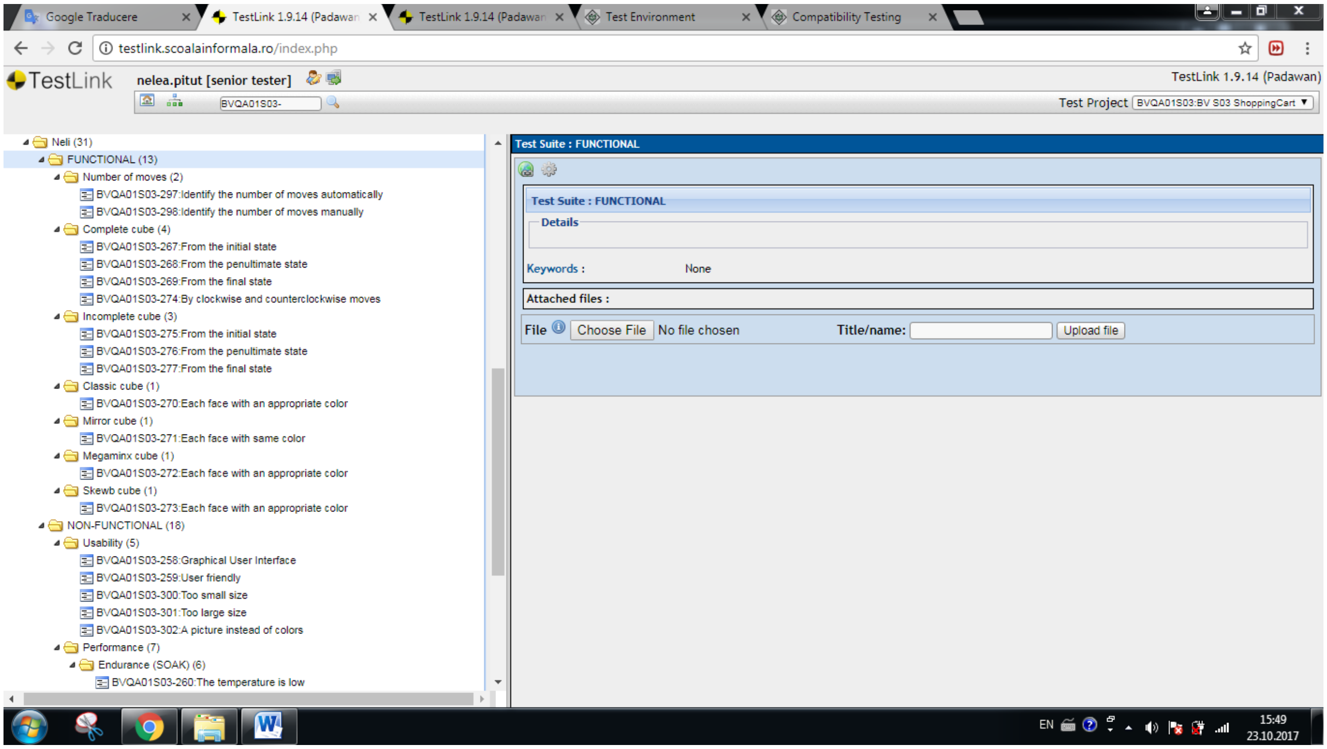
CATEGORY	SUBCATEGORY	NAME
FUNCTIONAL		
NUMBER OF MOVES		1. Identify the number of moves automatically
		2. Identify the number of moves manually
COMPLETE CUBE		3. From the initial state
		4. From the penultimate state
		5. From the final state
		6. By clockwise and counterclockwise moves
INCOMPLETE CUBE		7. From the initial state
		8. From the penultimate state
		9. From the final state
CLASSIC CUBE		10. Each face with an appropriate color
MIRROR CUBE		11. Each face with same color
MEGAMINX CUBE		12. Each face with an appropriate color
SKEWB CUBE		13. Each face with an appropriate color
NON-FUNCTIONAL		
USABILITY		14. Graphical User Interface
		15. User friendly
		16. Too small size
		17. Too large size
		18. A picture instead of colors
PERFORMANCE	ENDURANCE (SOAK)	19. The temperature is low
		20. The temperature is high
		21. The room is full with dust
		22. The cube is used after a falling into the sand
		23. The cube is used after the falling in water
		24. The cube is used after fall
	STRESS TESTING	25. The cube after long uninterrupted use
COMPATIBILITY		26. Operating systems
		27. Different devices
		28. Different programs
ENVIRONMENT		29. Dependent of internet connection
		30. Independent of internet connection
		31. Program not working after internet off



- Neli (31)
  - FUNCTIONAL (13)
    - Number of moves (2)
      - BVQA01S03-297:Identify the number of moves automatically
      - BVQA01S03-298:Identify the number of moves manually
    - Complete cube (4)
      - BVQA01S03-267:From the initial state
      - BVQA01S03-268:From the penultimate state
      - BVQA01S03-269:From the final state
      - BVQA01S03-274:By clockwise and counterclockwise moves
    - Incomplete cube (3)
      - BVQA01S03-275:From the initial state
      - BVQA01S03-276:From the penultimate state
      - BVQA01S03-277:From the final state
    - Classic cube (1)
      - BVQA01S03-270:Each face with an appropriate color
    - Mirror cube (1)
      - BVQA01S03-271:Each face with same color
    - Megaminx cube (1)
      - BVQA01S03-272:Each face with an appropriate color
    - Skewb cube (1)
      - BVQA01S03-273:Each face with an appropriate color
  - NON-FUNCTIONAL (18)
    - Usability (5)
      - BVQA01S03-258:Graphical User Interface
      - BVQA01S03-259:User friendly
      - BVQA01S03-300:Too small size
      - BVQA01S03-301:Too large size
      - BVQA01S03-302:A picture instead of colors
    - Performance (7)
      - Endurance (SOAK) (6)
        - BVQA01S03-260:The temperature is low

Test Suite : FUNCTIONAL

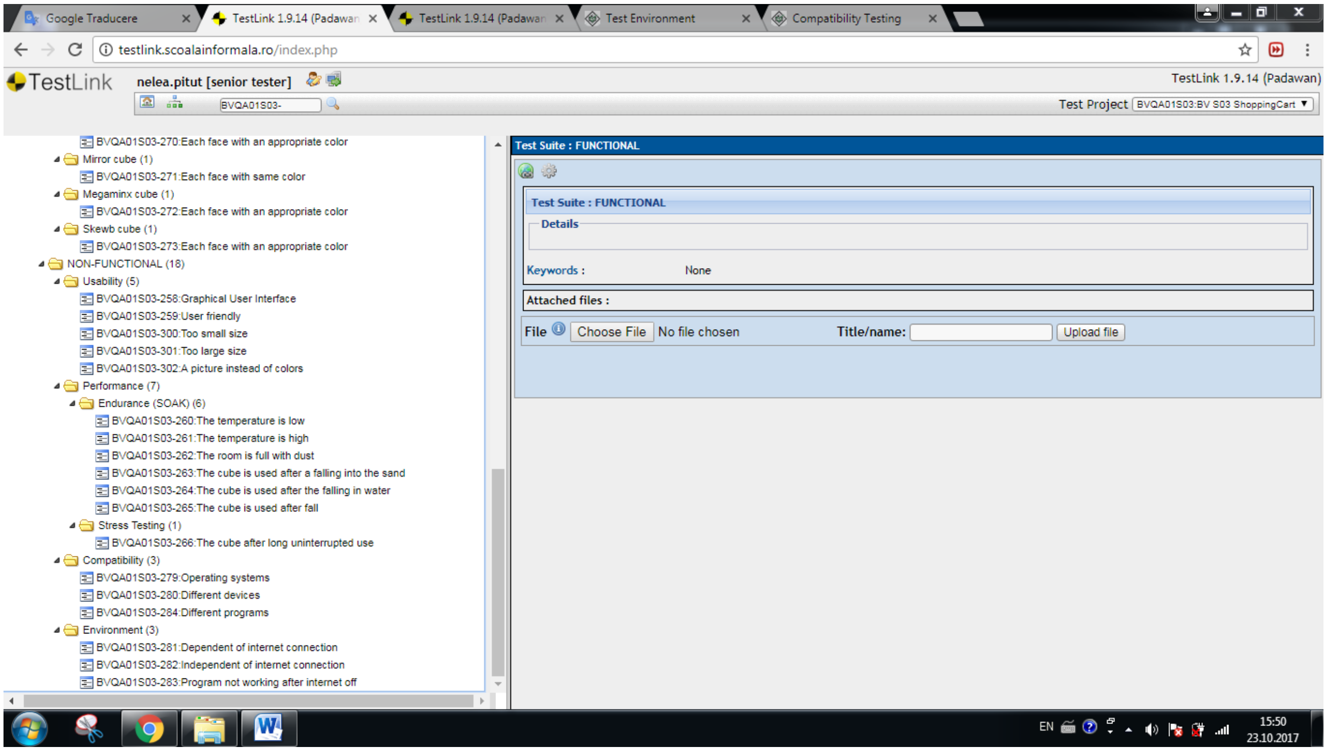
Test Suite : FUNCTIONAL

Details

Keywords : None

Attached files :

File No file chosen Title/name:



- BVQA01S03-270:Each face with an appropriate color
- Mirror cube (1)
  - BVQA01S03-271:Each face with same color
- Megaminx cube (1)
  - BVQA01S03-272:Each face with an appropriate color
- Skewb cube (1)
  - BVQA01S03-273:Each face with an appropriate color
- NON-FUNCTIONAL (18)
  - Usability (5)
    - BVQA01S03-258:Graphical User Interface
    - BVQA01S03-259:User friendly
    - BVQA01S03-300:Too small size
    - BVQA01S03-301:Too large size
    - BVQA01S03-302:A picture instead of colors
  - Performance (7)
    - Endurance (SOAK) (6)
      - BVQA01S03-260:The temperature is low
      - BVQA01S03-261:The temperature is high
      - BVQA01S03-262:The room is full with dust
      - BVQA01S03-263:The cube is used after a falling into the sand
      - BVQA01S03-264:The cube is used after the falling in water
      - BVQA01S03-265:The cube is used after fall
    - Stress Testing (1)
      - BVQA01S03-266:The cube after long uninterrupted use
  - Compatibility (3)
    - BVQA01S03-279:Operating systems
    - BVQA01S03-280:Different devices
    - BVQA01S03-284:Different programs
  - Environment (3)
    - BVQA01S03-281:Dependent of internet connection
    - BVQA01S03-282:Independent of internet connection
    - BVQA01S03-283:Program not working after internet off

Test Suite : FUNCTIONAL

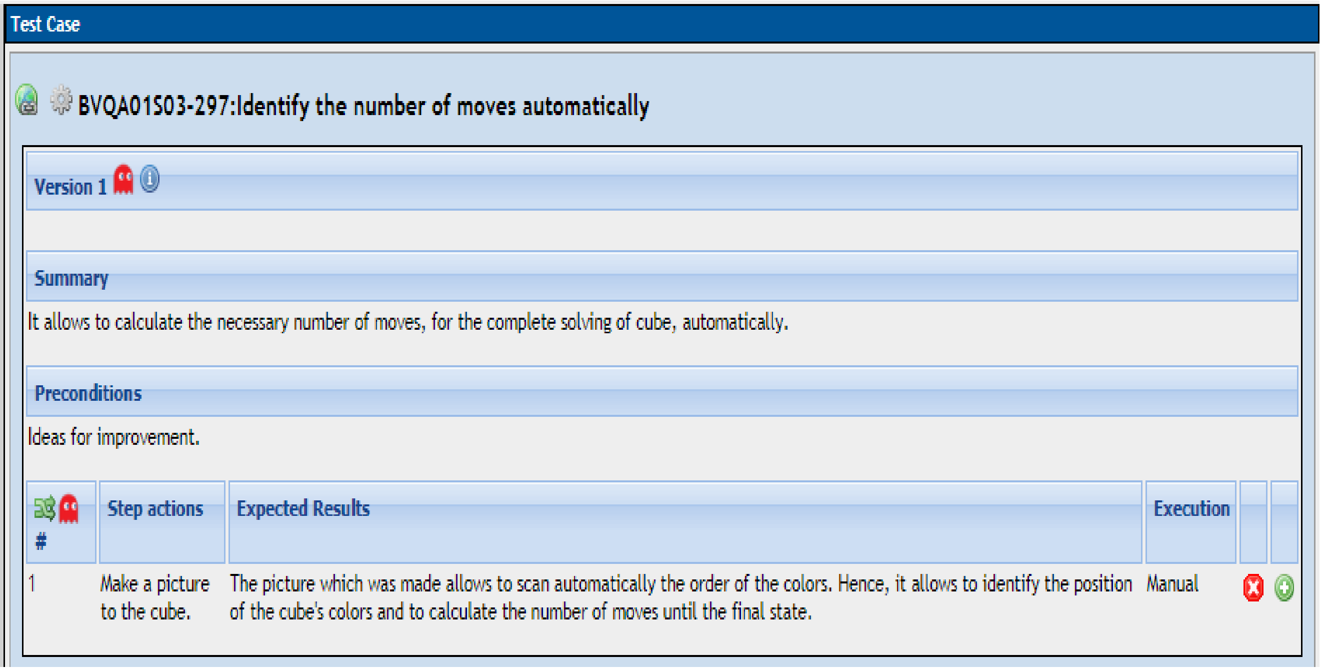
Test Suite : FUNCTIONAL

Details

Keywords : None

Attached files :

File No file chosen Title/name:



Test Case

BVQA01S03-297:Identify the number of moves automatically

Version 1

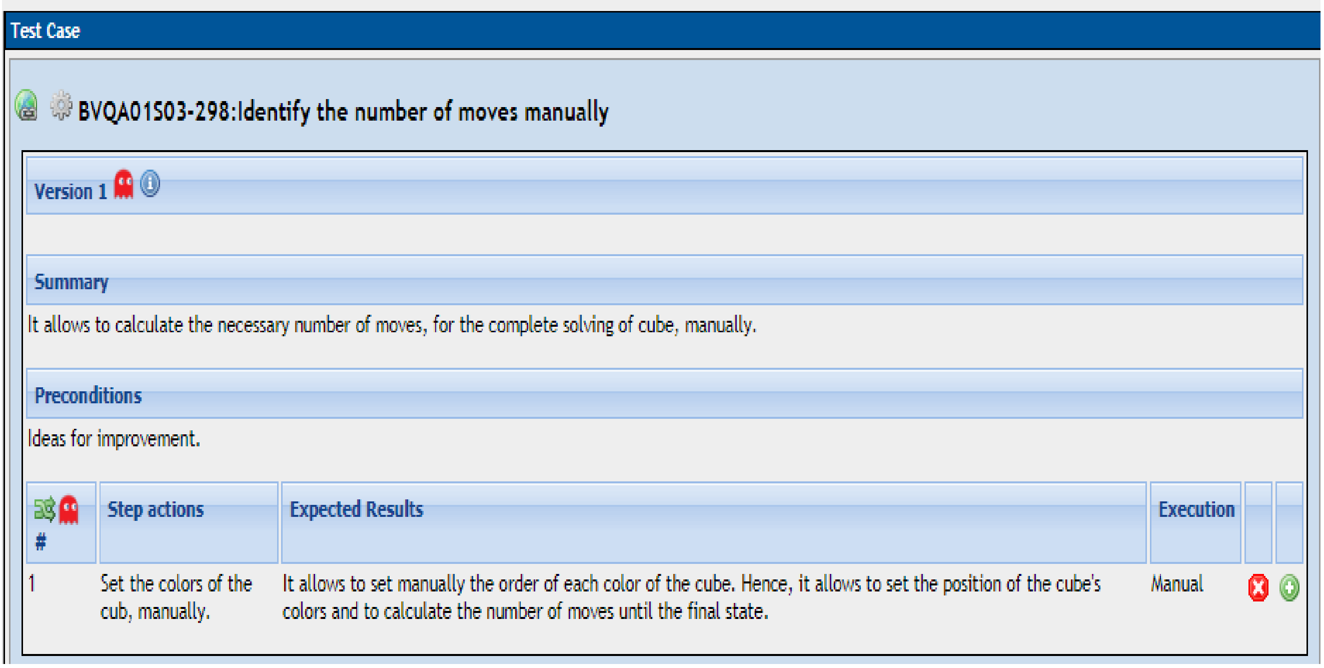
Summary

It allows to calculate the necessary number of moves, for the complete solving of cube, automatically.

Preconditions

Ideas for improvement.

#	Step actions	Expected Results	Execution		
1	Make a picture to the cube.	The picture which was made allows to scan automatically the order of the colors. Hence, it allows to identify the position of the cube's colors and to calculate the number of moves until the final state.	Manual	☒	⊕



Test Case

BVQA01S03-298:Identify the number of moves manually

Version 1

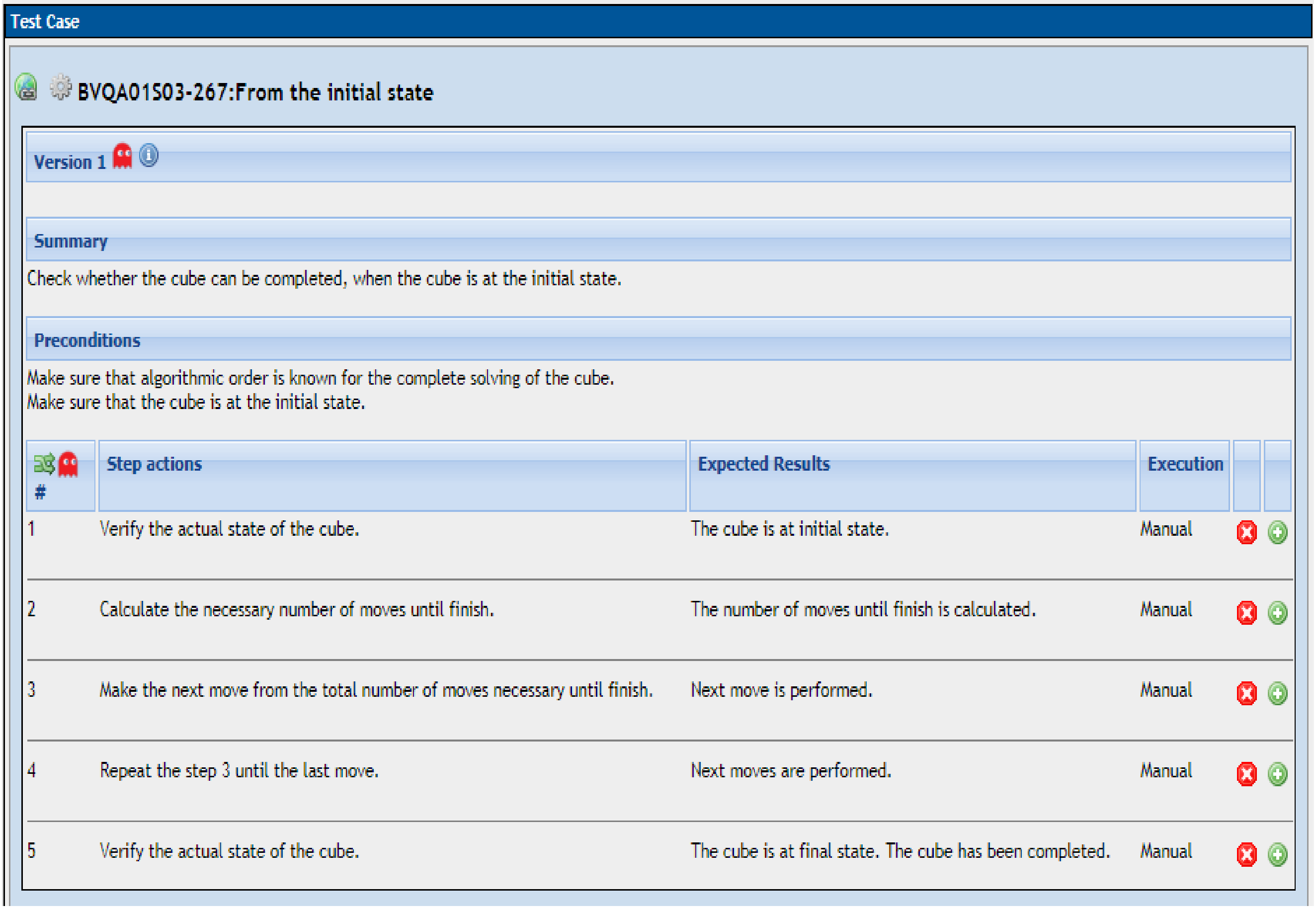
Summary

It allows to calculate the necessary number of moves, for the complete solving of cube, manually.

Preconditions

Ideas for improvement.

#	Step actions	Expected Results	Execution		
1	Set the colors of the cub, manually.	It allows to set manually the order of each color of the cube. Hence, it allows to set the position of the cube's colors and to calculate the number of moves until the final state.	Manual	☒	⊕



Test Case

BVQA01S03-267:From the initial state

Version 1

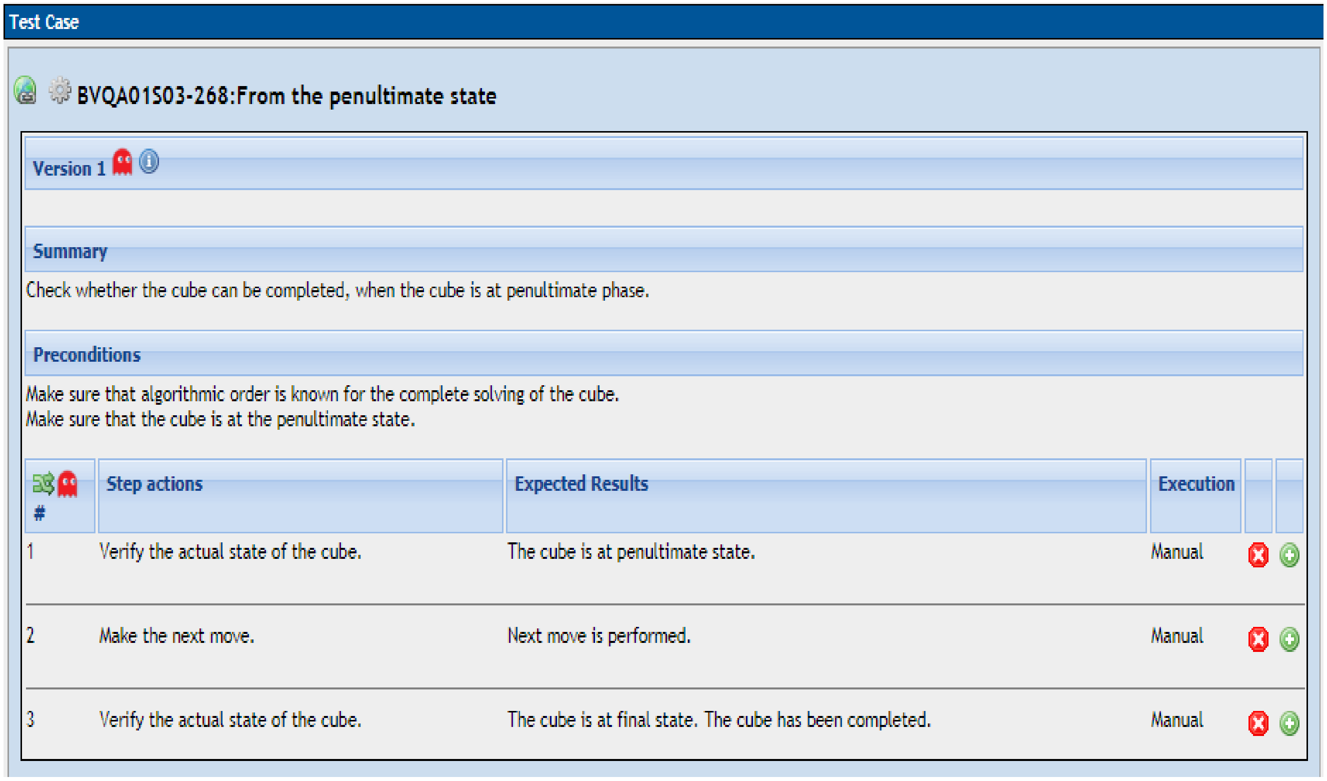
Summary

Check whether the cube can be completed, when the cube is at the initial state.

Preconditions

Make sure that algorithmic order is known for the complete solving of the cube.
Make sure that the cube is at the initial state.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at initial state.	Manual	✖	⊕
2	Calculate the necessary number of moves until finish.	The number of moves until finish is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until finish.	Next move is performed.	Manual	✖	⊕
4	Repeat the step 3 until the last move.	Next moves are performed.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at final state. The cube has been completed.	Manual	✖	⊕



Test Case

BVQA01S03-268:From the penultimate state

Version 1

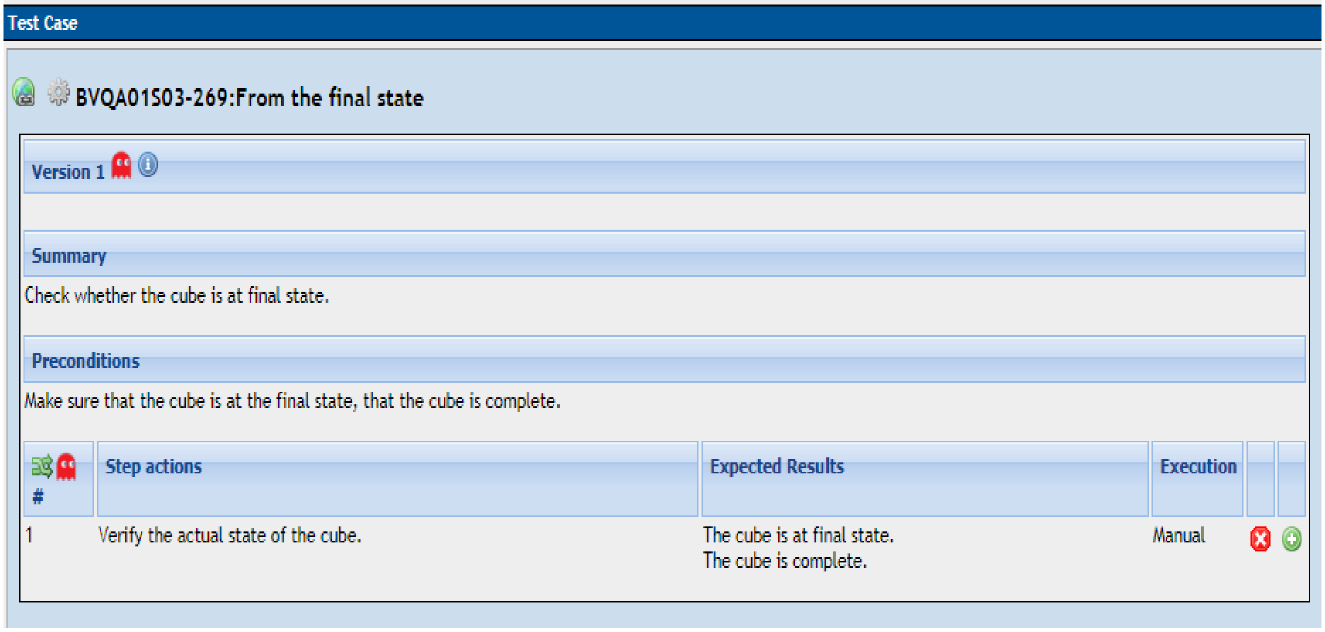
Summary

Check whether the cube can be completed, when the cube is at penultimate phase.

Preconditions

Make sure that algorithmic order is known for the complete solving of the cube.
Make sure that the cube is at the penultimate state.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at penultimate state.	Manual	✖	⊕
2	Make the next move.	Next move is performed.	Manual	✖	⊕
3	Verify the actual state of the cube.	The cube is at final state. The cube has been completed.	Manual	✖	⊕



Test Case

BVQA01S03-269:From the final state

Version 1

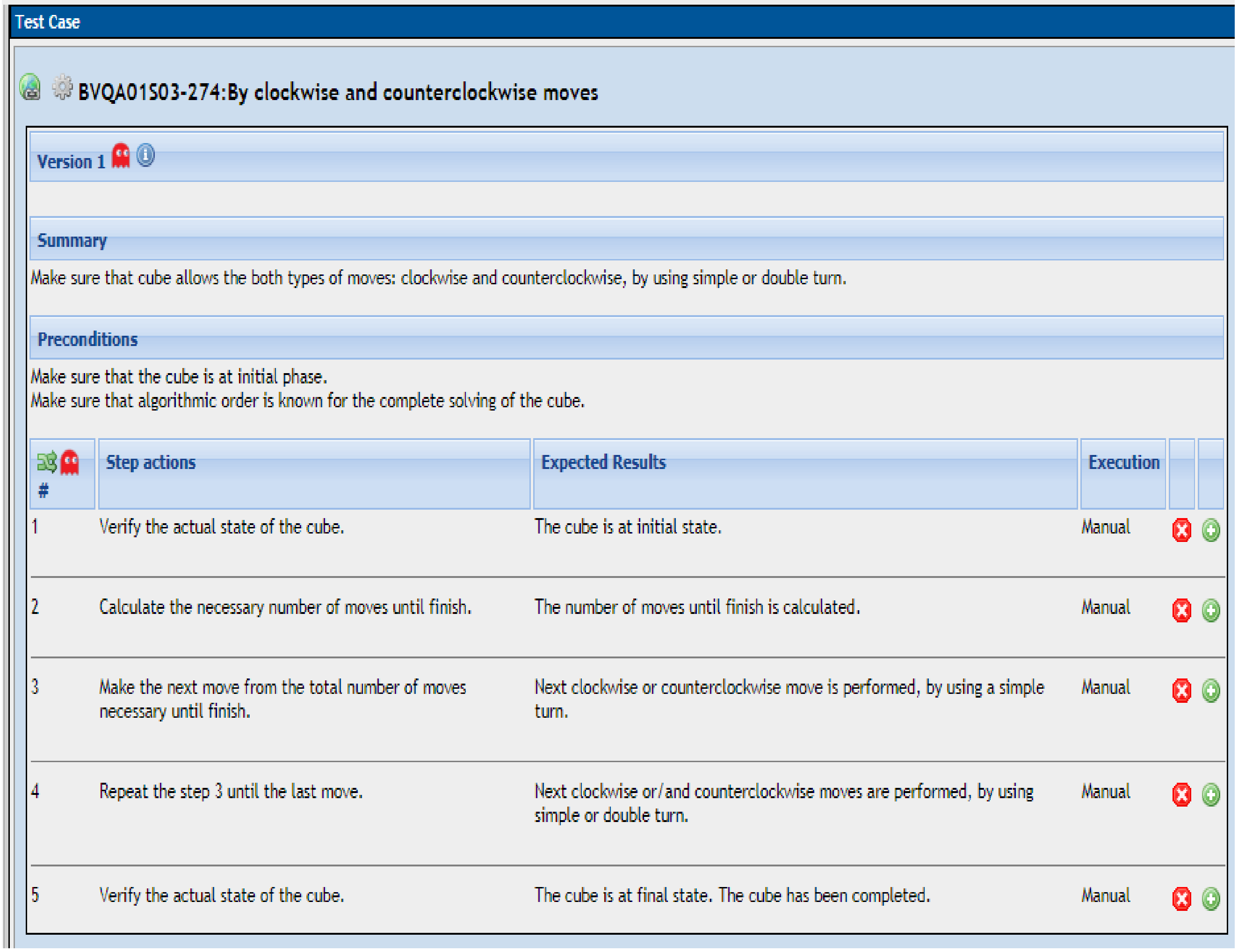
Summary

Check whether the cube is at final state.

Preconditions

Make sure that the cube is at the final state, that the cube is complete.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at final state. The cube is complete.	Manual	✖	⊕



Test Case

BVQA01S03-274:By clockwise and counterclockwise moves

Version 1

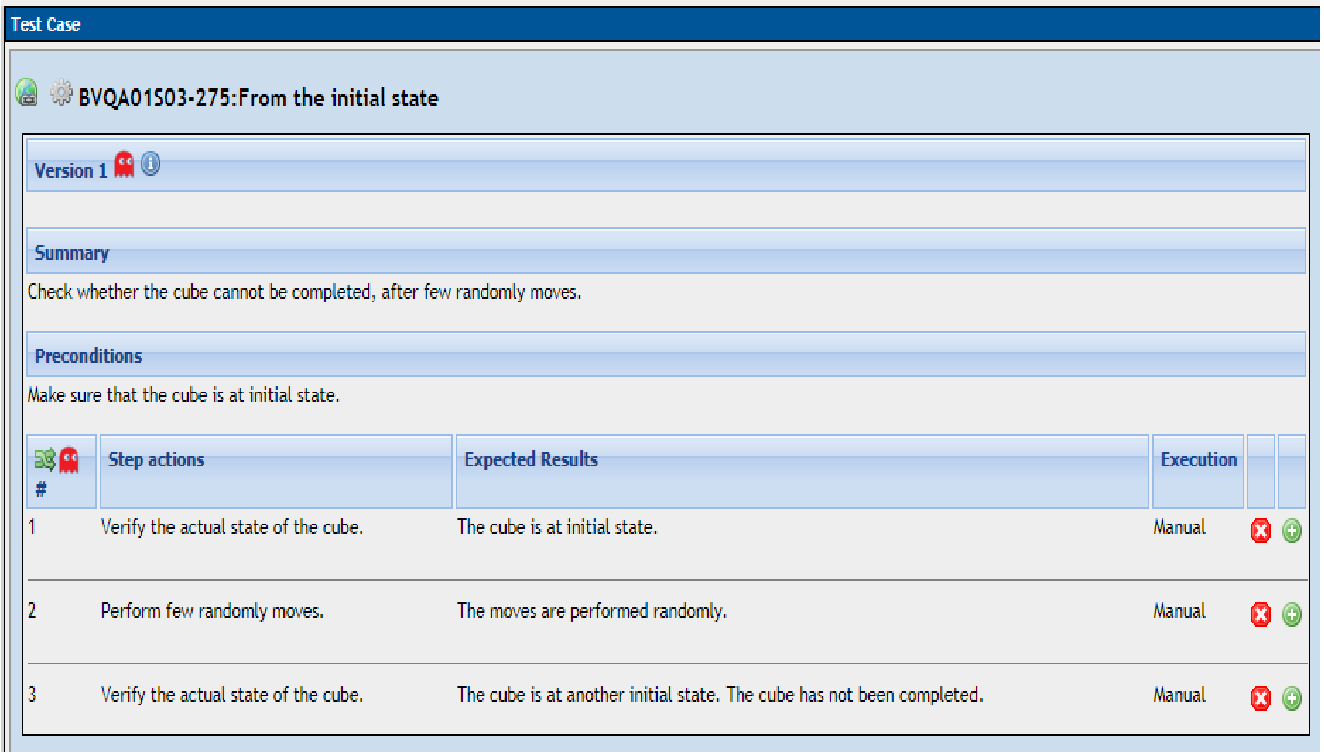
Summary

Make sure that cube allows the both types of moves: clockwise and counterclockwise, by using simple or double turn.

Preconditions

Make sure that the cube is at initial phase.
Make sure that algorithmic order is known for the complete solving of the cube.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at initial state.	Manual	✖	⊕
2	Calculate the necessary number of moves until finish.	The number of moves until finish is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until finish.	Next clockwise or counterclockwise move is performed, by using a simple turn.	Manual	✖	⊕
4	Repeat the step 3 until the last move.	Next clockwise or/and counterclockwise moves are performed, by using simple or double turn.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at final state. The cube has been completed.	Manual	✖	⊕



Test Case

BVQA01S03-275:From the initial state

Version 1

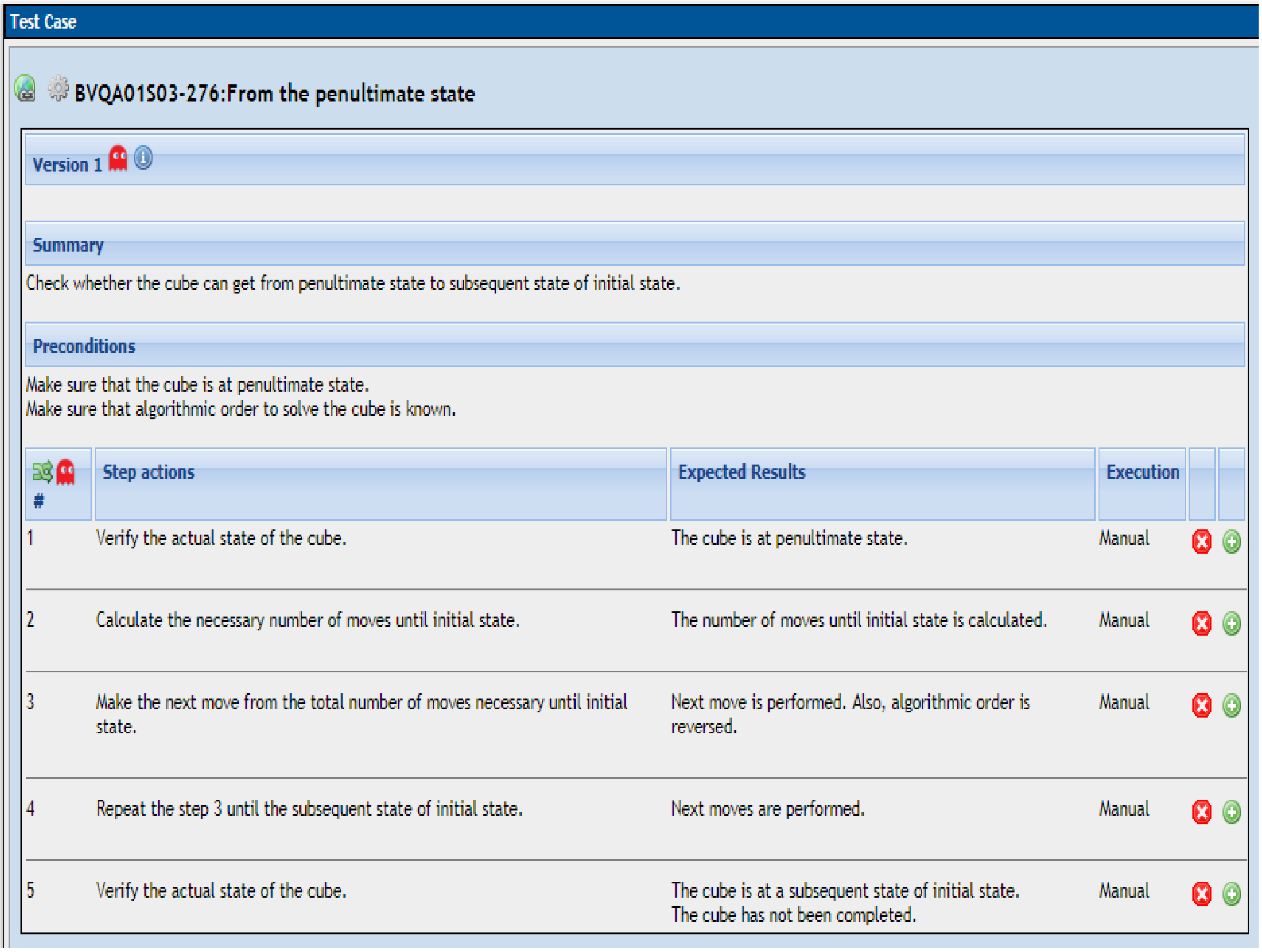
Summary

Check whether the cube cannot be completed, after few randomly moves.

Preconditions

Make sure that the cube is at initial state.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at initial state.	Manual	✖	⊕
2	Perform few randomly moves.	The moves are performed randomly.	Manual	✖	⊕
3	Verify the actual state of the cube.	The cube is at another initial state. The cube has not been completed.	Manual	✖	⊕



Test Case

BVQA01S03-276:From the penultimate state

Version 1

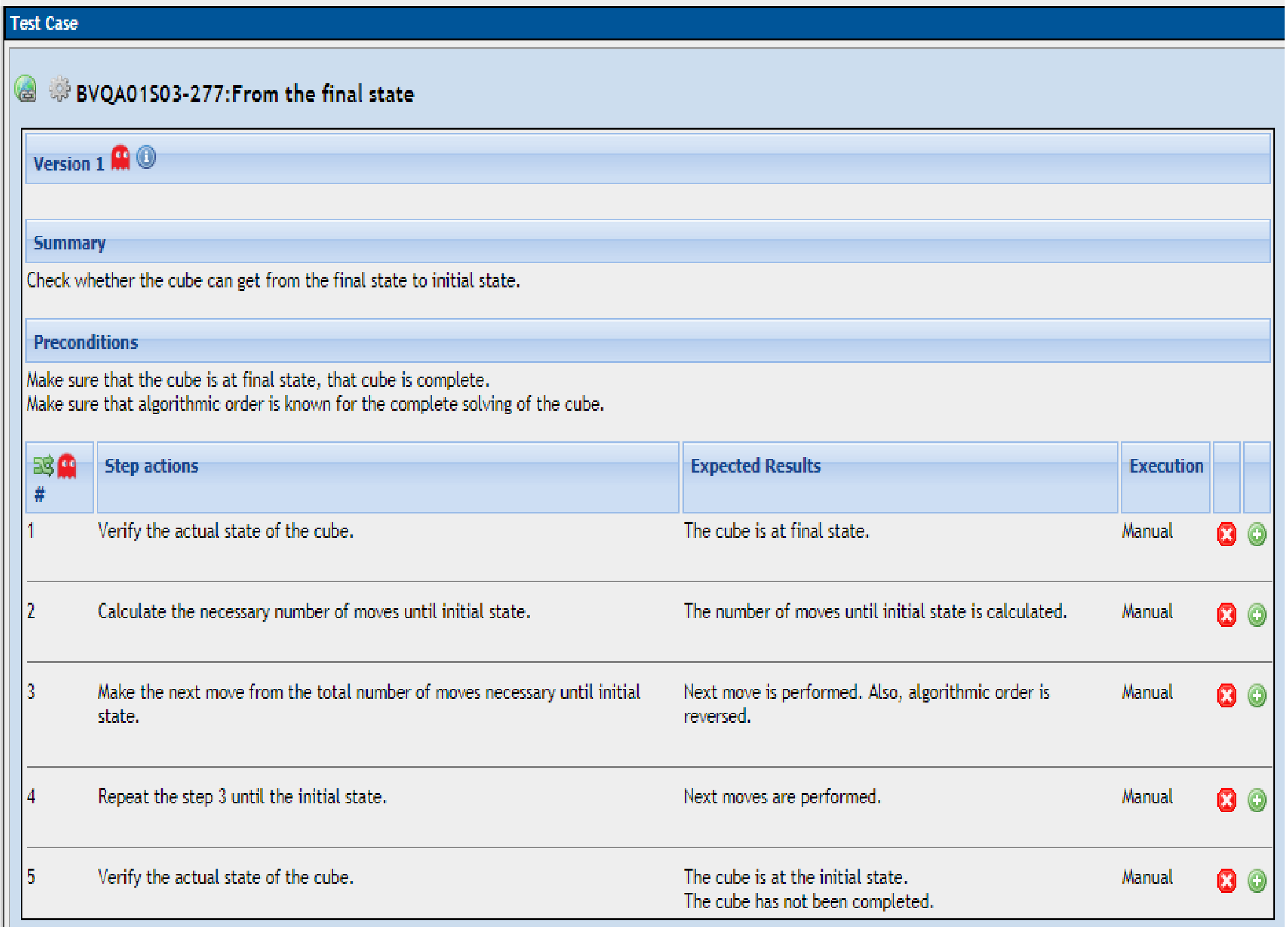
Summary

Check whether the cube can get from penultimate state to subsequent state of initial state.

Preconditions

Make sure that the cube is at penultimate state.
Make sure that algorithmic order to solve the cube is known.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at penultimate state.	Manual	✖	⊕
2	Calculate the necessary number of moves until initial state.	The number of moves until initial state is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until initial state.	Next move is performed. Also, algorithmic order is reversed.	Manual	✖	⊕
4	Repeat the step 3 until the subsequent state of initial state.	Next moves are performed.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at a subsequent state of initial state. The cube has not been completed.	Manual	✖	⊕



Test Case

BVQA01S03-277:From the final state

Version 1

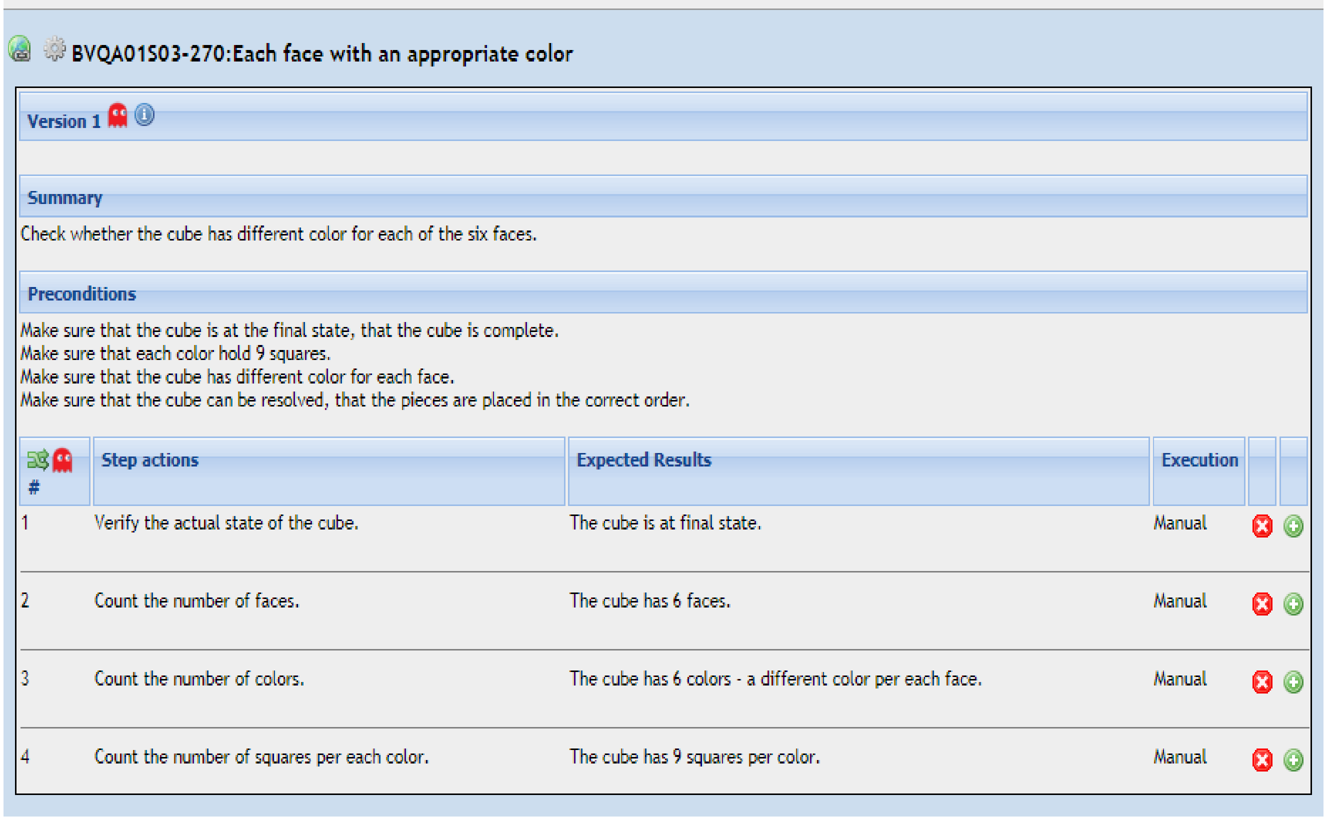
Summary

Check whether the cube can get from the final state to initial state.

Preconditions

Make sure that the cube is at final state, that cube is complete.
Make sure that algorithmic order is known for the complete solving of the cube.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at final state.	Manual	☒	⊕
2	Calculate the necessary number of moves until initial state.	The number of moves until initial state is calculated.	Manual	☒	⊕
3	Make the next move from the total number of moves necessary until initial state.	Next move is performed. Also, algorithmic order is reversed.	Manual	☒	⊕
4	Repeat the step 3 until the initial state.	Next moves are performed.	Manual	☒	⊕
5	Verify the actual state of the cube.	The cube is at the initial state. The cube has not been completed.	Manual	☒	⊕



BVQA01S03-270:Each face with an appropriate color

Version 1

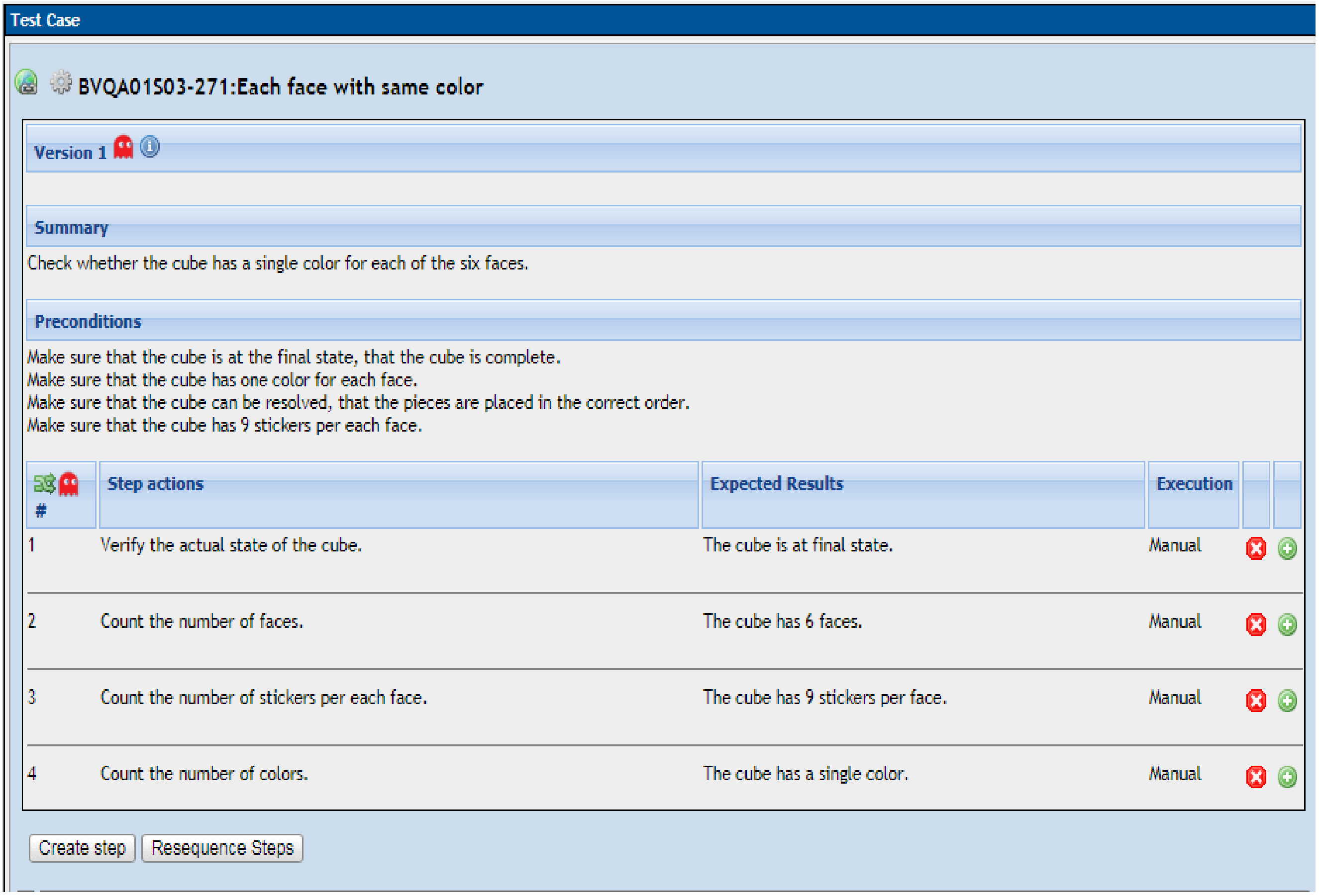
Summary

Check whether the cube has different color for each of the six faces.

Preconditions

Make sure that the cube is at the final state, that the cube is complete.
Make sure that each color hold 9 squares.
Make sure that the cube has different color for each face.
Make sure that the cube can be resolved, that the pieces are placed in the correct order.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at final state.	Manual	☒	⊕
2	Count the number of faces.	The cube has 6 faces.	Manual	☒	⊕
3	Count the number of colors.	The cube has 6 colors - a different color per each face.	Manual	☒	⊕
4	Count the number of squares per each color.	The cube has 9 squares per color.	Manual	☒	⊕



Test Case

BVQA01S03-271:Each face with same color

Version 1

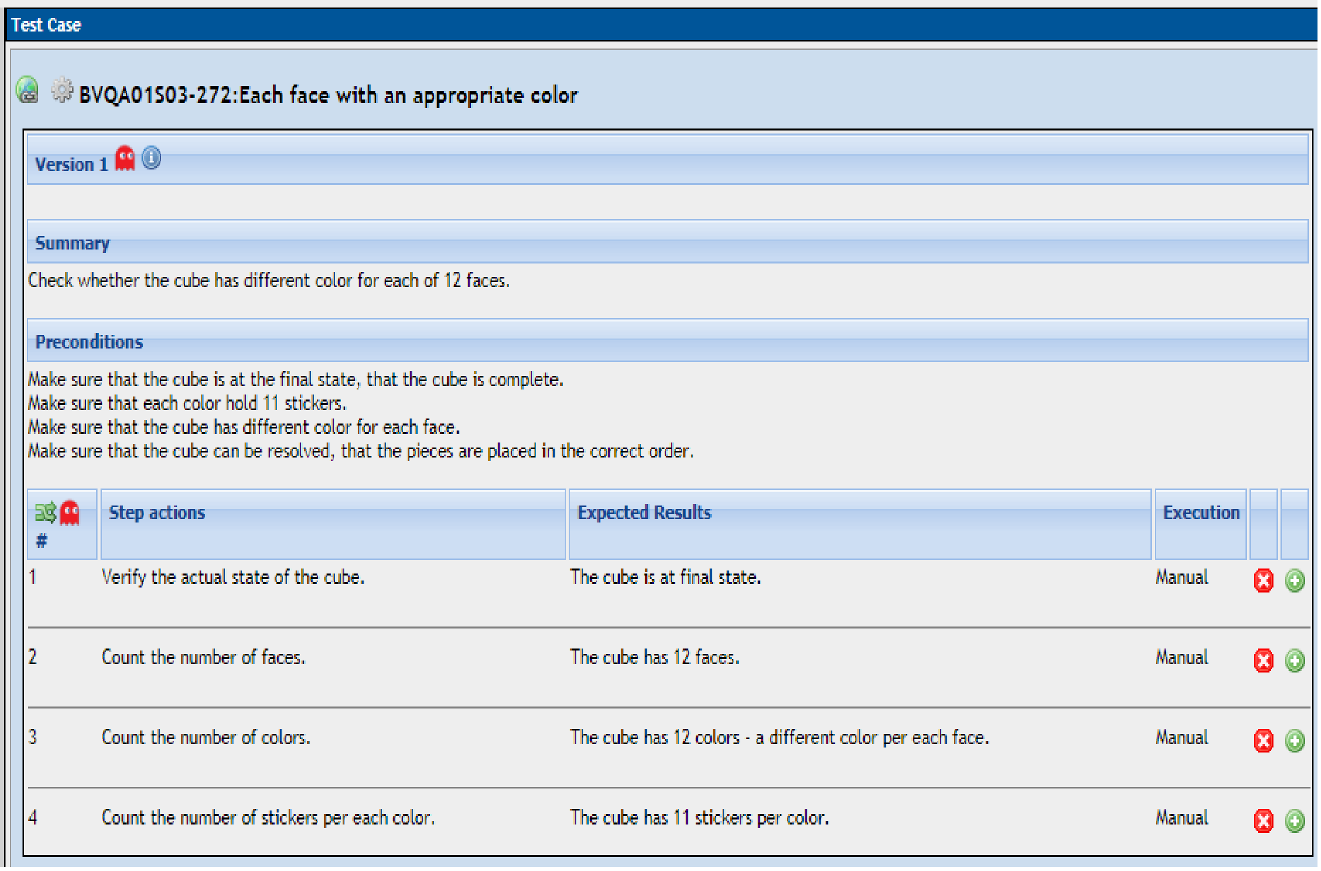
Summary

Check whether the cube has a single color for each of the six faces.

Preconditions

Make sure that the cube is at the final state, that the cube is complete.
Make sure that the cube has one color for each face.
Make sure that the cube can be resolved, that the pieces are placed in the correct order.
Make sure that the cube has 9 stickers per each face.

#	Step actions	Expected Results	Execution
1	Verify the actual state of the cube.	The cube is at final state.	Manual
2	Count the number of faces.	The cube has 6 faces.	Manual
3	Count the number of stickers per each face.	The cube has 9 stickers per face.	Manual
4	Count the number of colors.	The cube has a single color.	Manual



Test Case

BVQA01S03-272:Each face with an appropriate color

Version 1

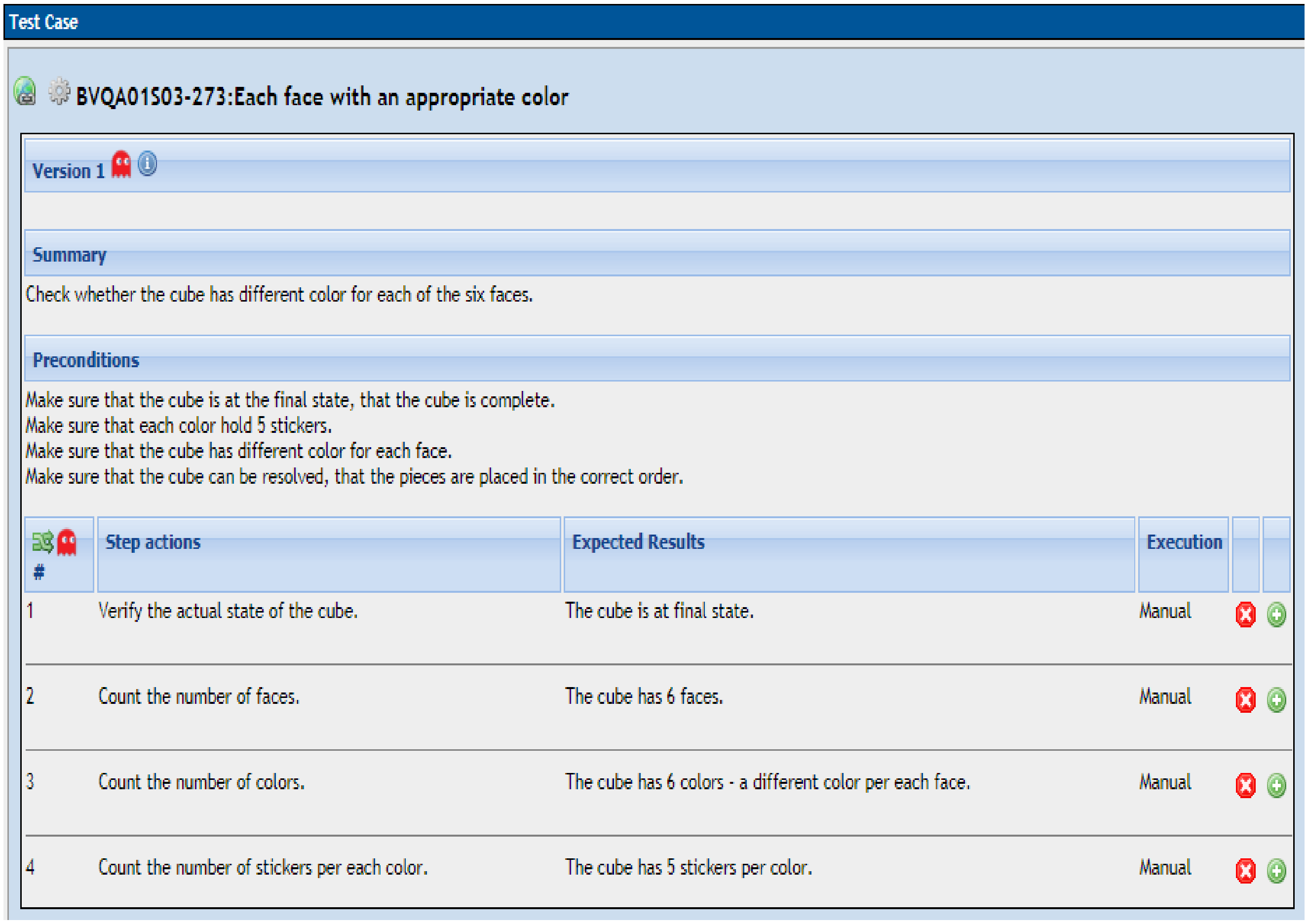
Summary

Check whether the cube has different color for each of 12 faces.

Preconditions

Make sure that the cube is at the final state, that the cube is complete.
Make sure that each color hold 11 stickers.
Make sure that the cube has different color for each face.
Make sure that the cube can be resolved, that the pieces are placed in the correct order.

#	Step actions	Expected Results	Execution
1	Verify the actual state of the cube.	The cube is at final state.	Manual
2	Count the number of faces.	The cube has 12 faces.	Manual
3	Count the number of colors.	The cube has 12 colors - a different color per each face.	Manual
4	Count the number of stickers per each color.	The cube has 11 stickers per color.	Manual



Test Case

BVQA01S03-273:Each face with an appropriate color

Version 1

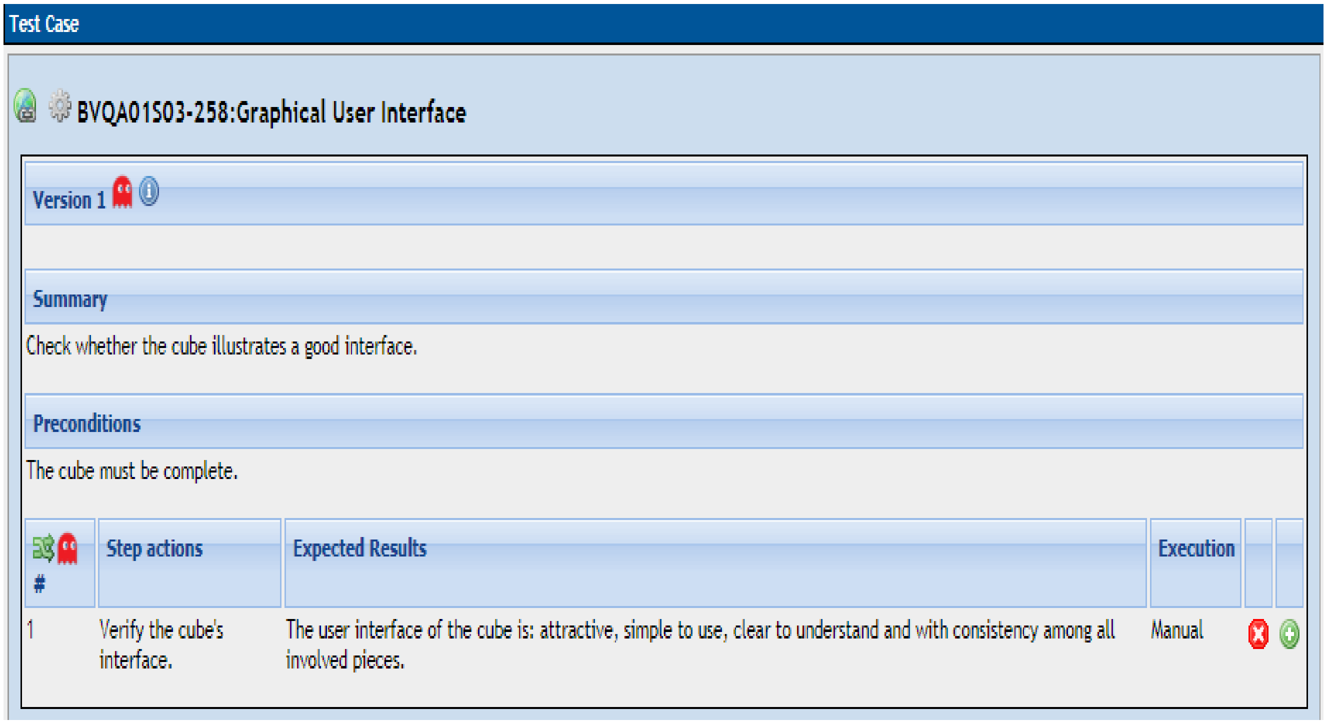
Summary

Check whether the cube has different color for each of the six faces.

Preconditions

Make sure that the cube is at the final state, that the cube is complete.
Make sure that each color hold 5 stickers.
Make sure that the cube has different color for each face.
Make sure that the cube can be resolved, that the pieces are placed in the correct order.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at final state.	Manual	✖	⊕
2	Count the number of faces.	The cube has 6 faces.	Manual	✖	⊕
3	Count the number of colors.	The cube has 6 colors - a different color per each face.	Manual	✖	⊕
4	Count the number of stickers per each color.	The cube has 5 stickers per color.	Manual	✖	⊕



Test Case

BVQA01S03-258:Graphical User Interface

Version 1

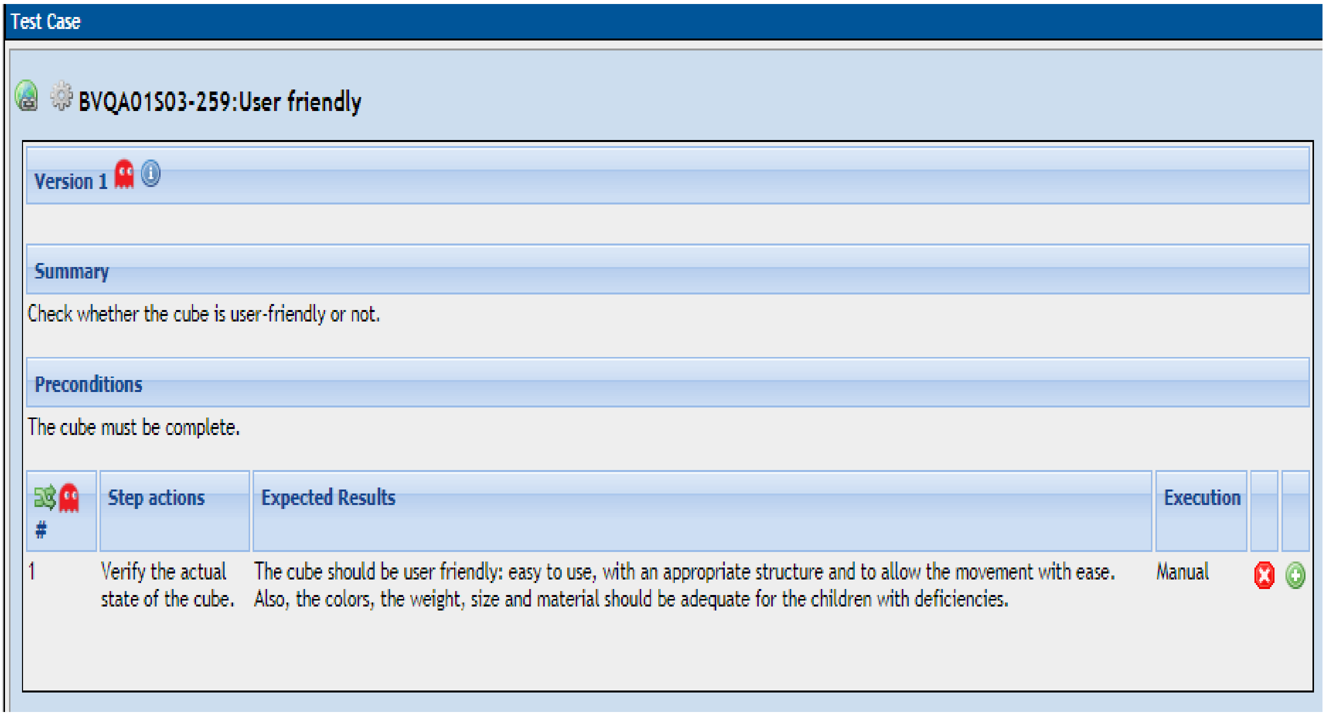
Summary

Check whether the cube illustrates a good interface.

Preconditions

The cube must be complete.

#	Step actions	Expected Results	Execution		
1	Verify the cube's interface.	The user interface of the cube is: attractive, simple to use, clear to understand and with consistency among all involved pieces.	Manual	✖	⊕



Test Case

BVQA01S03-259:User friendly

Version 1

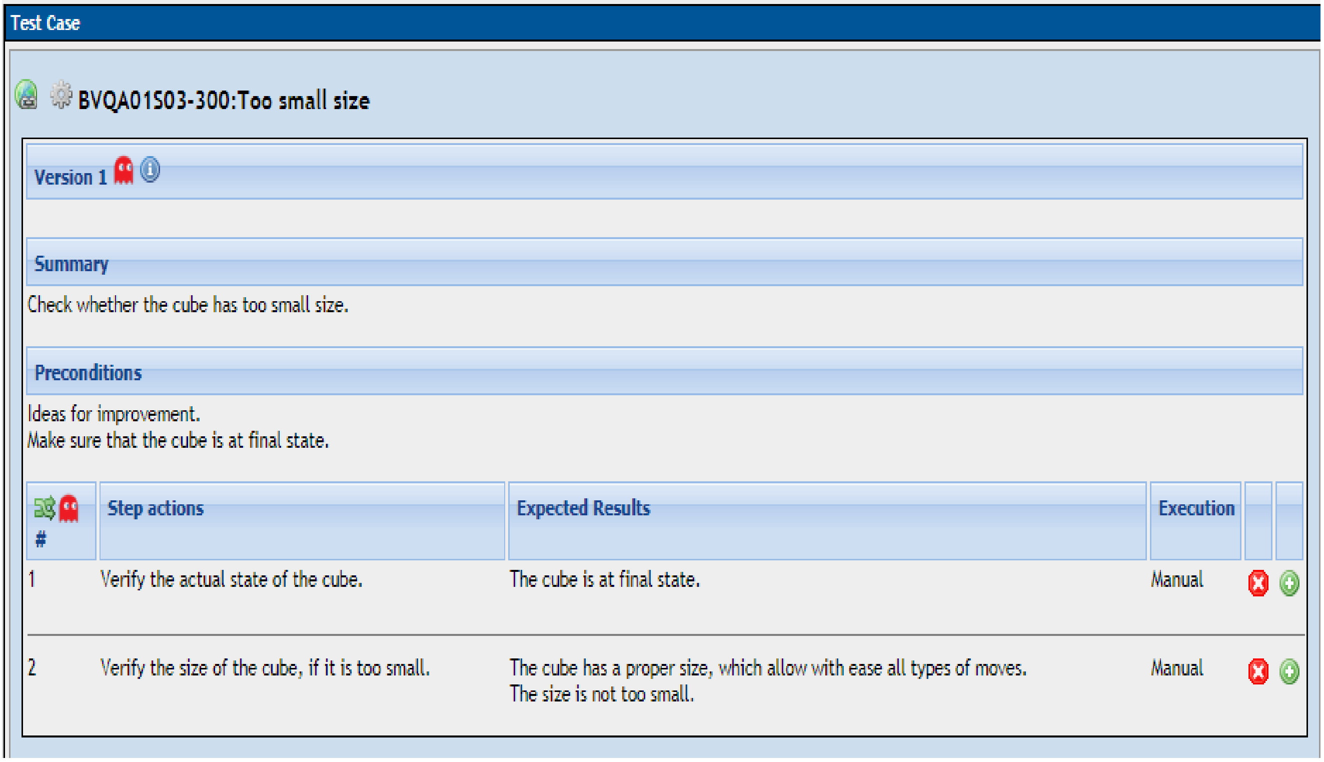
Summary

Check whether the cube is user-friendly or not.

Preconditions

The cube must be complete.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube should be user friendly: easy to use, with an appropriate structure and to allow the movement with ease. Also, the colors, the weight, size and material should be adequate for the children with deficiencies.	Manual	✖	⊕



Test Case

BVQA01S03-300:Too small size

Version 1

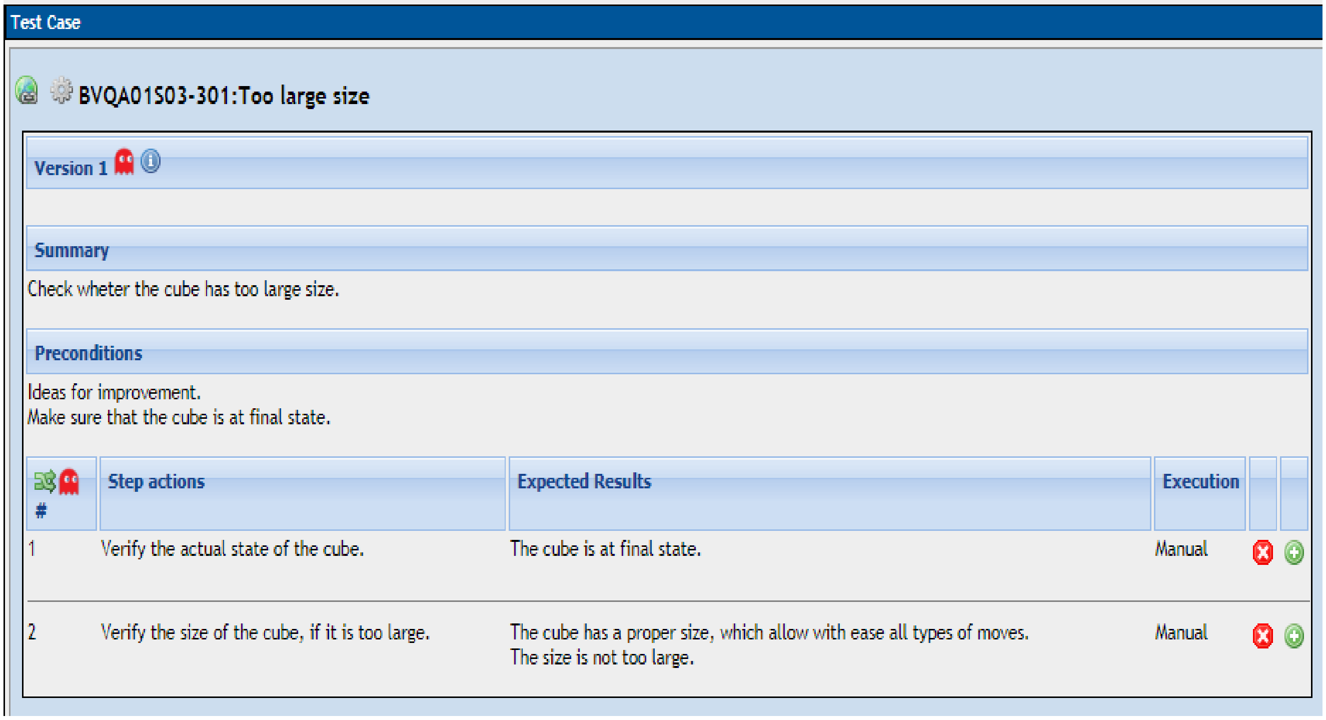
Summary

Check whether the cube has too small size.

Preconditions

Ideas for improvement.
Make sure that the cube is at final state.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at final state.	Manual	✖	⊕
2	Verify the size of the cube, if it is too small.	The cube has a proper size, which allow with ease all types of moves. The size is not too small.	Manual	✖	⊕



Test Case

BVQA01S03-301:Too large size

Version 1

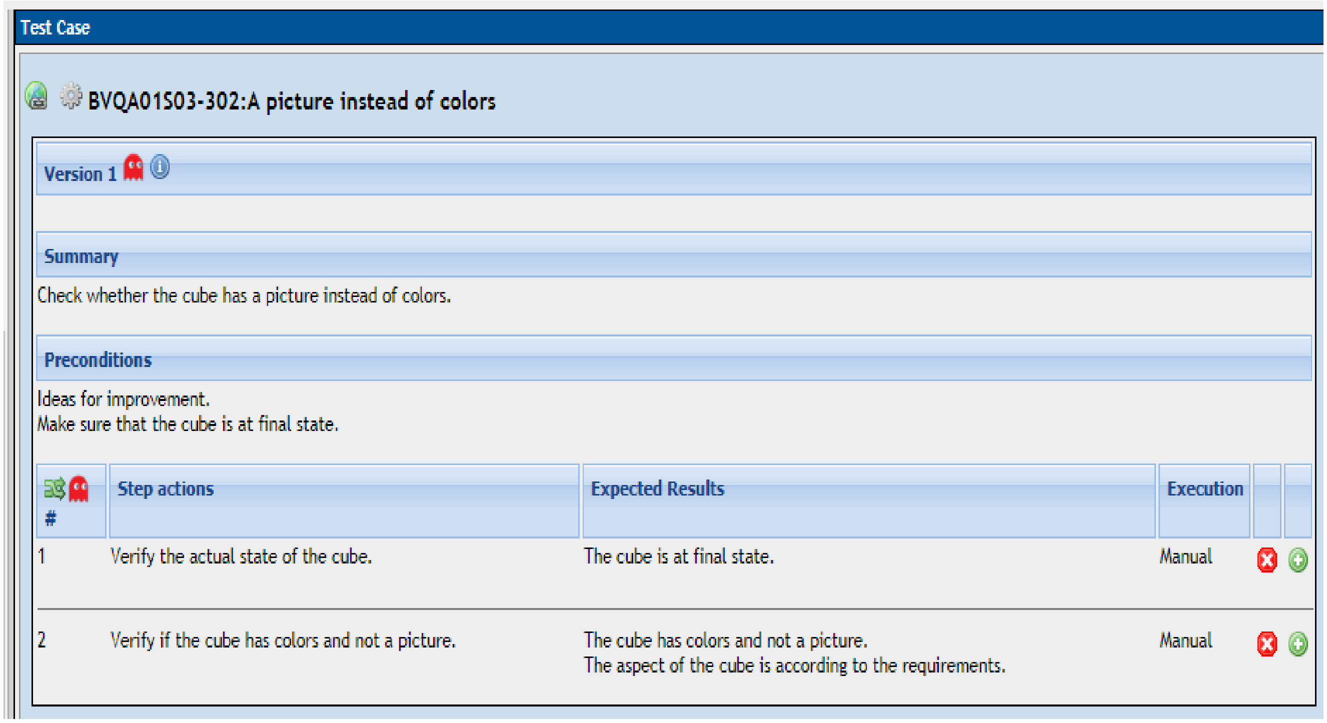
Summary

Check wheter the cube has too large size.

Preconditions

Ideas for improvement.
Make sure that the cube is at final state.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at final state.	Manual	✖	⊕
2	Verify the size of the cube, if it is too large.	The cube has a proper size, which allow with ease all types of moves. The size is not too large.	Manual	✖	⊕



Test Case

BVQA01S03-302:A picture instead of colors

Version 1

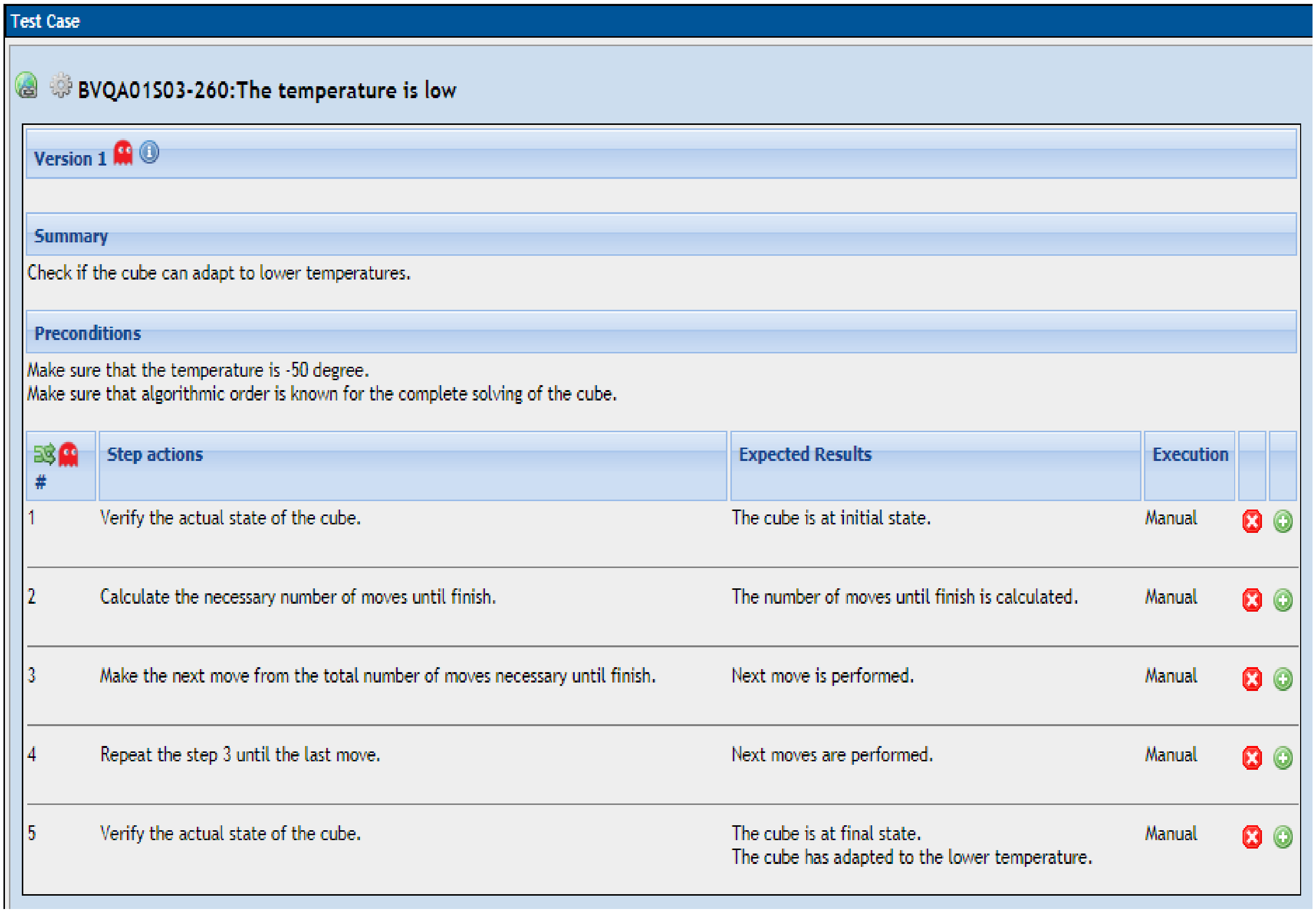
Summary

Check whether the cube has a picture instead of colors.

Preconditions

Ideas for improvement.
Make sure that the cube is at final state.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at final state.	Manual	✖	⊕
2	Verify if the cube has colors and not a picture.	The cube has colors and not a picture. The aspect of the cube is according to the requirements.	Manual	✖	⊕



Test Case

BVQA01S03-260:The temperature is low

Version 1

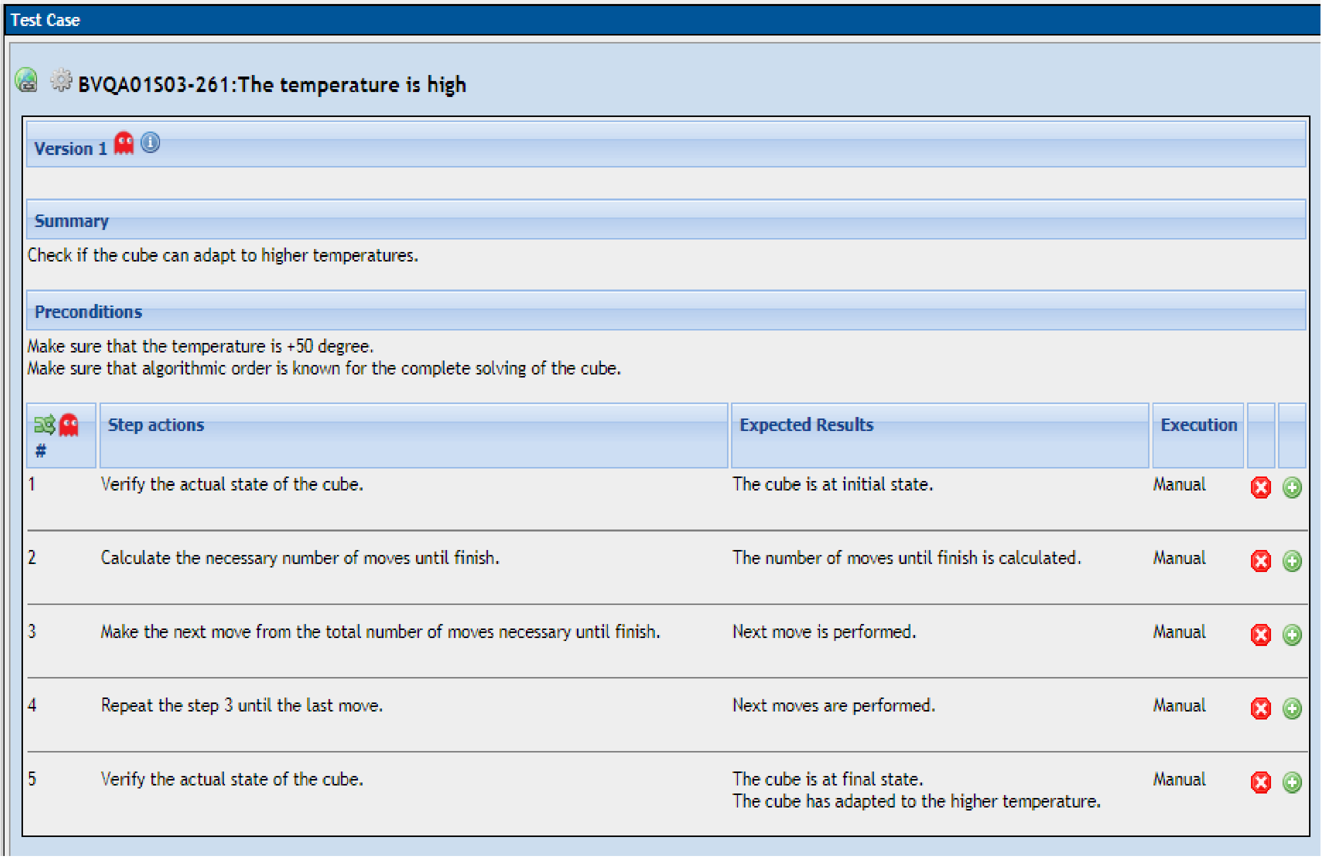
Summary

Check if the cube can adapt to lower temperatures.

Preconditions

Make sure that the temperature is -50 degree.
Make sure that algorithmic order is known for the complete solving of the cube.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at initial state.	Manual	✖	⊕
2	Calculate the necessary number of moves until finish.	The number of moves until finish is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until finish.	Next move is performed.	Manual	✖	⊕
4	Repeat the step 3 until the last move.	Next moves are performed.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at final state. The cube has adapted to the lower temperature.	Manual	✖	⊕



Test Case

BVQA01S03-261:The temperature is high

Version 1

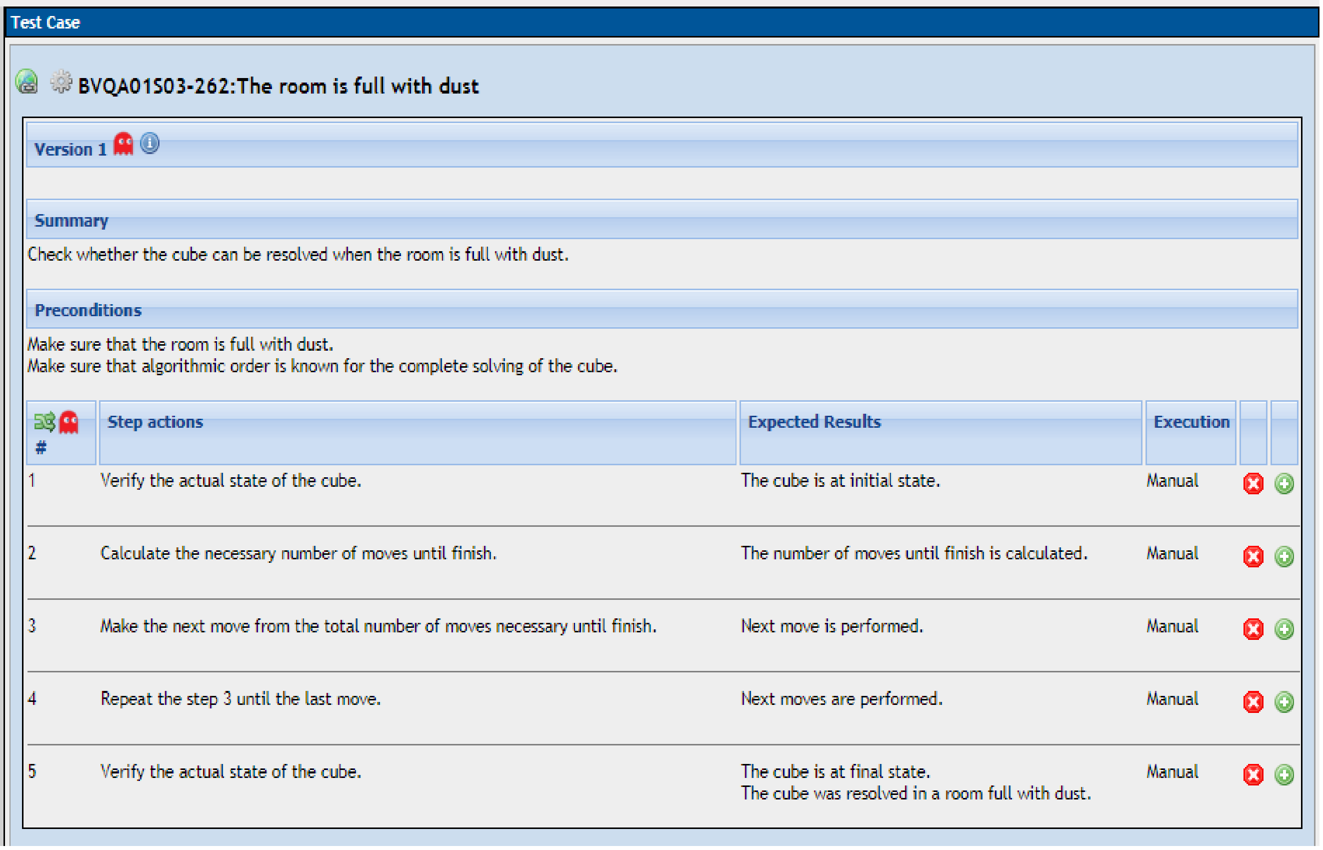
Summary

Check if the cube can adapt to higher temperatures.

Preconditions

Make sure that the temperature is +50 degree.
Make sure that algorithmic order is known for the complete solving of the cube.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at initial state.	Manual	✖	⊕
2	Calculate the necessary number of moves until finish.	The number of moves until finish is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until finish.	Next move is performed.	Manual	✖	⊕
4	Repeat the step 3 until the last move.	Next moves are performed.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at final state. The cube has adapted to the higher temperature.	Manual	✖	⊕



Test Case

BVQA01S03-262:The room is full with dust

Version 1

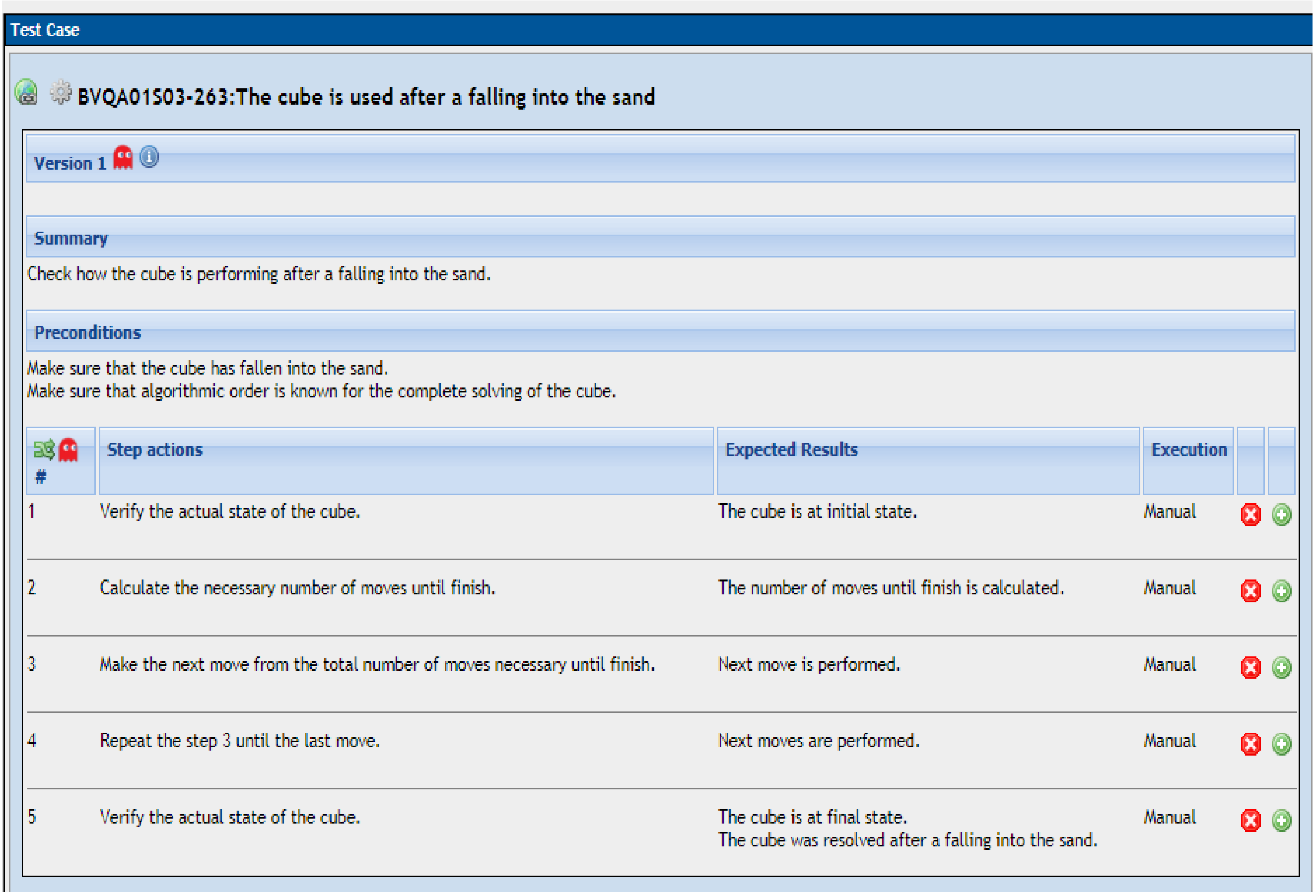
Summary

Check whether the cube can be resolved when the room is full with dust.

Preconditions

Make sure that the room is full with dust.
Make sure that algorithmic order is known for the complete solving of the cube.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at initial state.	Manual	✖	⊕
2	Calculate the necessary number of moves until finish.	The number of moves until finish is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until finish.	Next move is performed.	Manual	✖	⊕
4	Repeat the step 3 until the last move.	Next moves are performed.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at final state. The cube was resolved in a room full with dust.	Manual	✖	⊕



Test Case

BVQA01S03-263:The cube is used after a falling into the sand

Version 1

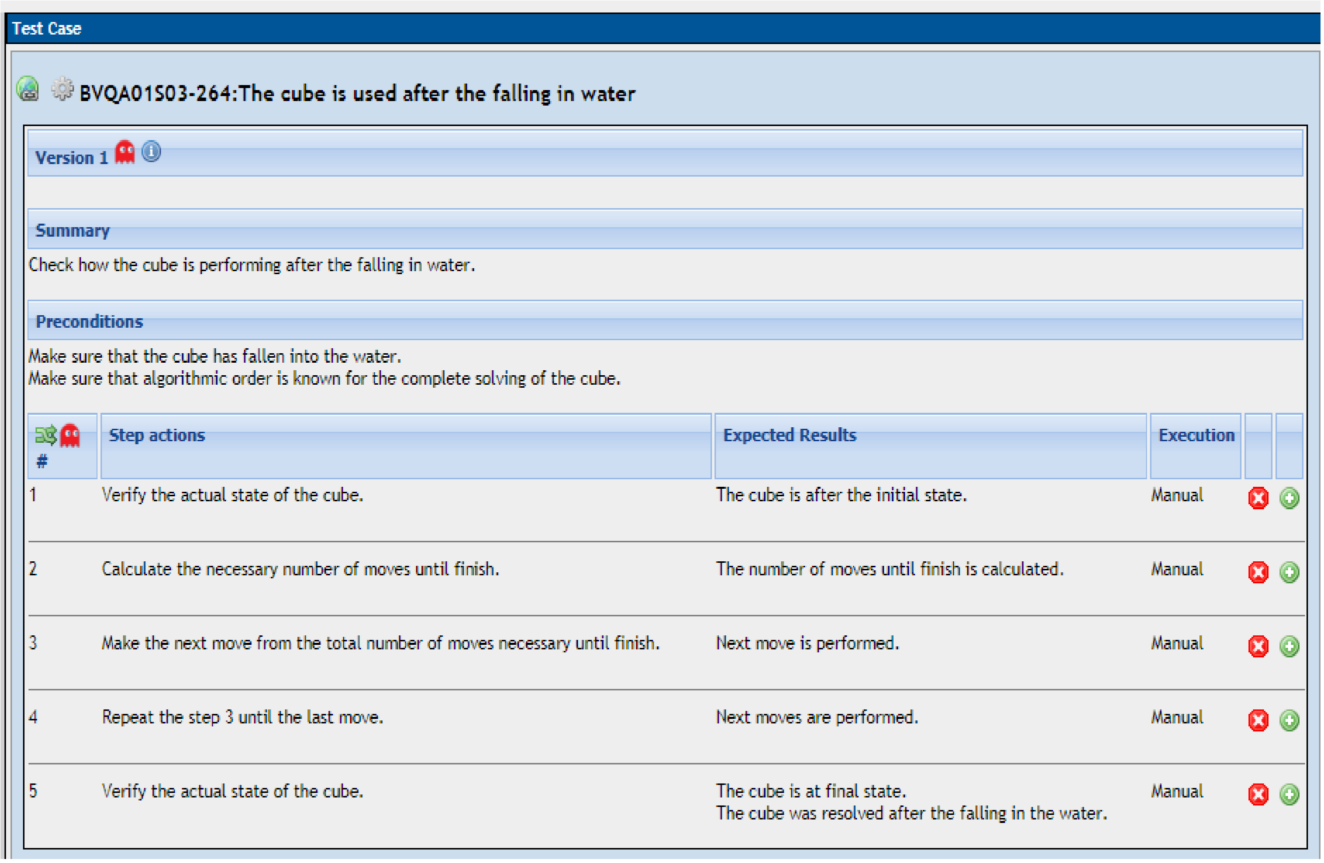
Summary

Check how the cube is performing after a falling into the sand.

Preconditions

Make sure that the cube has fallen into the sand.
Make sure that algorithmic order is known for the complete solving of the cube.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is at initial state.	Manual	✖	⊕
2	Calculate the necessary number of moves until finish.	The number of moves until finish is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until finish.	Next move is performed.	Manual	✖	⊕
4	Repeat the step 3 until the last move.	Next moves are performed.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at final state. The cube was resolved after a falling into the sand.	Manual	✖	⊕



Test Case

BVQA01S03-264:The cube is used after the falling in water

Version 1

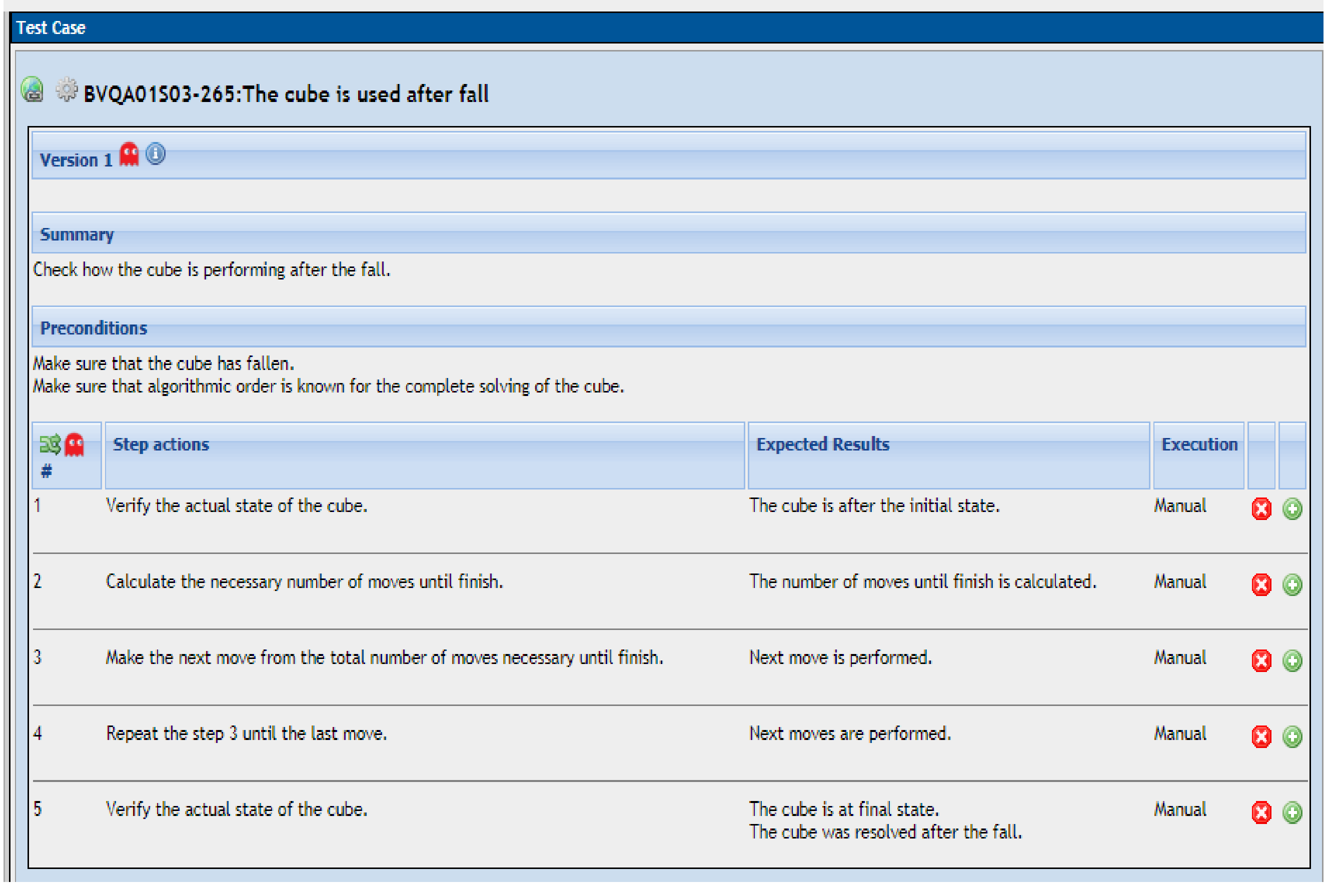
Summary

Check how the cube is performing after the falling in water.

Preconditions

Make sure that the cube has fallen into the water.
Make sure that algorithmic order is known for the complete solving of the cube.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is after the initial state.	Manual	✖	⊕
2	Calculate the necessary number of moves until finish.	The number of moves until finish is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until finish.	Next move is performed.	Manual	✖	⊕
4	Repeat the step 3 until the last move.	Next moves are performed.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at final state. The cube was resolved after the falling in the water.	Manual	✖	⊕



Test Case

BVQA01S03-265:The cube is used after fall

Version 1

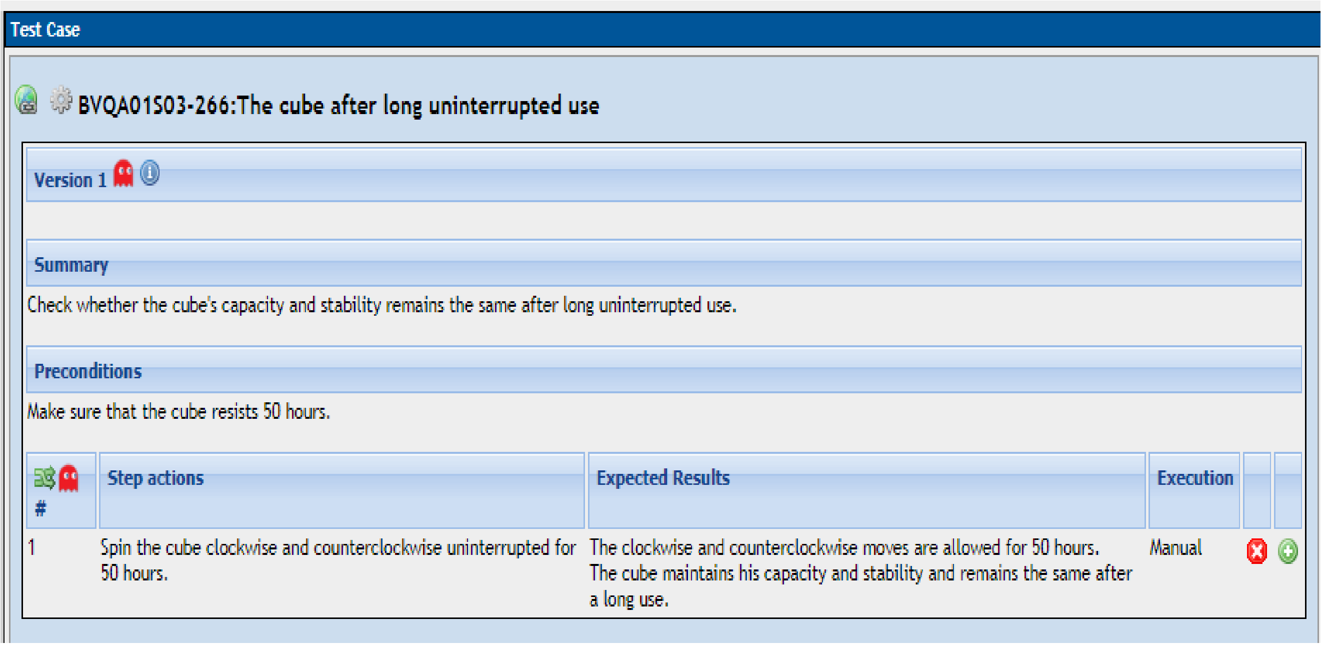
Summary

Check how the cube is performing after the fall.

Preconditions

Make sure that the cube has fallen.
Make sure that algorithmic order is known for the complete solving of the cube.

#	Step actions	Expected Results	Execution		
1	Verify the actual state of the cube.	The cube is after the initial state.	Manual	✖	⊕
2	Calculate the necessary number of moves until finish.	The number of moves until finish is calculated.	Manual	✖	⊕
3	Make the next move from the total number of moves necessary until finish.	Next move is performed.	Manual	✖	⊕
4	Repeat the step 3 until the last move.	Next moves are performed.	Manual	✖	⊕
5	Verify the actual state of the cube.	The cube is at final state. The cube was resolved after the fall.	Manual	✖	⊕



Test Case

BVQA01S03-266:The cube after long uninterrupted use

Version 1

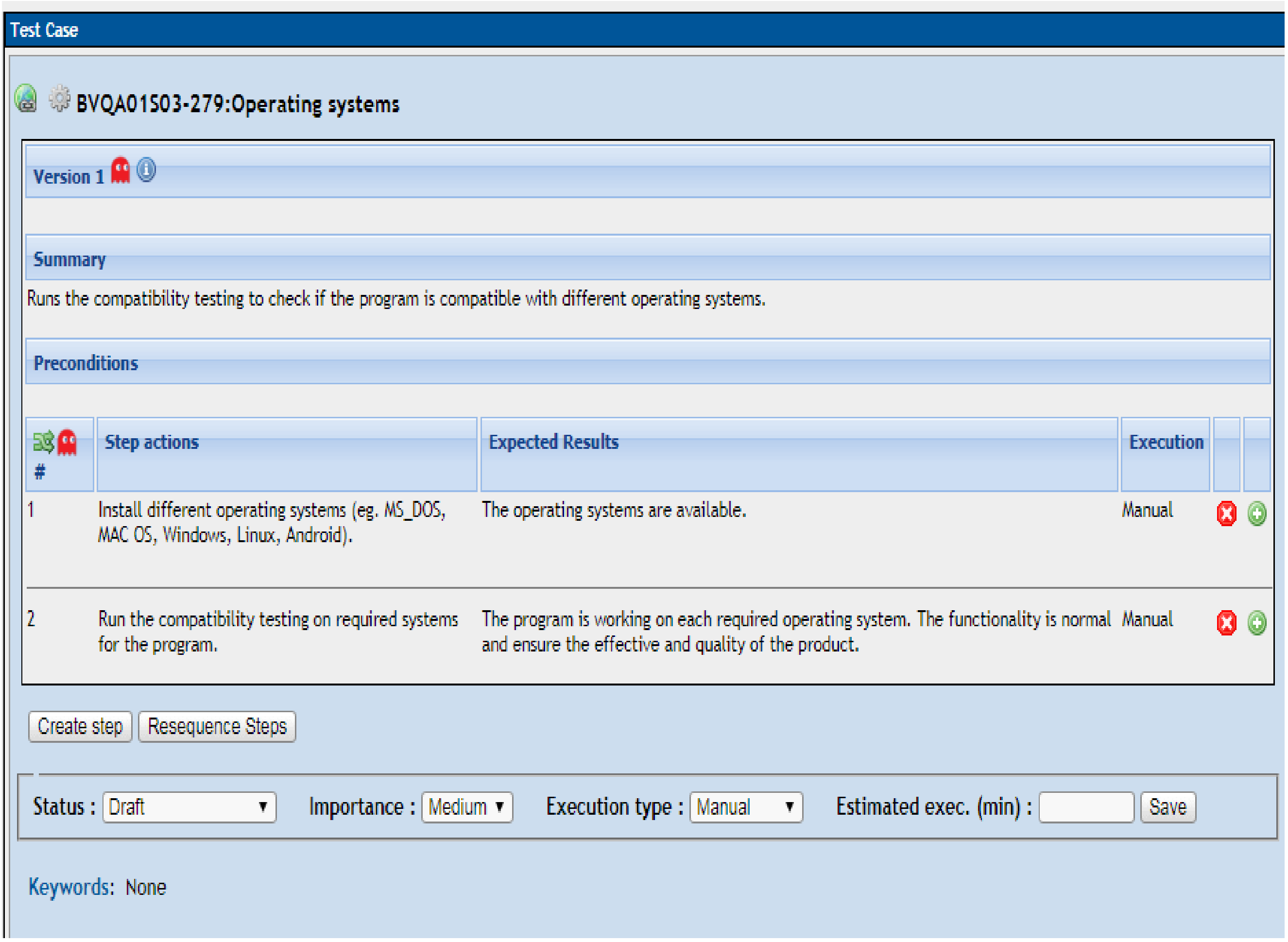
Summary

Check whether the cube's capacity and stability remains the same after long uninterrupted use.

Preconditions

Make sure that the cube resists 50 hours.

#	Step actions	Expected Results	Execution		
1	Spin the cube clockwise and counterclockwise uninterrupted for 50 hours.	The clockwise and counterclockwise moves are allowed for 50 hours. The cube maintains his capacity and stability and remains the same after a long use.	Manual	☒	⊕



Test Case

BVQA01S03-279:Operating systems

Version 1

Summary

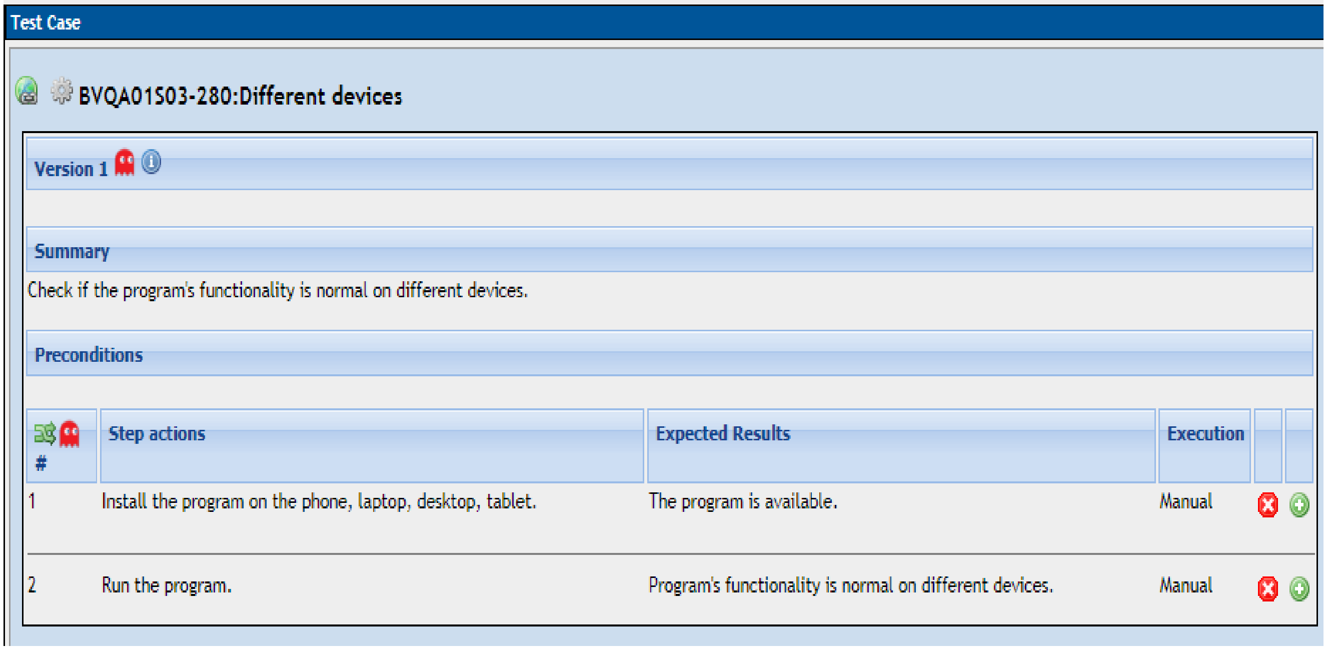
Runs the compatibility testing to check if the program is compatible with different operating systems.

Preconditions

#	Step actions	Expected Results	Execution		
1	Install different operating systems (eg. MS_DOS, MAC OS, Windows, Linux, Android).	The operating systems are available.	Manual	☒	⊕
2	Run the compatibility testing on required systems for the program.	The program is working on each required operating system. The functionality is normal and ensure the effective and quality of the product.	Manual	☒	⊕

Status : Draft ▾ Importance : Medium ▾ Execution type : Manual ▾ Estimated exec. (min) : ▭ Save

Keywords: None



Test Case

BVQA01S03-280:Different devices

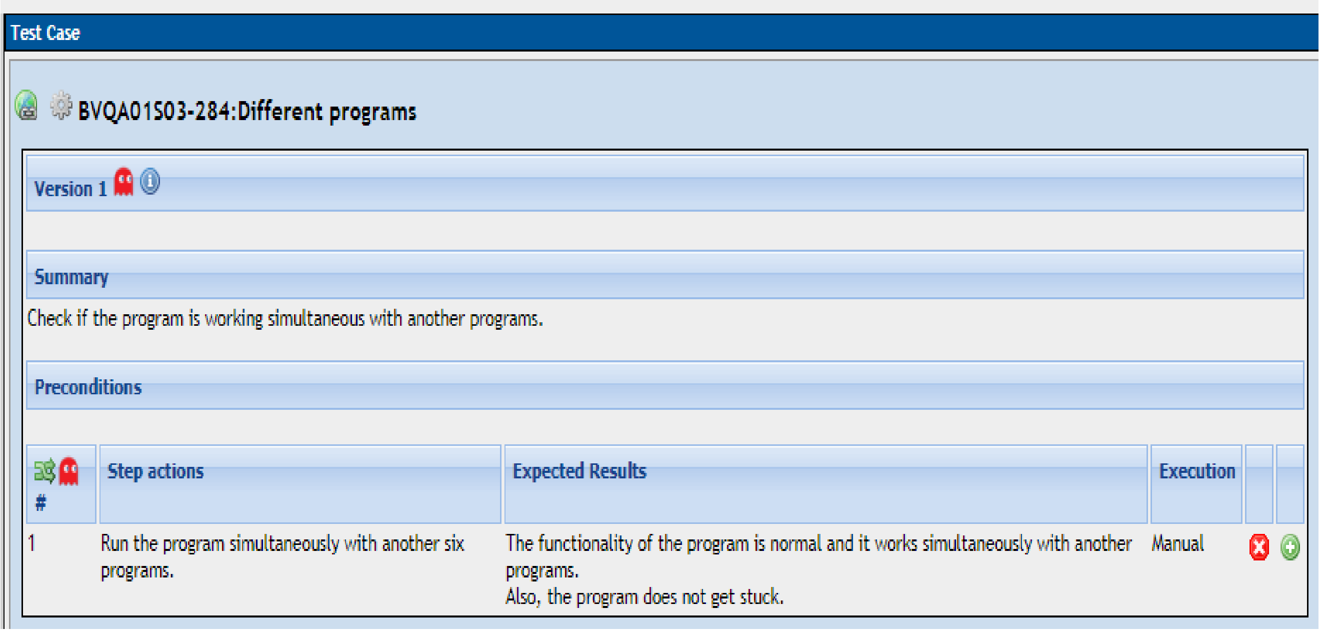
Version 1

Summary

Check if the program's functionality is normal on different devices.

Preconditions

#	Step actions	Expected Results	Execution		
1	Install the program on the phone, laptop, desktop, tablet.	The program is available.	Manual	☒	⊕
2	Run the program.	Program's functionality is normal on different devices.	Manual	☒	⊕



Test Case

BVQA01S03-284:Different programs

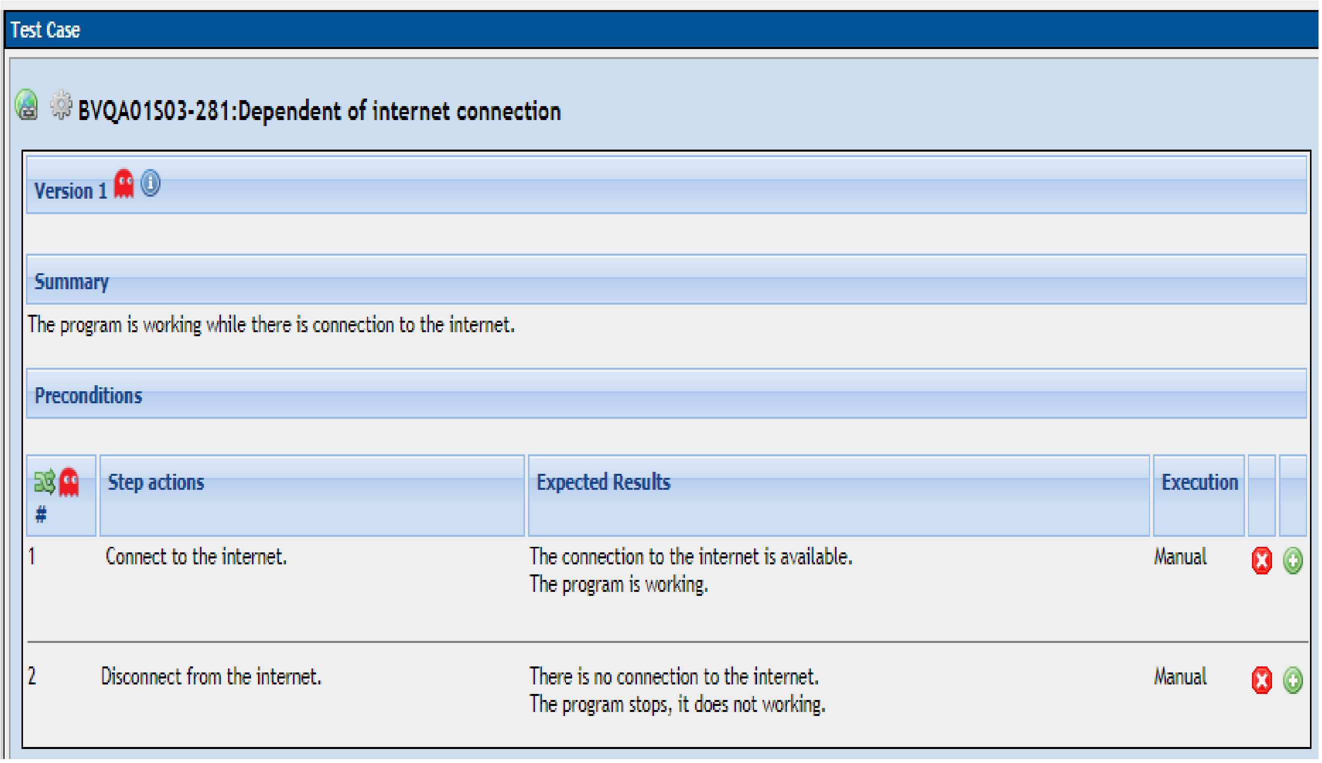
Version 1

Summary

Check if the program is working simultaneous with another programs.

Preconditions

#	Step actions	Expected Results	Execution		
1	Run the program simultaneously with another six programs.	The functionality of the program is normal and it works simultaneously with another programs. Also, the program does not get stuck.	Manual	☒	⊕



Test Case

BVQA01S03-281:Dependent of internet connection

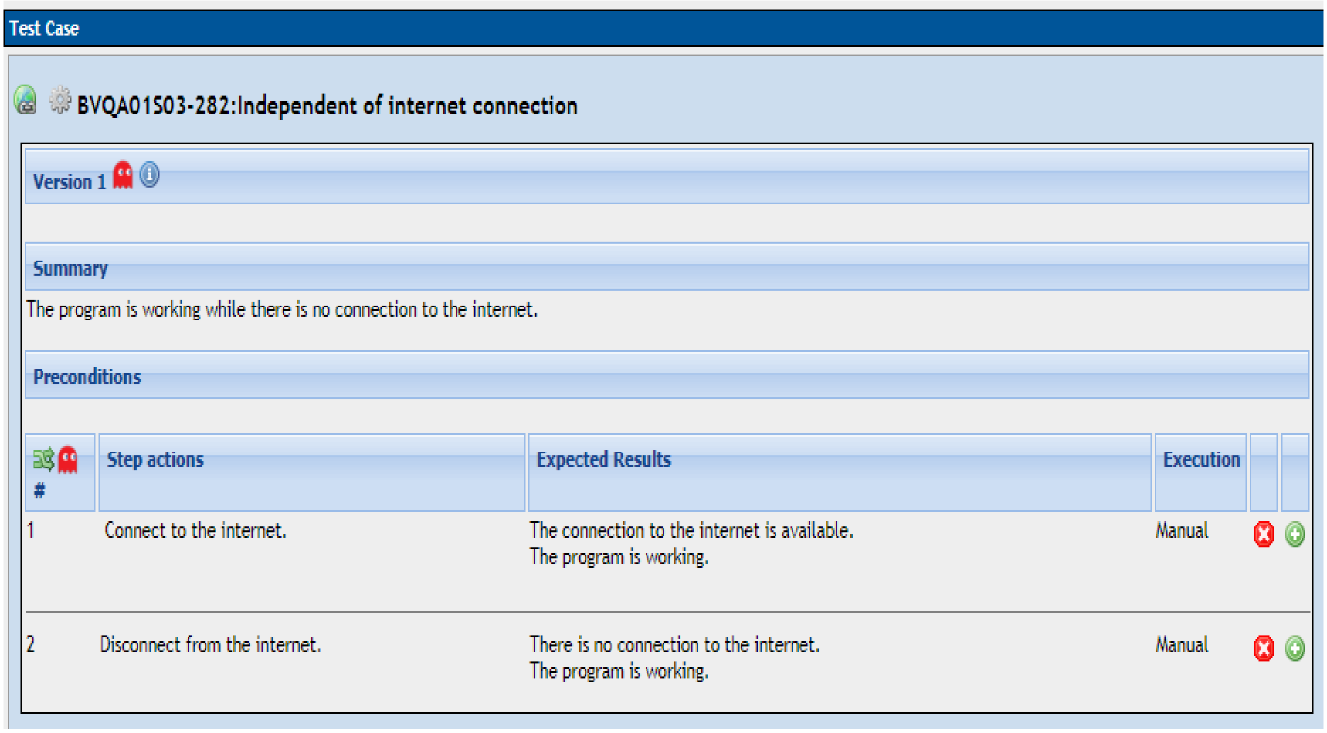
Version 1

Summary

The program is working while there is connection to the internet.

Preconditions

#	Step actions	Expected Results	Execution		
1	Connect to the internet.	The connection to the internet is available. The program is working.	Manual	✖	⊕
2	Disconnect from the internet.	There is no connection to the internet. The program stops, it does not working.	Manual	✖	⊕



Test Case

BVQA01S03-282:Independent of internet connection

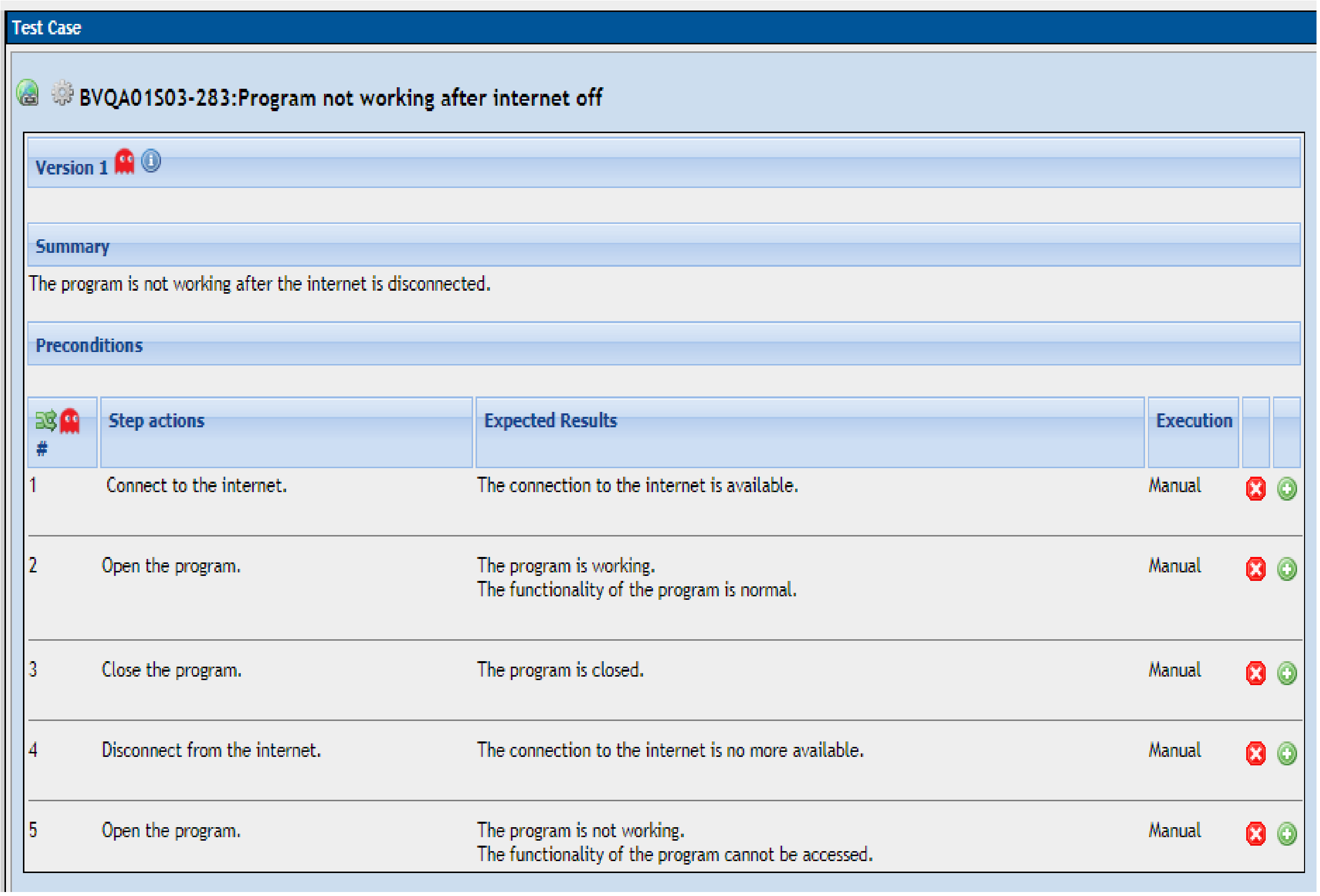
Version 1

Summary

The program is working while there is no connection to the internet.

Preconditions

#	Step actions	Expected Results	Execution		
1	Connect to the internet.	The connection to the internet is available. The program is working.	Manual	✖	⊕
2	Disconnect from the internet.	There is no connection to the internet. The program is working.	Manual	✖	⊕



Test Case

BVQA01S03-283:Program not working after internet off

Version 1

Summary

The program is not working after the internet is disconnected.

Preconditions

#	Step actions	Expected Results	Execution		
1	Connect to the internet.	The connection to the internet is available.	Manual	✖	⊕
2	Open the program.	The program is working. The functionality of the program is normal.	Manual	✖	⊕
3	Close the program.	The program is closed.	Manual	✖	⊕
4	Disconnect from the internet.	The connection to the internet is no more available.	Manual	✖	⊕
5	Open the program.	The program is not working. The functionality of the program cannot be accessed.	Manual	✖	⊕