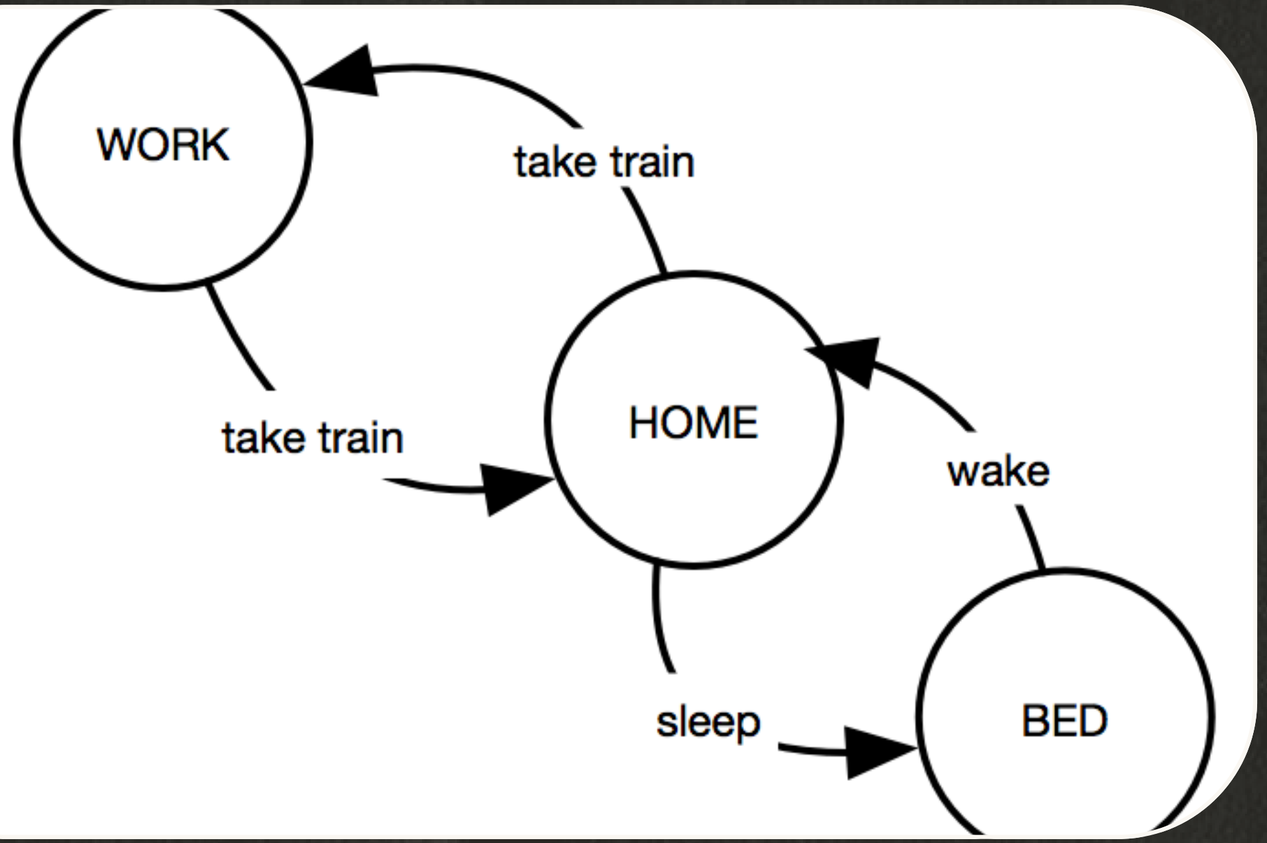
SOFTWARE TESTING COURSE
PROJECT

FSM GENERATOR

GROUP 4 - KOUSHAL PARUPUDI
SAMIR MAHMOOD

FINITE STATE MACHINES

SOFTWARE TESTING COURSE PROJECT



FSMs or Finite state machines are used to describe the behavior of a system in a finite number of states.

FSM's are made up of the following:

States: Different configurations of the system

Transitions: Describe how a system changes to another state. Depends on guard and triggers

Inputs: Cause a system to transition to another state

Outputs: Behavior of a system as it transitions to another state

PROBLEM DEFINITION

Objective: Develop a tool that can generate a large number of random FSMs following specified parameters.

Input: The following parameters are specified through the command line:

- Number of FSMs generated
- Number of states per FSM
- Number of possible inputs
- Number of possible outputs
- Whether FSM is initially connected
- Whether FSM is complete

Output: Text file containing the generated FSMs. FSM name on line 1 followed by a list of transitions, one on each line. Each transition is in the format Current_State, Input, Next_State, Output. FSMs are separated by a single newline.

PROBLEM DEFINITION

Process:

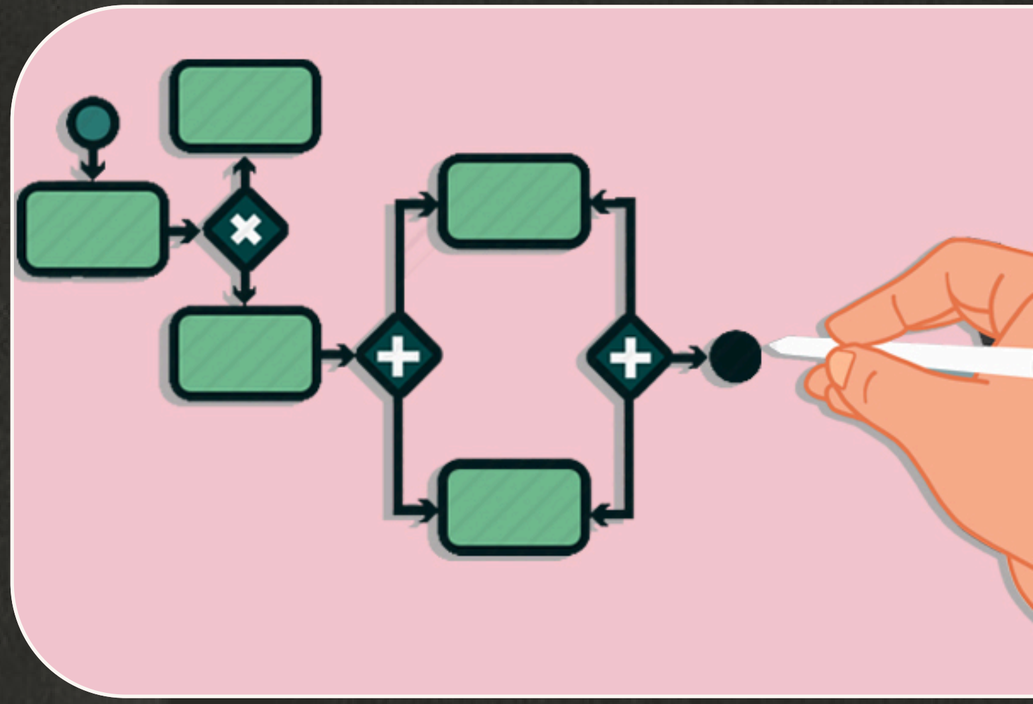
FSM model generation: Generating FSMs based on requirements from the user (User input).

Initially or fully connected (or both):

Checks FSM whether it is either initially connected or not, then discards the FSM if not and generates a new FSM that is initially connected. Also generates complete FSM based on the user input by ensuring that the initial state is connected to every state in the FSM model.

Result Mapping and Output:

Output of the generated FSM is displayed on the console along with a generated text file of the user defined number of FSMs



RESULTS


```
Enter the number of FSMs to be generated: 1
Enter the number of states: 3
Enter the number of inputs: 2
Enter the number of outputs: 1
Is the FSM initially connected? (y/n): y
Is the FSM complete? (y/n): y

fsm0
State 0
  Input 0 → Next State: 2 , Output: 0
  Input 1 → Next State: 2 , Output: 0
State 1
  Input 0 → Next State: 1 , Output: 0
  Input 1 → Next State: 1 , Output: 0
State 2
  Input 0 → Next State: 2 , Output: 0
  Input 1 → Next State: 1 , Output: 0
```

Implementation Success: Successfully integrated the requirements of user defined FSM generation and text file output.

Algorithm Efficiency: Demonstrated high efficiency, effectively generating FSMs with numerous states and transitions while adhering to the user requirements.

Output formatting: Output on console is formatted efficiently with arrows indicating transitions. Output text file is also error free

RESULTS


```
0 5 9 4 3 void-data
0 4 b 1
0 4 c 0
1 4 c 0
2 2 b 0
2 3 d 0
3 1 c 0
4 3 a 2
4 2 c 1
4 2 d 0
1 5 7 4 3 void-data
0 2 d 0
1 3 b 2
2 1 b 0
2 4 c 2
3 3 c 1
4 3 c 2
4 4 d 0
```

[INITIAL STATE, ENDING STATE, INPUT, OUTPUT]

EVALUATION OF OUTPUT

Randomness: Evaluate the randomness of the generated FSMs. This involves analyzing statistical properties such as distribution of states, transitions, and other characteristics to ensure that the generator produces diverse and unpredictable FSMs.