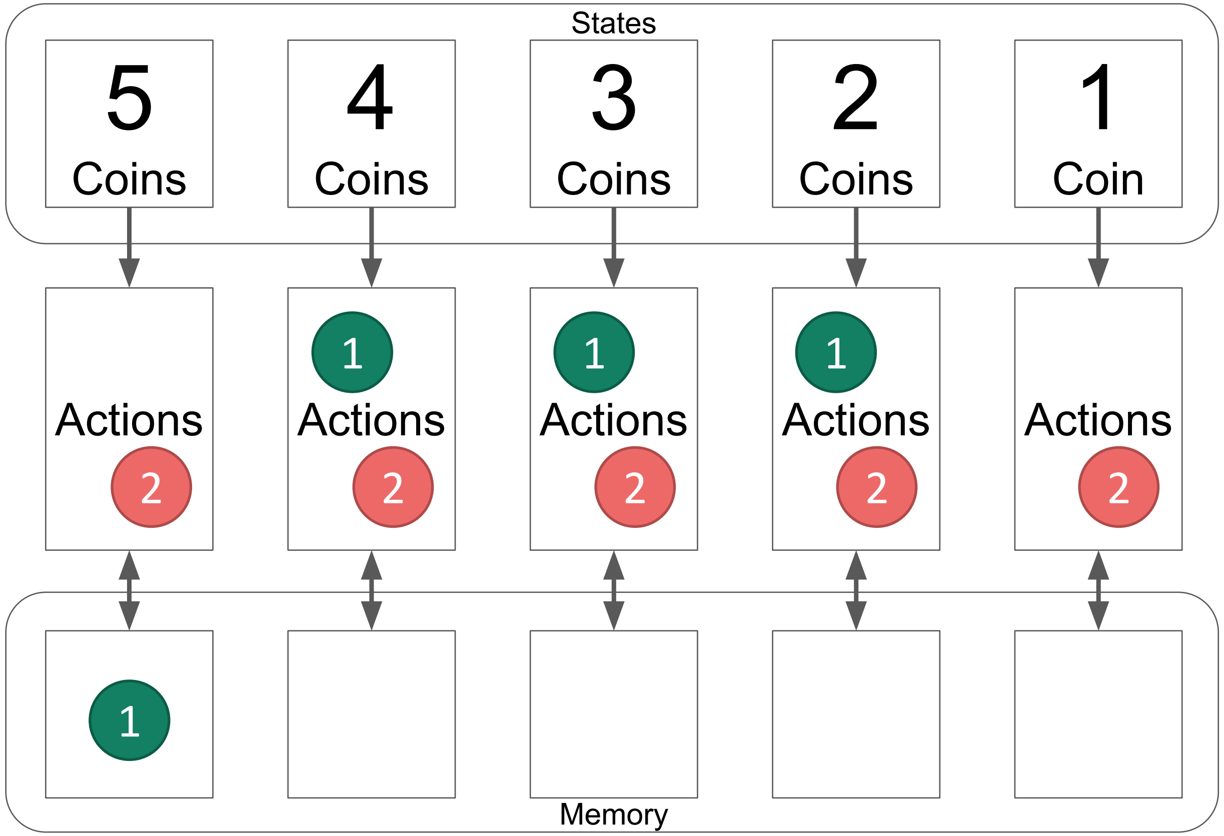
States

5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
Actions 2	1 Actions 2	1 Actions 2	1 Actions 2	Actions 2
1				

Memory

1. The game starts again with five coins, this time **Bob** begins
2. There are **five coins** left, therefore **Bob** takes a **random action stone** from **state 5** and puts it into **memory**
3. **Bob** then performs the action and **takes one coin**

Game 2

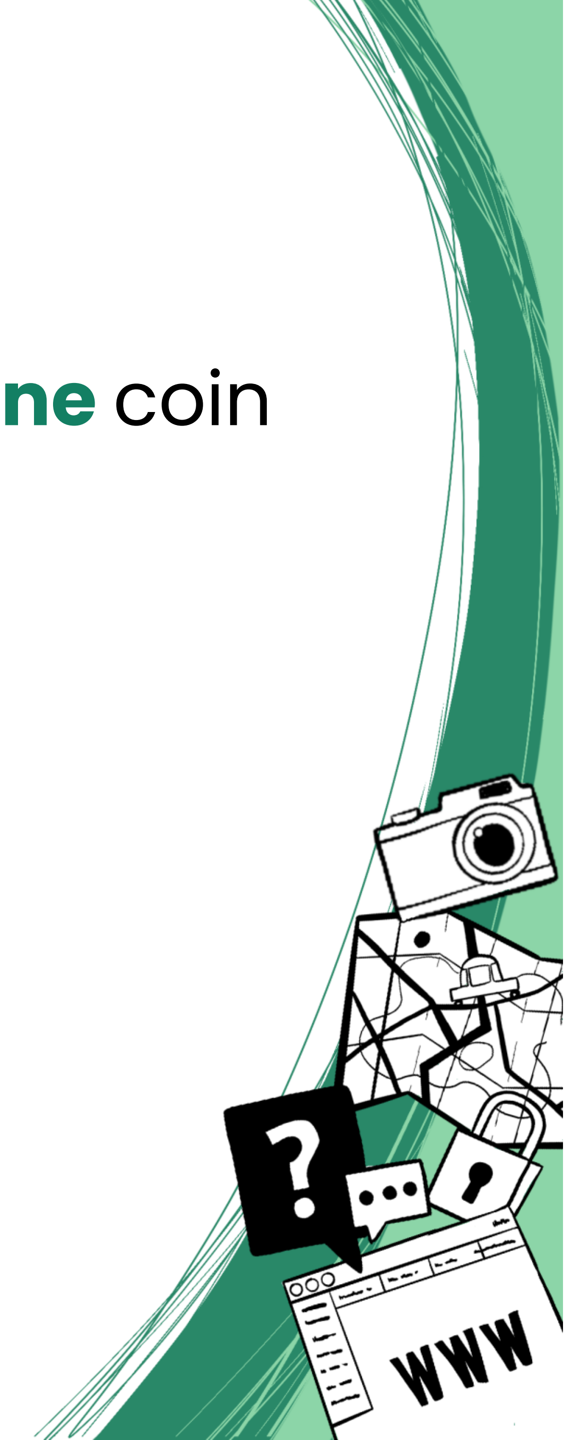


States

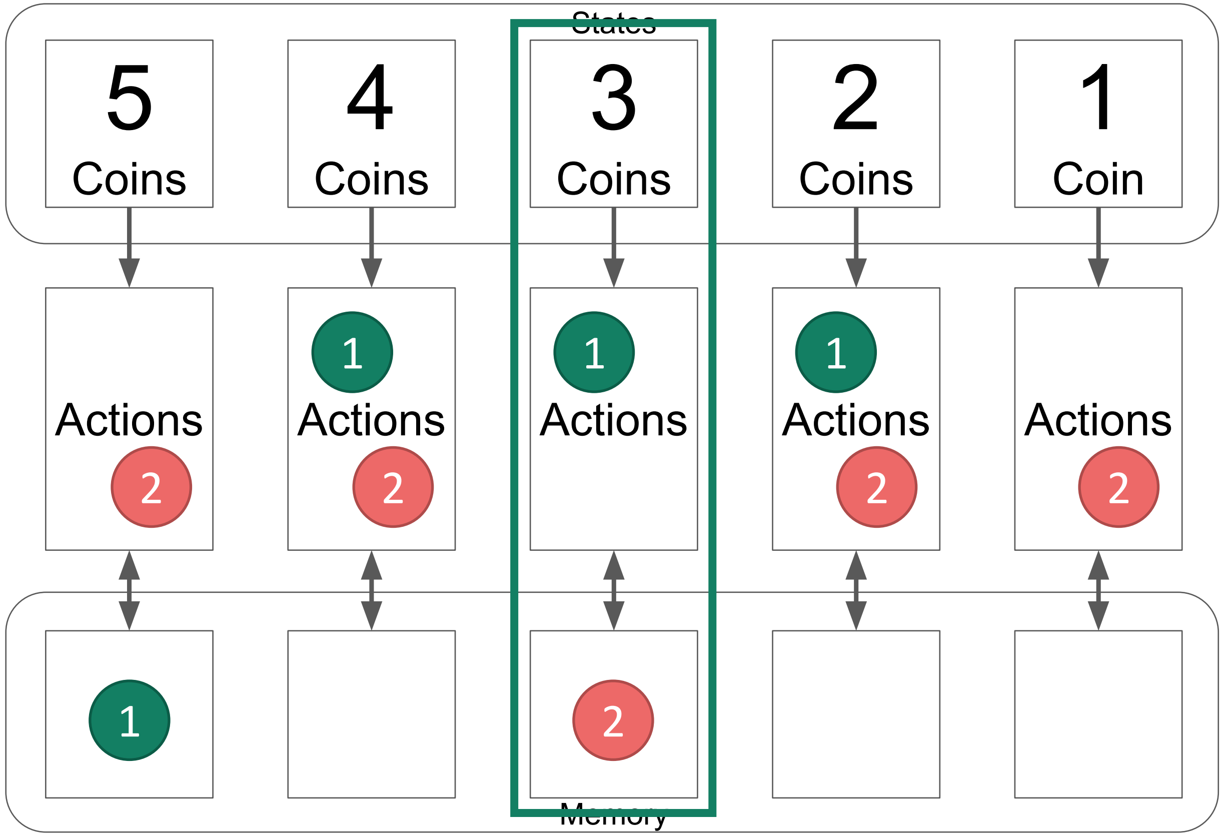
5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
Actions 2	1 Actions 2	1 Actions 2	1 Actions 2	Actions 2
1				

Memory

4. **Alice takes one coin**



Game 2



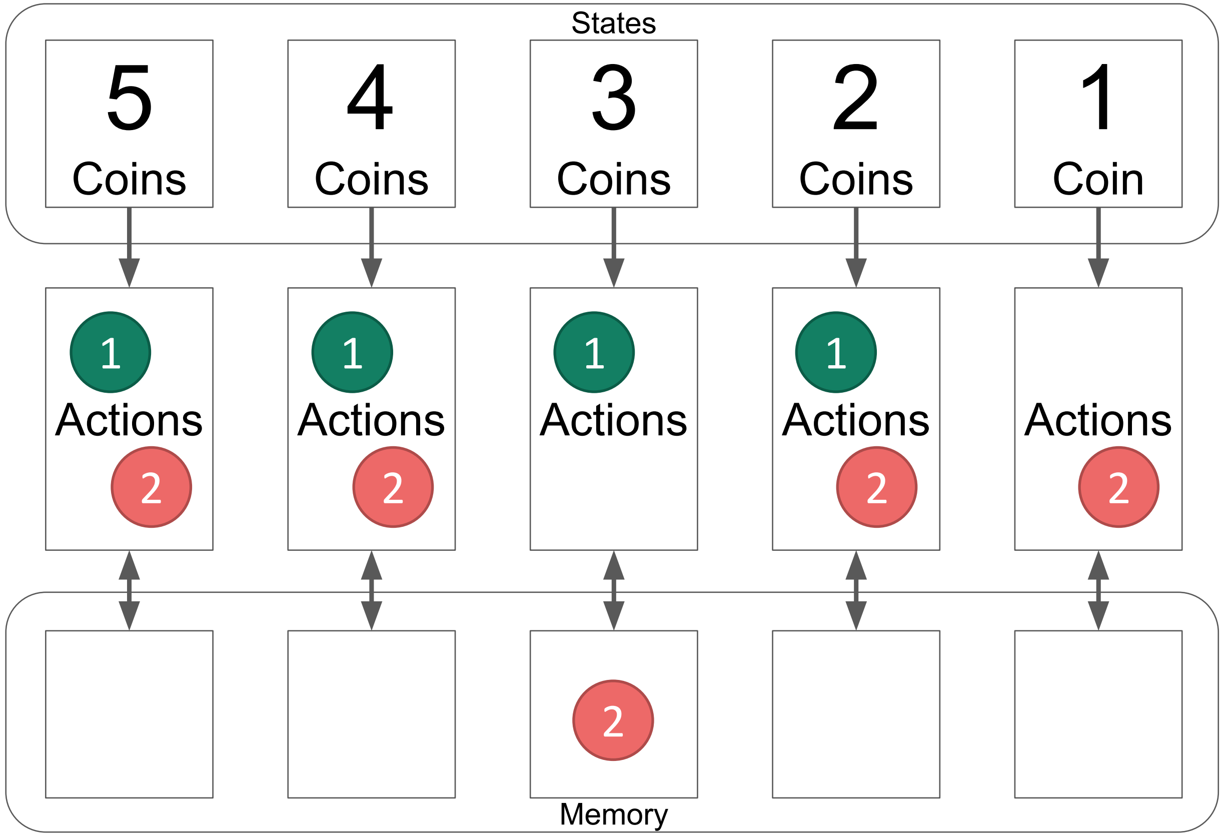
States

5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
Actions 2	1 Actions 2	1 Actions	1 Actions 2	Actions 2
1		2		

Memory

4. **Alice takes one coin**
5. There are **three coins** left, therefore **Bob** takes a **random action stone** from **state 3** and puts it into **memory**

Game 2

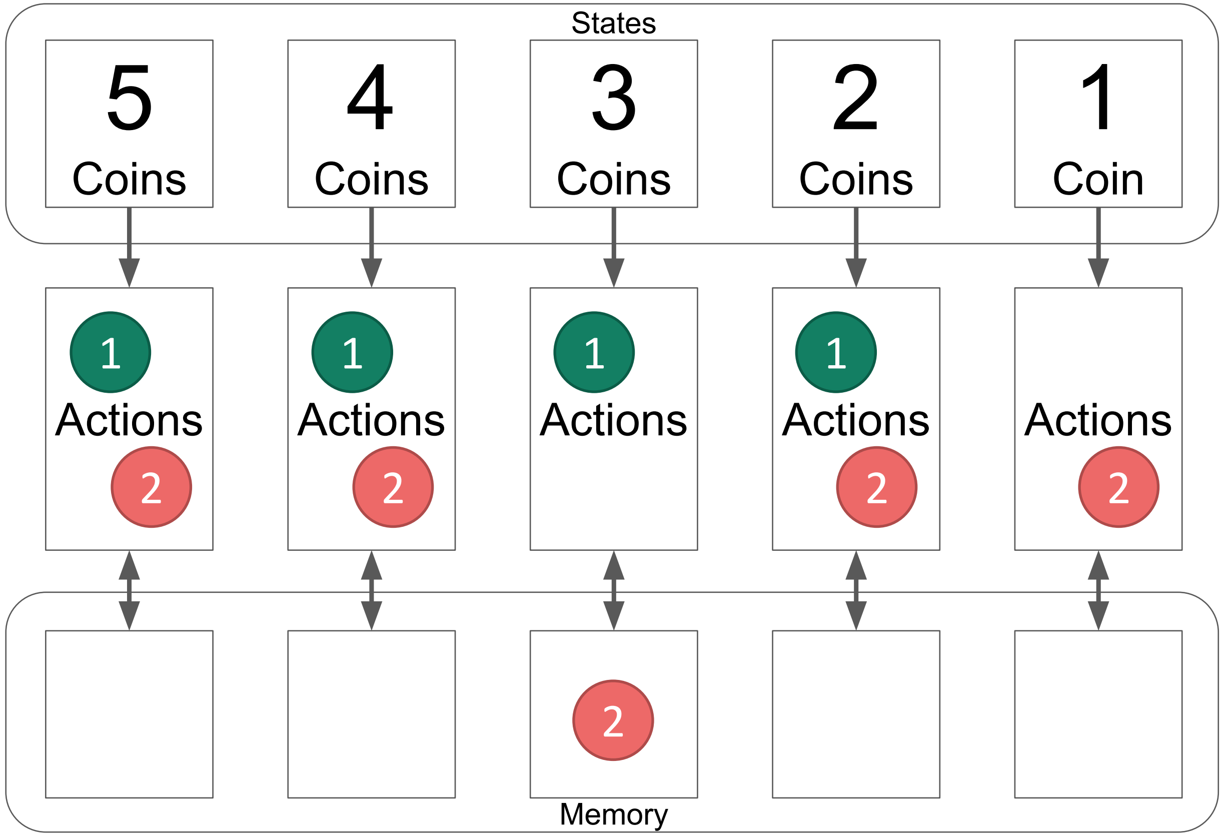


States

	5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
Actions	1 2	1 2	1	1 2	2
Memory			2		

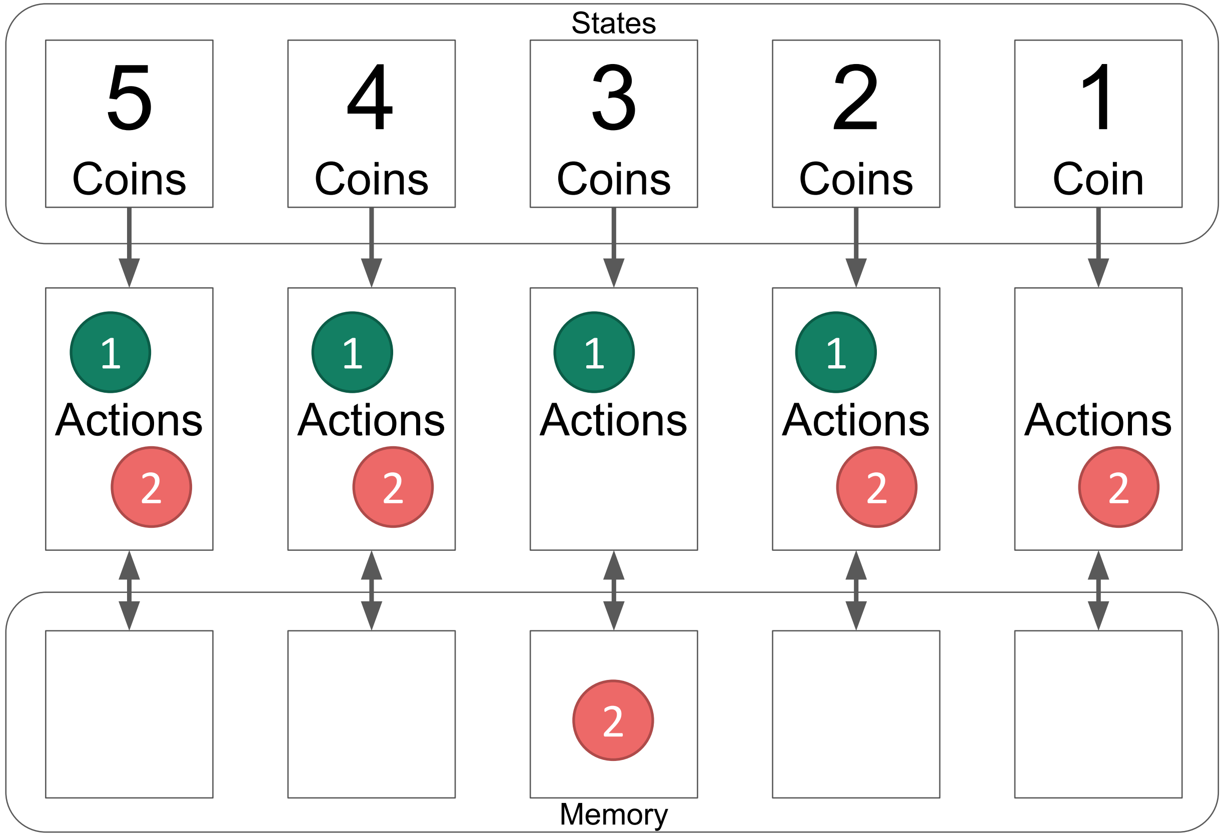
4. **Alice** **takes one** coin
5. There are **three coins** left, therefore **Bob** takes a **random action stone** from **state 3** and puts it into **memory**
6. **Bob** also returns the **stone** which was **previously in memory** back to its actions field

Game 2



7. **Bob** then performs the action and **takes two** coins

Game 2

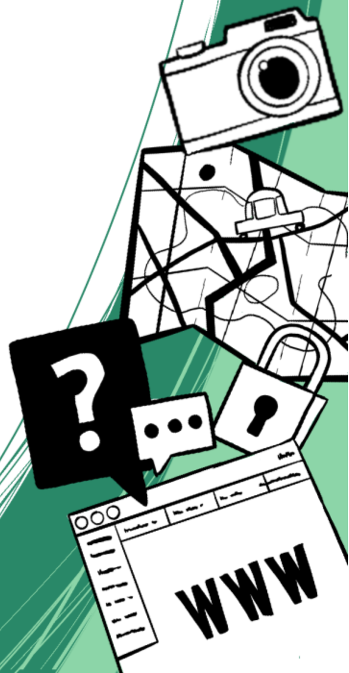


States

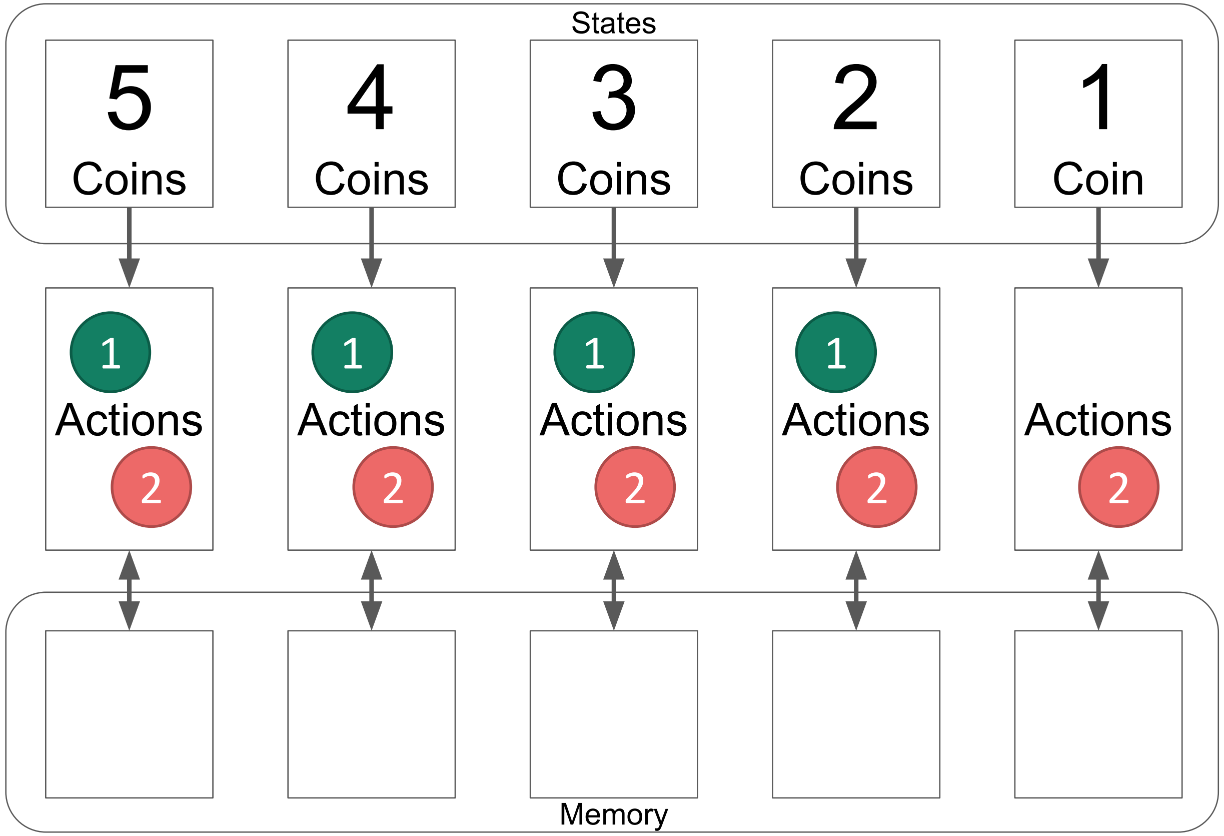
5 Coins	4 Coins	3 Coins	2 Coins	1 Coin
1 Actions 2	1 Actions 2	1 Actions	1 Actions 2	Actions 2
		2		

Memory

7. **Bob** then performs the action and **takes two** coins
8. **Alice takes one** coin



Game 2



States

5 Coins | 4 Coins | 3 Coins | 2 Coins | 1 Coin

1 Actions 2 | 1 Actions 2 | 1 Actions 2 | 1 Actions 2 | Actions 2

Memory

7. **Bob** then performs the action and **takes two** coins
8. **Alice takes one** coin
9. **Bob** has **won**, therefore he **returns** the **stone in memory** back to its original location

Game 3…

- Continue with more games, until your AI plays perfectly!

Special Rules

- If an **action** is **not possible** (e.g. take two when there is only one stone), **remove** the **action stone** and **take another one**

Special Rules

- If an **action** is **not possible** (e.g. take two when there is only one stone), **remove** the **action stone** and **take another one**
- If there are **no action stones** for the state, **remove** the **action stone in memory** from the game and continue with a **random action** (take one or two coins)