**PAC-MAN**

**THEORY OF COMPUTATION**    
**CASE STUDY – TERM 3**    
**GROUP-2**

|  |  |
| --- | --- |
| **NAME** | **ROLL NUMBER** |
| Harini S | CB.EN.U4CSE19023 |
| Rithika Sri J | CB.EN.U4CSE19025 |
| Tarsana B | CB.EN.U4CSE19057 |

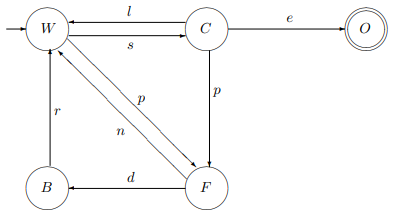
**REVISITING LAST TERM:**

Pacman is a game board with a player-character moves inside a maze having full of pellets. Objects in Pac-man game are Pac-man and ghosts.   
The DFA was modelled based on the Pac-man ghost’s behaviour. The ghosts had four behaviours:     
1. Randomly wander the maze     
2. Chase Pac-Man, when he is within line of sight    
3. Flee Pac-Man, after Pac-Man has consumed a power pellet    
4. Return to the central base to regenerate

This model does not need any memory and there is no need to store nor count the alphabets in the input string.

W --> “wander the maze”; C --> “chase Pac-Man”; F --> “Flee Pac-Man”; B --> “return to base”; O --> “game over”.

 s --> “spots Pac-Man”; p --> “Pac-Man eats power pellet; l --> “loses Pac-Man”; e --> “eats Pac-Man”; n --> “power pellet expires, Pac-Man returns to normal”; d --> “eaten by Pac-Man and dies”; r --> “regenerate from the central base.”



**LIMITATIONS:**    
This model does not predict whether or not the player has already won the game because the count of pellets is not stored. So, when the Pac-man has eaten all the pills the game gets over but this model does not recognise it.     
    
**There is a need to upgrade this model!!**    
Pushdown Automata(PDA) is chosen in the development of Pacman because the system could memorize all pellets that are available to eat or have been eaten by the Pac-man.

**PROBLEM STATEMENT:**

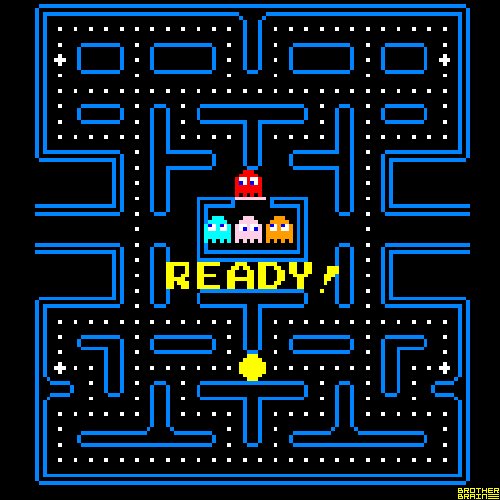
Pacman is a game board with a player-character moves inside a maze having full of pellets. Objects in Pac-man game are Pac-man, pellets and ghosts.

**Pac-man’s mission:**

In order to complete the mission, Pac-man must eat all the pellets and be aware of the ghosts. If Pac-man eat super-pellets, it will become invincible and can eat ghosts. So, the ghosts start avoiding Pac-man till the power exhausts.

**When does this game end? (FINAL STATE)**   
(1) Pac-man is eaten up by the any one of the ghosts (already visited in FA)

(2) Pac-man consumes all the pellets à Upgraded feature in the **PDA**



**UPGRADING TO PDA:**   
PDA for Pacman from the point of view of Pac-Man’s state and activity,

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STATES** | | **ALPHABETS** | | **STACK SYMBOLS** | |
| I | Food loaded in maze | f | Load pellets into the maze | f | Pellet |
| N | Normal Pac-man | m | Pac-man starts to move | z | Stack initial symbol |
| S | Super Pac-man | c | Consumes pellet in the maze |  |  |
| O | Game over | p | Eat Power-up pellets |  |  |
| W | Pac-man won | n | Power-up time exhausts |  |  |
|  |  | e | Eaten up by the ghosts |  |  |
|  |  |  |  |  |  |

**CONTEXT FREE LANGUAGE:**

L = {fxm(cap cb n)y st: x = ax+by} U {fxm(cap cb n)ye st: x > ax+by }

**TRANSITION FUNCTION:**

M= (Q,∑, Γ, δ  ,Q0,Z,F)

Q= {I,N,S,O,W}

∑= {f, m, c, p, n, e}

Γ= {f, z}

Q0= I

F= {W,O}

δ=

δ (I, f, z)  = (I, zf)

δ (I, f, f)   = (I, ff)

δ (I, m, f) = (N, f)

δ (N, c, f) = (N, ε)

δ (N, p, f) = (S, f)

δ (S, n, f) = (N, f)

δ (S, c, f) = (S, ε)

δ (S, ε, z) = (W, ε)

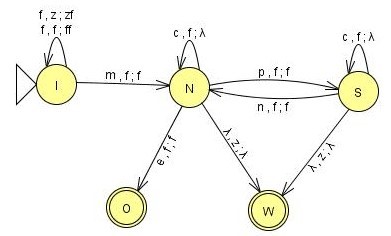
δ (N, ε, z) = (W, ε)

δ (N, e, f) = (O, f)

**TRANSITION TABLE:**

|  |  |  |  |
| --- | --- | --- | --- |
| **STATE** | **INPUT** | **STACK SYMBOL** | **MOVES** |
| I | f | z | (I, fz) |
| I | f | f | (I, ff) |
| I | m | f | (N, f) |
| N | c | f | (N, λ) |
| N | p | f | (S, f) |
| N | e | f | (O, f) |
| N | λ | z | (W, λ) |
| S | c | f | (S, λ) |
| S | n | f | (N, f) |
| S | λ | z | (W, λ) |

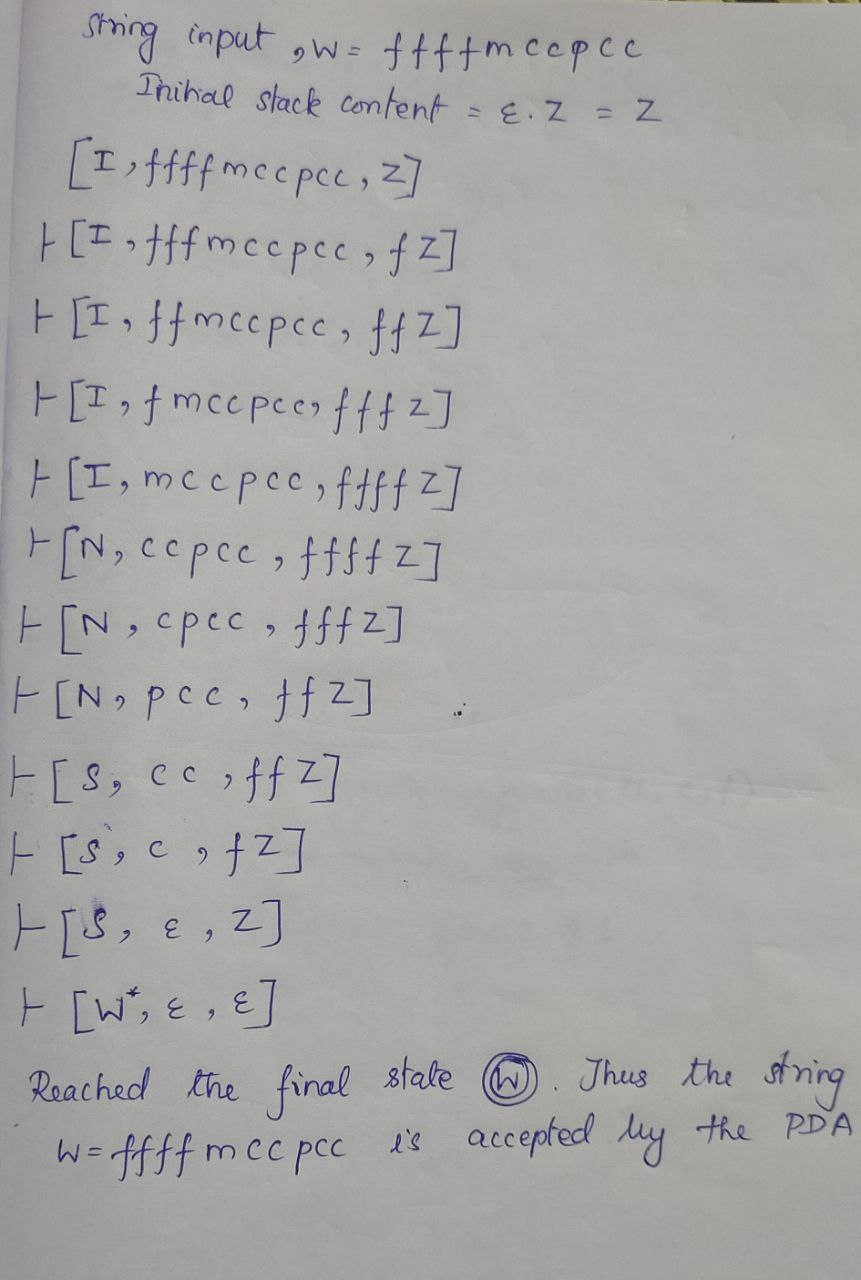
**PUSH DOWN AUTOMATA:**



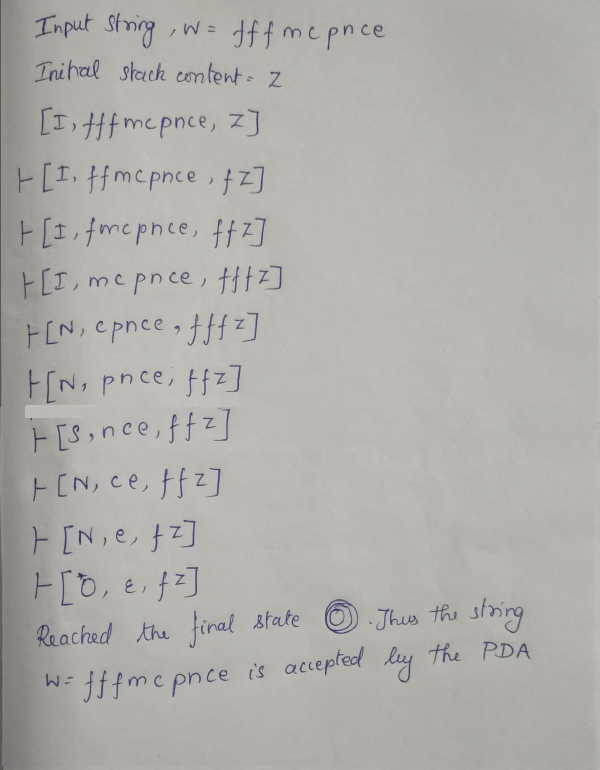
fz

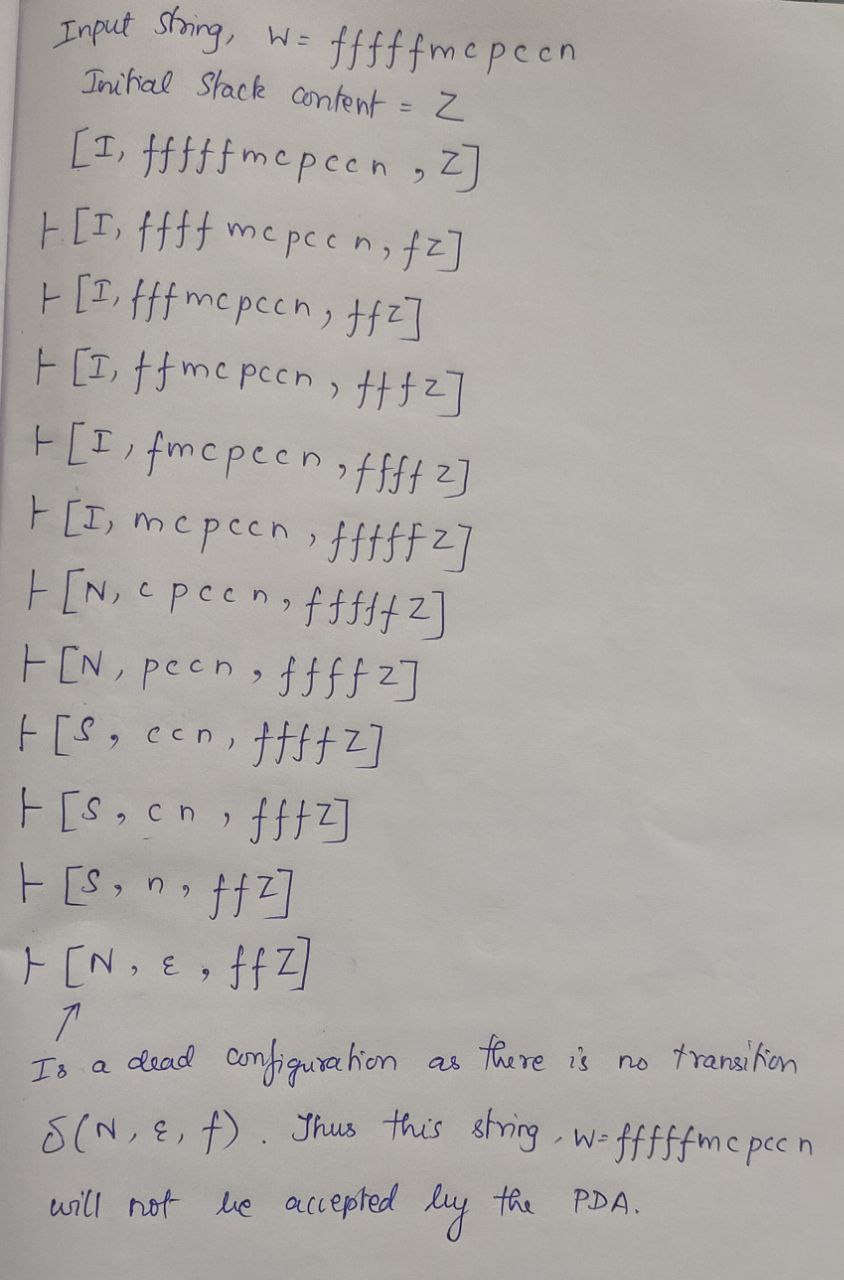
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STATES** | | **ALPHABETS** | | **STACK SYMBOLS** | |
| I | Food loaded in maze | f | Load pellets into the maze | f | Pellet |
| N | Normal Pac-man | m | Pac-man starts to move | z | Stack initial symbol |
| S | Super Pac-man | c | Consumes pellet in the maze |  |  |
| O | Game over | p | Eat Power-up pellets |  |  |
| W | Pac-man won | n | Power-up time exhausts |  |  |
|  |  | e | Eaten up by the ghosts |  |  |
|  |  |  |  |  |  |

**INSTANTANEOUS DESCRIPTION**:

STRING- 1

STRING-2



STRING - 3

**LIMITATIONS:**    
This model does not give another life(chance) to the Pac-man when it is eaten up by the ghosts because the count of life(heart) is not stored. So, when the Pac-man needs another life to finish the game, this model does not support it.

**There is a need to upgrade this model!!**    
2 Stack Pushdown Automata(PDA) is chosen in the development of Pacman because the system could memorize remaining pellets and Pac-man’s lives.

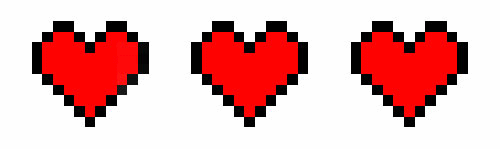
**PROBLEM STATEMENT:**

Pacman is a game board with a player-character moves inside a maze having full of pellets. Objects in Pac-man game are Pac-man, pellets, lives and ghosts.

**When does this game end? (FINAL STATE)**

(1) Pac-man consumes all the pellets.

(2) Pac-man is eaten up by the any one of the ghosts and  no more life      
remaining for Pac-man à upgraded feature in the **2 STACK PDA**.



**UPGRADING TO 2 STACK PDA:**

A two-stack pushdown automaton (2-stack PDA) is exactly like a pushdown automaton except that it has two stacks from which we can push and pop at each step.

2 STACK PDA for Pacman from the point of view of Pac-Man’s state and activity,

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STATES** | | **ALPHABETS** | | **STACK SYMBOLS** | |
| I | Food loaded in maze | f | Load pellets into the maze | Stack 1 | |
| N | Normal Pac-man | m | Pac-man starts to move | f | pellet |
| S | Super Pac-man | c | Consumes pellet in the maze | z1 | stack1 initial symbol |
| L | Lost | p | Eat Power-up pellets | Stack 2 | |
| O | Game over | n | Power-up time exhausts | h | heart |
| W | Pac-man won | e | Eaten up by the ghosts | z2 | Stack2 initial symbol |
|  |  | r | reincarnate |  |  |

**TRANSITION FUNCTION:**

A 2-stack PDA consists of a 6-tuple (Q, Σ, Γ, δ, q0, F) where the transition function is defined as δ : Q × Σ × Γ × Γ → Q × Γ1 × Γ2.

M=(Q,∑, Γ, δ  ,Q0,Z,F)

Q= {I, N, S, L, W, O}

∑= {f, m, c, p, n, e, r}

Γ= {f, z1, h, z2}

Q0= I

F= {W,O}

δ : Q × Σ × Γ1 × Γ2 → P(Q × Γ1 × Γ2)

δ =

(I, f, z1, z2) → (I, fz1, z2)

(I, f, z1, h) → (I, fz1, h)

(I, f, f, z2) → (I, ff, z2)

(I, f, f, h) à (I, ff, h)

(I, h, z1, z2) → (I, z1, hz2)

(I, h, z1, h) → (I, z1, hh)

(I, h, f, z2) → (I, f, hz2)

(I, h, f, h) → (I, f, hh)

(I, m, f, h) → (N, f, λ)

(N, c, f, h) → (N, λ, h)

(N, c, f, z2) → (N, λ, z2)

(N, p, f, h) → (S, f, h)

(N, p, f, z2) → (S, f, z2)

(N, e, f, h) → (L, f, h)

(N, e, f, z2) → (L, f, z2)

(N, λ, z1, h) → (W, λ, h)

(N, λ, z1, z2) → (W, λ, z2)

(S, c, f, h) → (S, λ, h)

(S, c, f, z2) → (S, λ, z2)

(S, n, f, h) → (N, f, h)

(S, n, f, z2) → (N, f, z2)

(S, λ, z1, h) → (W, λ, h)

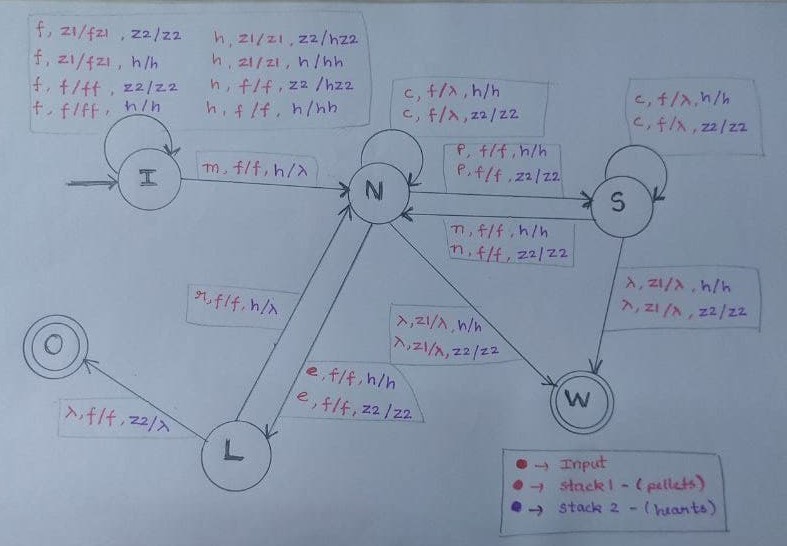
(S, λ, z1, z2) → (W, λ, z2)

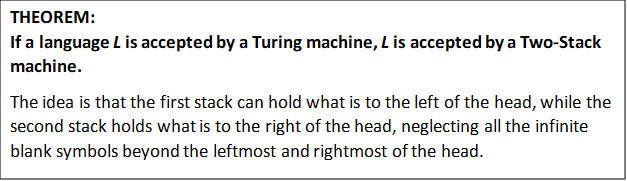
(L, r, f, h) → (N, f, λ)

(L, λ, f, z2) → (O, f, λ)

**2 STACK PUSH DOWN AUTOMATA:**

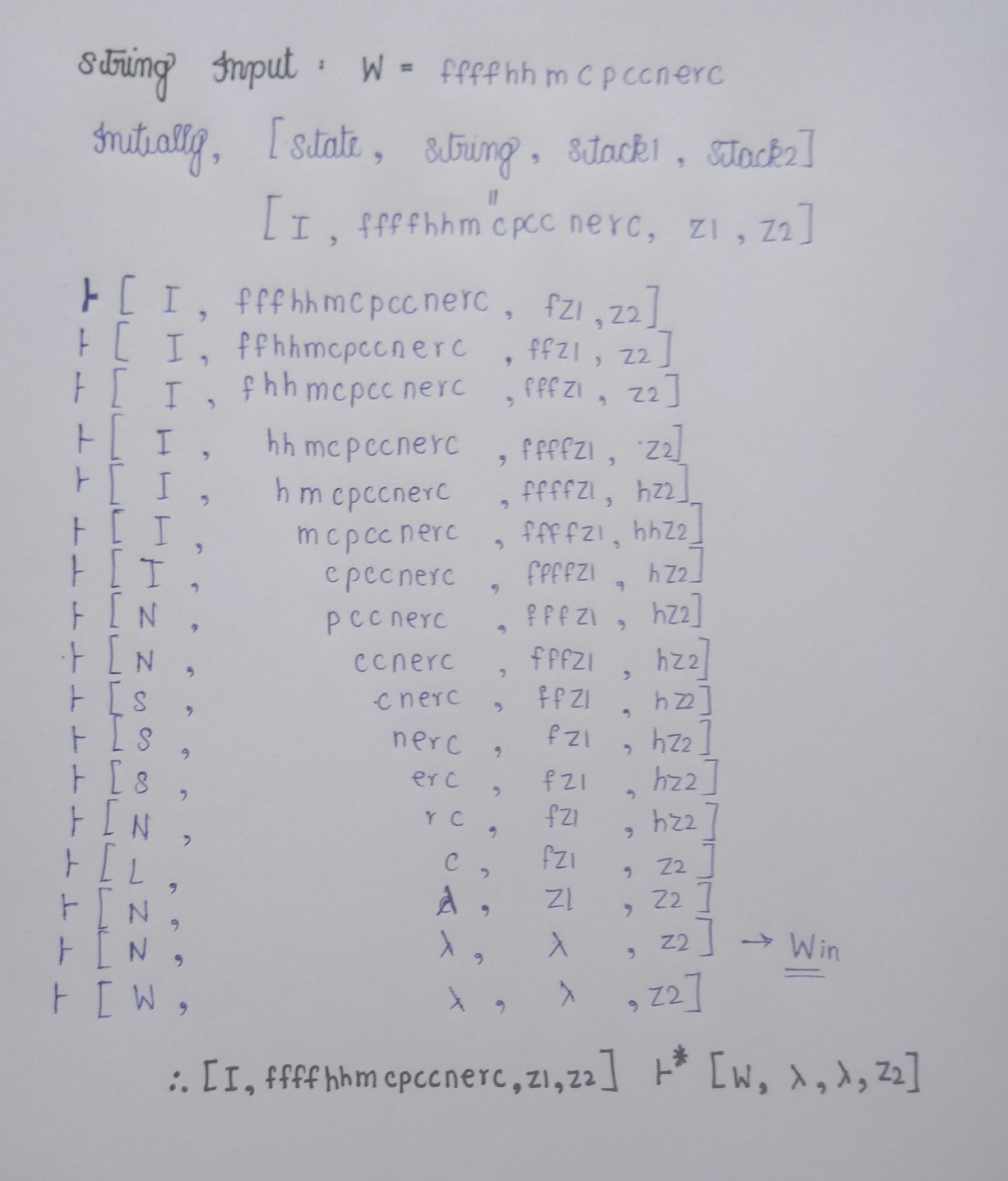
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STATES** | | **ALPHABETS** | | **STACK SYMBOLS** | |
| I | Food loaded in maze | f | Load pellets into the maze | Stack 1 | |
| N | Normal Pac-man | m | Pac-man starts to move | f | pellet |
| S | Super Pac-man | c | Consumes pellet in the maze | z1 | stack1 initial symbol |
| L | Lost | p | Eat Power-up pellets | Stack 2 | |
| O | Game over | n | Power-up time exhausts | h | heart |
| W | Pac-man won | e | Eaten up by the ghosts | z2 | Stack2 initial symbol |
|  |  | r | reincarnate |  |  |



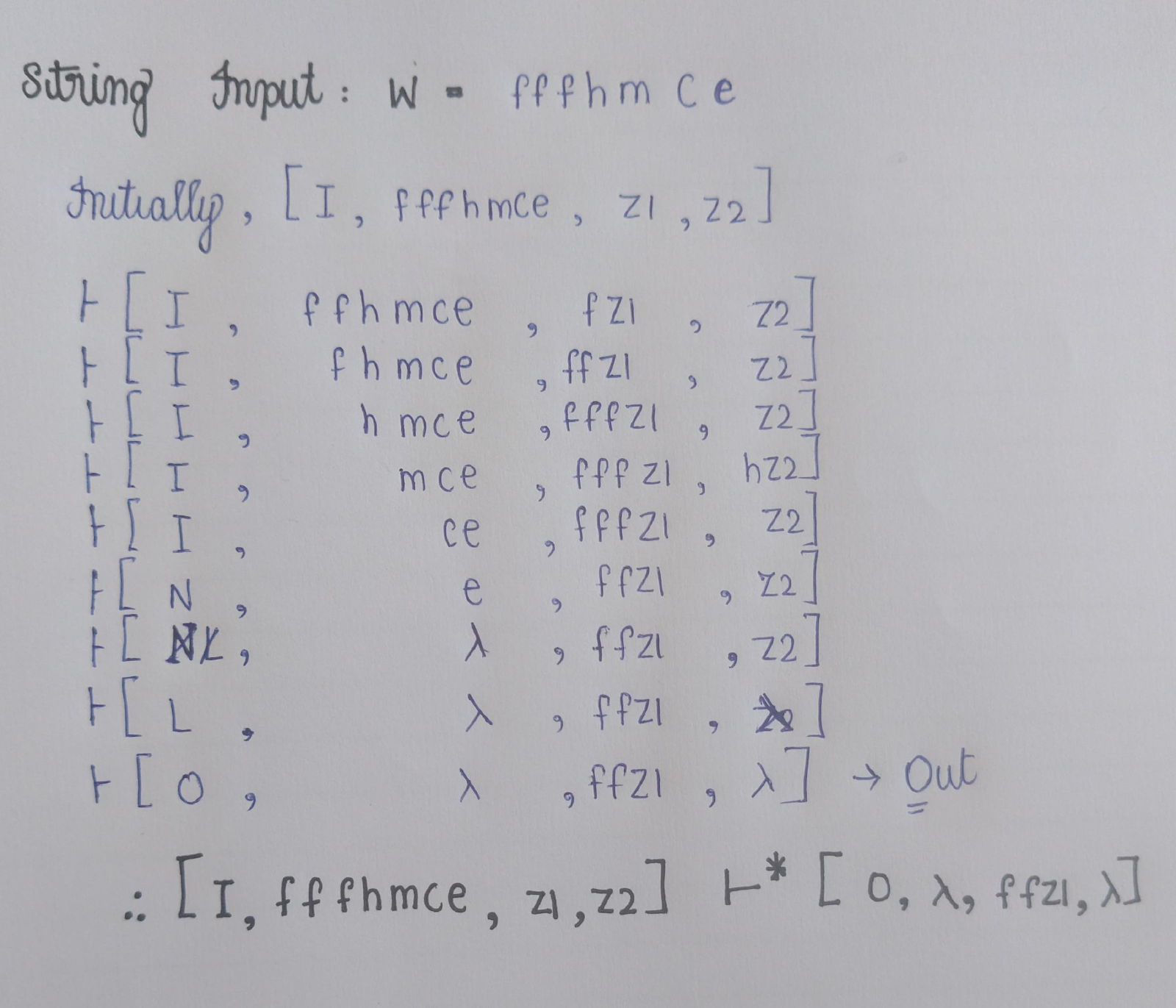


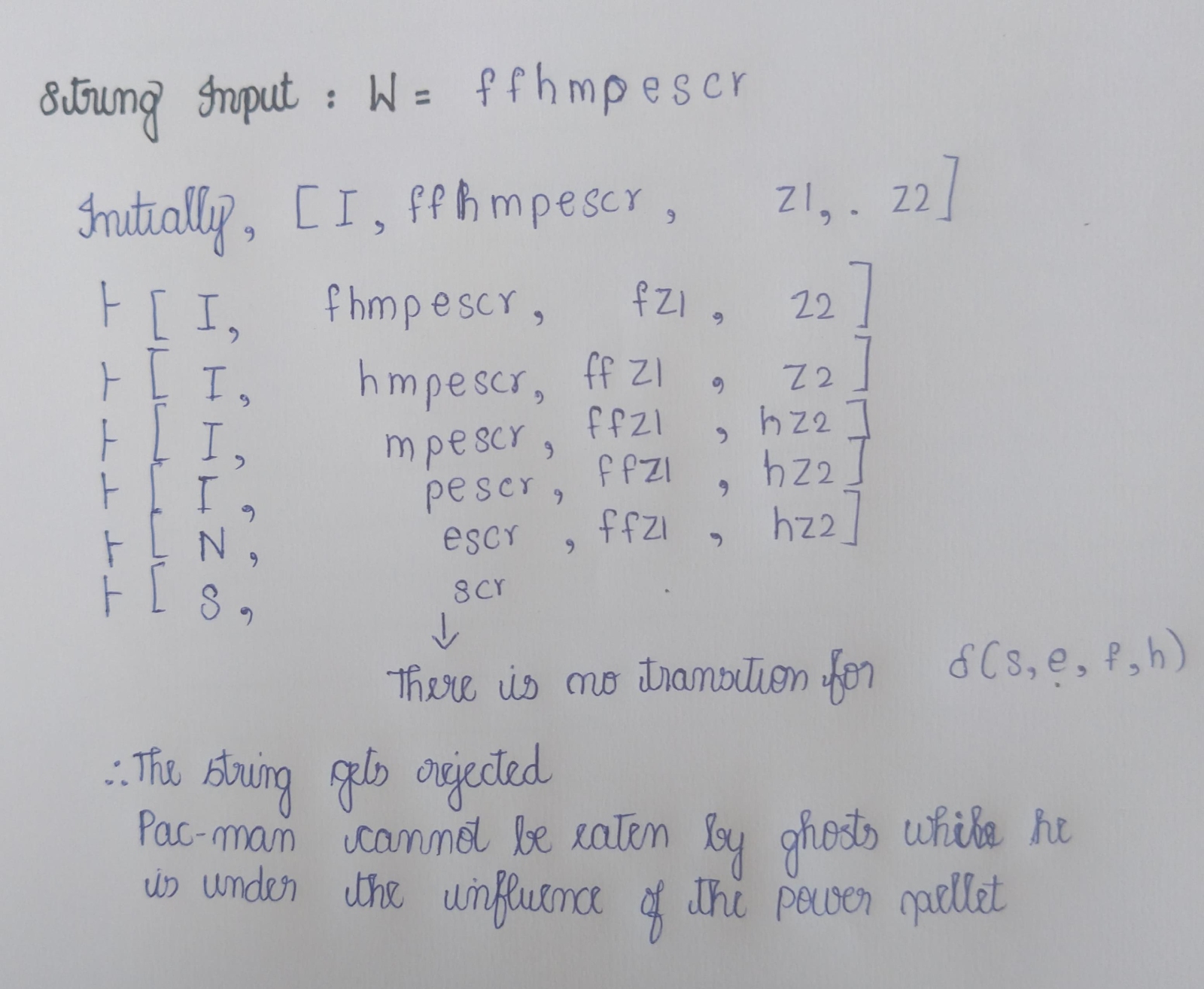
**INSTANTANEOUS DESCRIPTION**:

Accepting String 1:



Accepting String 2:



Rejecting String: