REPORT FOR THEORY WITH PRACTICAL COMPONENT

FORMAL LANGUAGE AND AUTOMATA – CSE18R252

Submitted by

RAMYA SRUTHI(9917004012)

Under the guidance of

Mr.Muthamil sudar

(Assistant Professor, Department of Computer Science and Engineering)

In partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

of

KALASALINGAM ACADEMY OF RESEARCH AND EDUCATION

(Deemed to be University)

Anand Nagar, Krishnankoil – 626 126

Academic Year Odd Semester (2019-20)

KALASALINGAM ACADEMY OF RESEARCH AND EDUCATION

(Deemed to be University)

Anand Nagar, Krishnankoil – 626 126

BONAFIDE CERTIFICATE

This is to certify that the Theory with Practical Component Report titled **"FORMAL LANGUAGE AND AUTOMATA"** is a bonafide record of the work done by **RAMYASHRUTHI (9917004012)** in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Specialization of the Computer Science and Engineering, during the Academic year Even Semester (2019-2020).

Mr.Muthamil sudar,	**Dr.A.Francis Saviour Devaraj M.E.,Ph.D.,**
Assistant Professor & Supervisor	Professor & Head
Department of CSE	Department of CSE

ACKNOWLEDGEMENT

First and foremost, I wish to thank the **Almighty God** for his grace and benediction to complete this Project work successfully. I would like to convey my special thanks from the bottom of my heart to my dear **Parents** and affectionate **Family members** for their honest support for the completion of this Project work.

I express deep sense of gratitude to “Kalvivallal” Thiru. **T. Kalasalingam** B.com., Founder Chairman, “Ilayavallal” **Dr.K.Sridharan** Ph.D., Chancellor, **Dr.S.ShasiAnanth**, Ph.D., Vice President (Academic) , **Mr.S.Arjun Kalasalingam** M.S., Vice President (Administration) , **Dr.R.Nagaraj** Ph.D., Vice-Chancellor, **Dr.V.Vasudevan** Ph.D., Registrar , **Dr.P.Deepalakshmi** M.E., Ph.D., Dean (School of Computing) . And also a special thanks to **Dr.A.Francis Saviour Devaraj** M.E., Ph.D., Professor & Head, Department of CSE, Kalasalingam Academy of Research and Education for granting the permission and providing necessary facilities to carry out Project work.

I would like to express my special appreciation and profound thanks to my enthusiastic Project Supervisor **Mr.K.MUTHAMIL SUDAR** M.E., Assistant Professor/ CSE of Kalasalingam Academy of Research and Education [KARE] for his inspiring guidance, constant encouragement with my work during all stages. I am extremely glad that I had a chance to do my Project under my Guide, who truly practices and appreciates deep thinking. I will be forever indebted to my Guide for all the time he has spent with me in discussions. And during the most difficult times when writing this report, he gave me the moral support and the freedom I needed to move on.

Besides my Project guide, I would like to thank the rest of Class committee members and all faculty members and Non-Teaching staff for their insightful comments and encouragement.Finally, but by no means least, thanks go to all my school and college teachers, well wishers, friends for almost unbelievable support

INDEX

EX.NO 1 DERMINISTIC FINITE AUTOMATON

AIM:

To design a DFA to accept the given languages over the alphabet {a,b}.

ALGORITHM:

- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Stop the program.

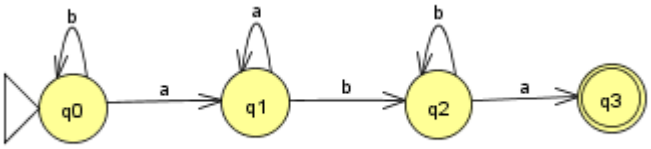
PROCEDURE:

- Start the JFLAP8_beta.jar file.
- Click Finite Automaton button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

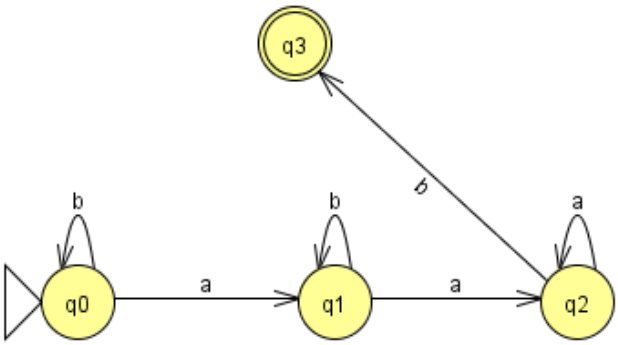
SCREENSHOTS:

.

1. Set of all substrings with aba.



2. Set of all strings starting with two a's and ending with b.



RESULT:

Thus, the given DFA has been successfully designed.

EX NO: 2 **NON - DETERMINISTIC FINITE AUTOMATION**

AIM:

To check the input string is accepted or not.

ALGORITHM:

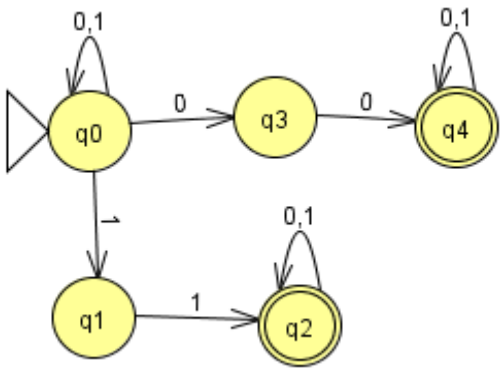
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Stop the program.

PROCEDURE:

- Start the JFLAP8_beta.jar file.
- Click Finite Automaton button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

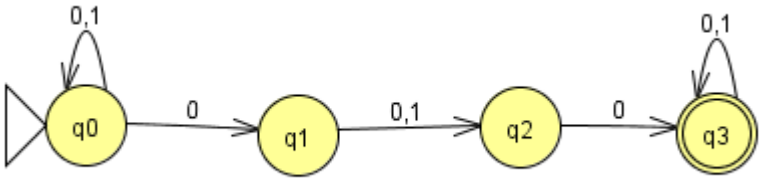
SCREENSHOTS:

1.



Input	Result
01001	Reject
1100101	Reject
001101	Reject
11	Accept
100	Reject
000	Reject

2.



Input	Result
000	Reject
010	Reject
10101	Reject
00000	Reject

RESULT:

Thus, the given NFA has been successfully designed.

EX NO: 3 **ε - NON - DETERMINISTIC FINITE AUTOMATON**

AIM:

To check the input string for ε -nfa and to accept the given strings

ALGORITHM:

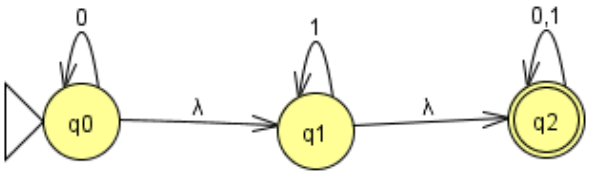
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Stop the program.

PROCEDURE:

- Start the JFLAP8_beta.jar file.
- Click Finite Automaton button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

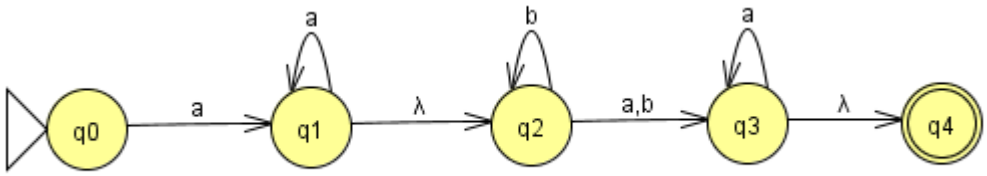
SCREENSHOTS:

1.



Input	Result
0011	Accept
1100	Reject
1010	Reject

2.



Input	Result
abab	Reject
aabba	Reject
ababab	Reject

RESULT:

Thus, the given NFA has been successfully designed.

EX NO: 4 **CONVERT AUTOMATON TO DFA**

AIM:

To convert the given automaton to DFA.

ALGORITHM:

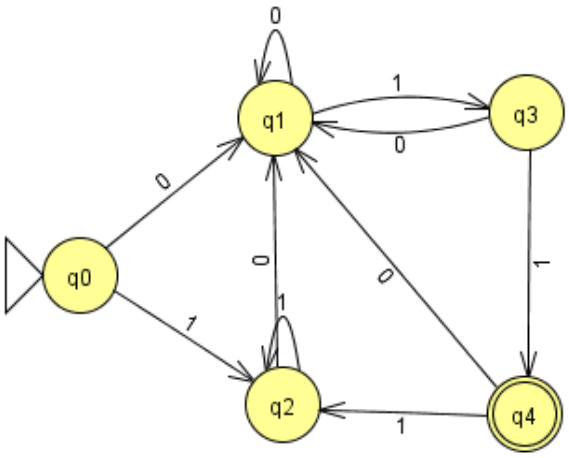
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Click convert to DFA button.
- Stop the program.

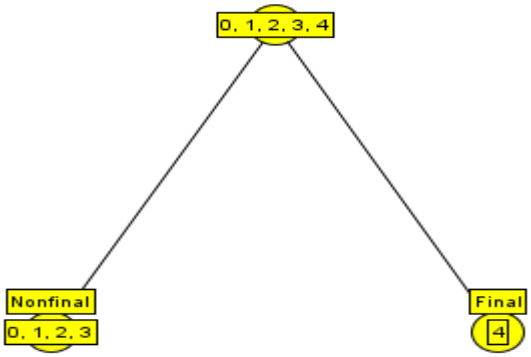
PROCEDURE:

- Start the JFLAP8_beta.jar file.
- Click Finite Automaton button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

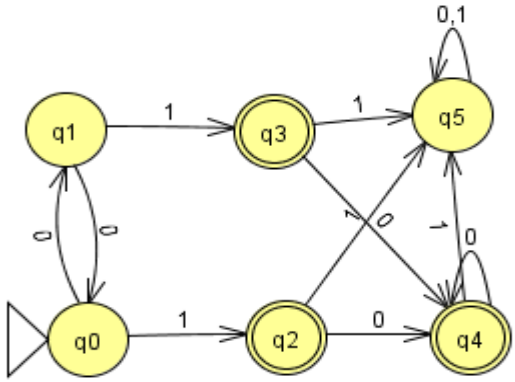
SCREENSHOTS:

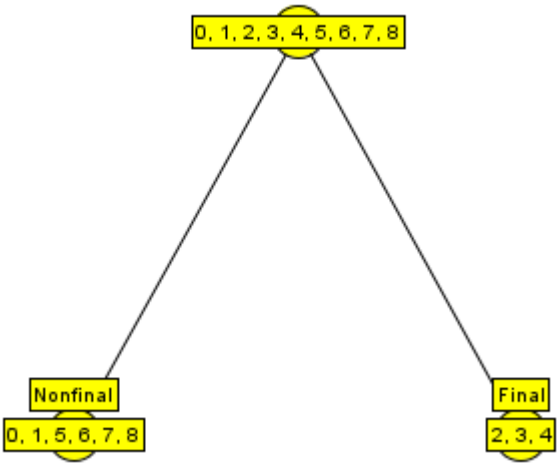
1.


0
1
q1
q3
0
0
q0
0
0
1
1
1
q2
q4
1


0, 1, 2, 3, 4
Nonfinal
0, 1, 2, 3
Final
4

2.


0,1
q1
1
q3
1
q5
0
0
1
0
1
0
q0
1
q2
0
q4



RESULT:

Thus, the given automaton has been successfully converted to DFA.

EX NO: 5 **CONVERT NFA TO DFA**

AIM:

To design DFA from the given NFA transitions.

ALGORITHM:

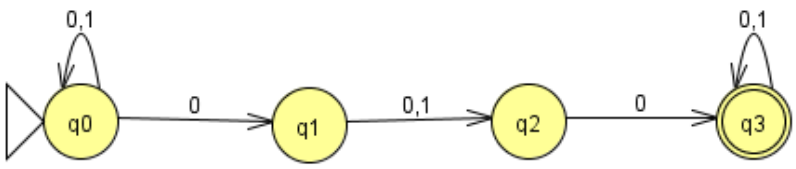
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Click convert to DFA button.
- Stop the program.

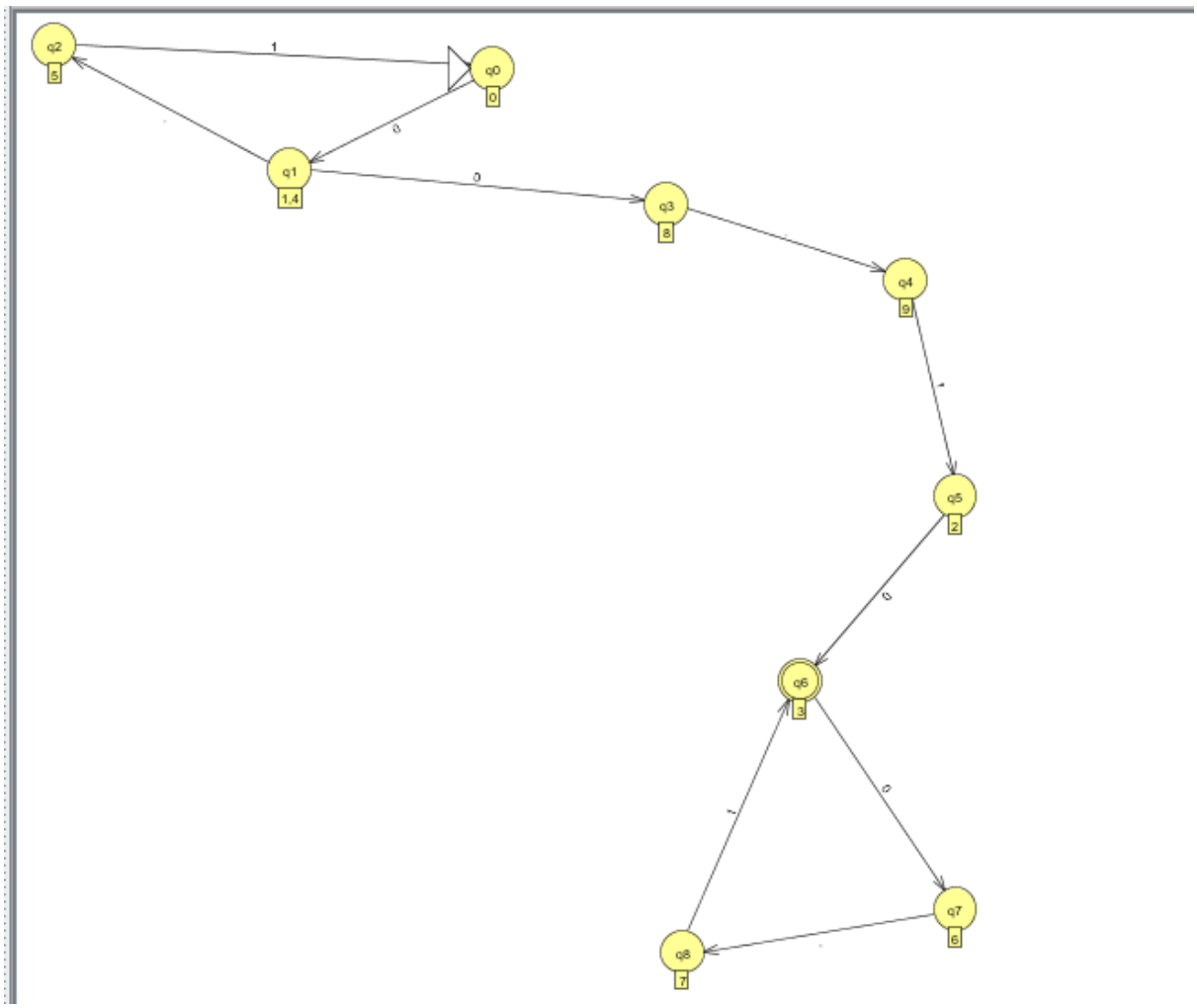
PROCEDURE:

- Start the JFLAP8_beta.jar file.
- Click Finite Automaton button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

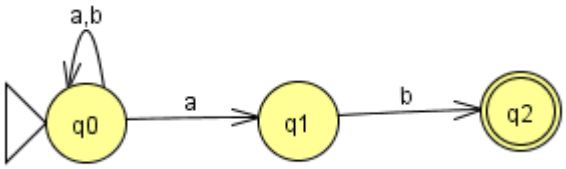
SCREENSHOTS:

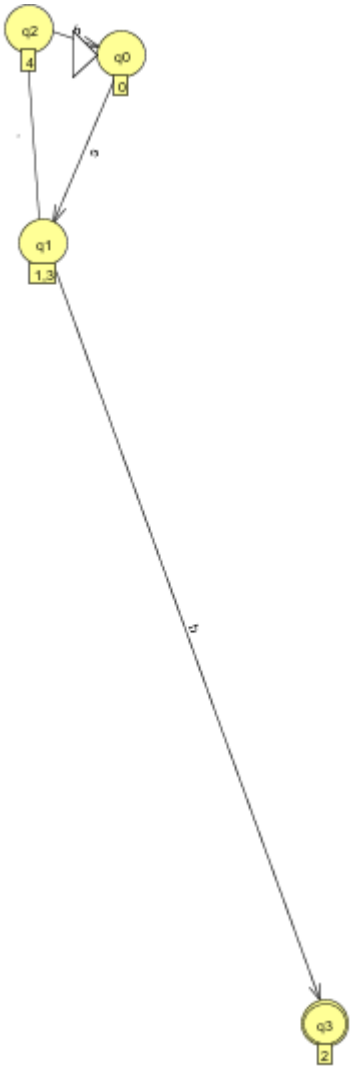
1.





2.





RESULT:

Thus, the DFA has been designed successfully from the given NFA transition.

EX NO: 6 **CONVERT ε - NFA TO DF**

AIM:

To design DFA from the given ε - NFA transition table.

ALGORITHM:

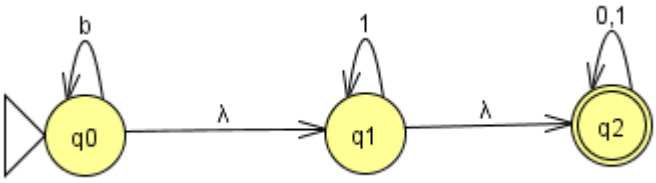
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Click convert to DFA button.
- Stop the program.

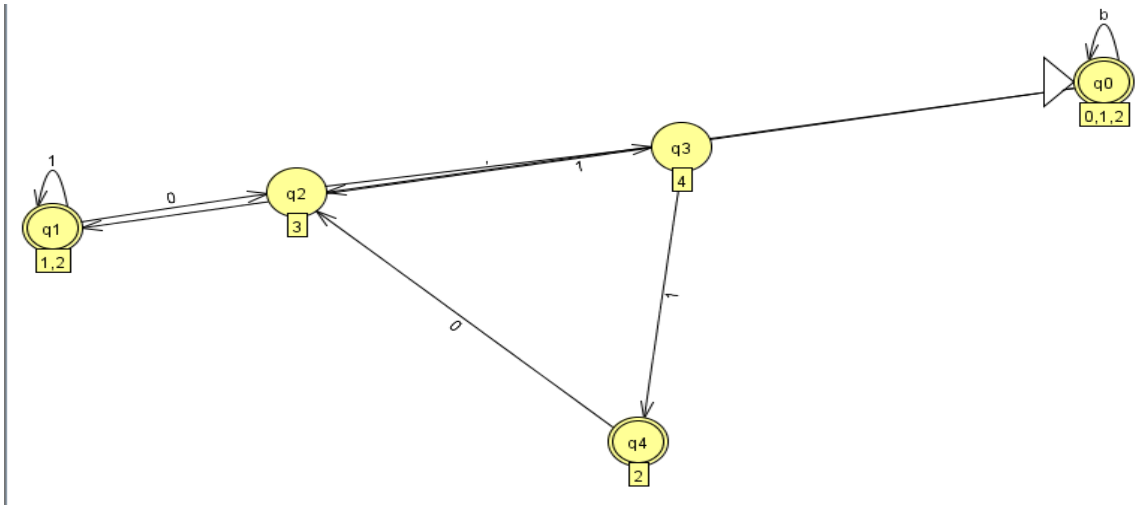
PROCEDURE:

- Start the JFLAP8_beta.jar file.
- Click Finite Automaton button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

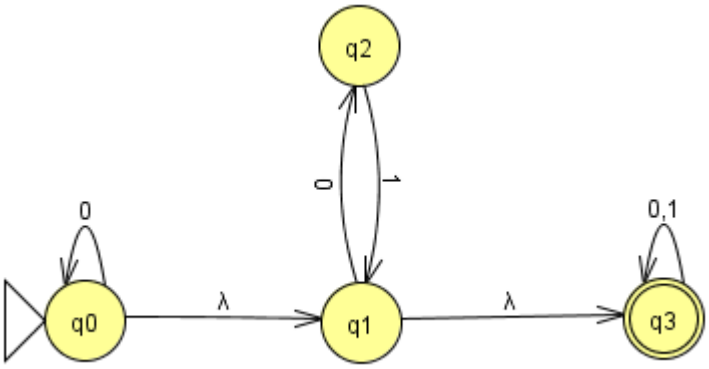
SCREENSHOTS:

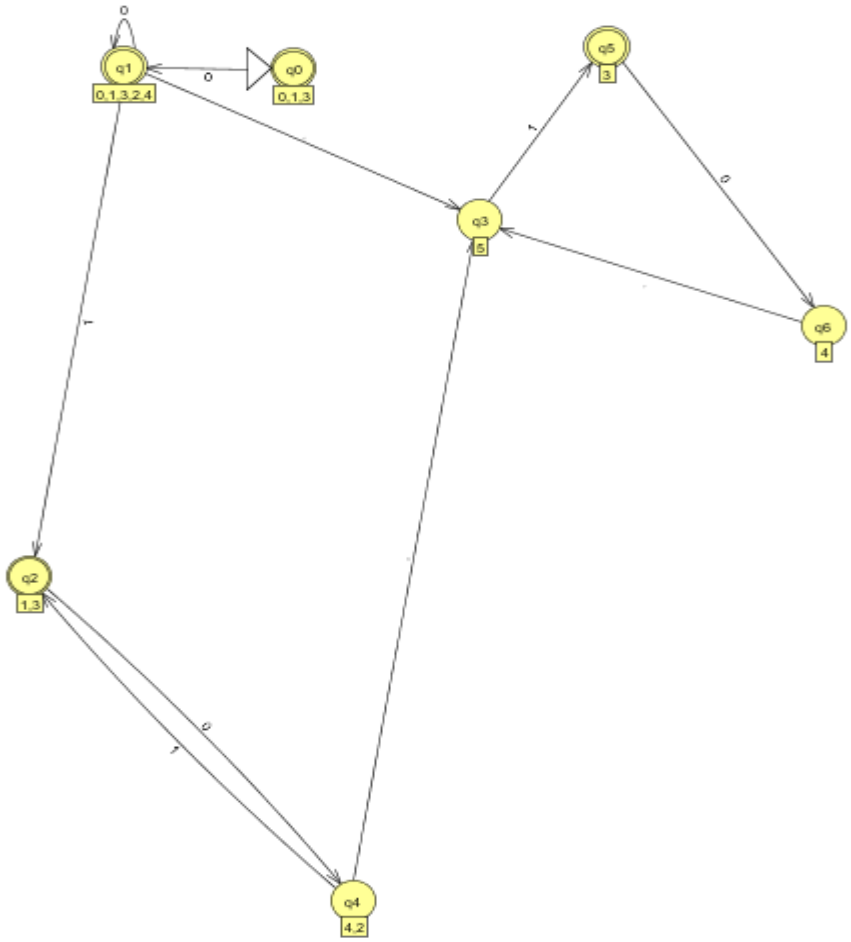
1.


b
1
0,1
q0
λ
q1
λ
q2


b
q0
0,1,2
q3
4
1
0
q2
3
1
1,2
q1
1
0
q4
2

2


q2
0
1
0
0,1
q0
λ
q1
λ
q3



RESULT:

Thus, the DFA has been successfully designed from the given ε - NFA.

EX NO: 7 **REGULAR EXPRESSION TO ε - NFA**

AIM:

To construct ε - NFA from the given regular expression.

ALGORITHM:

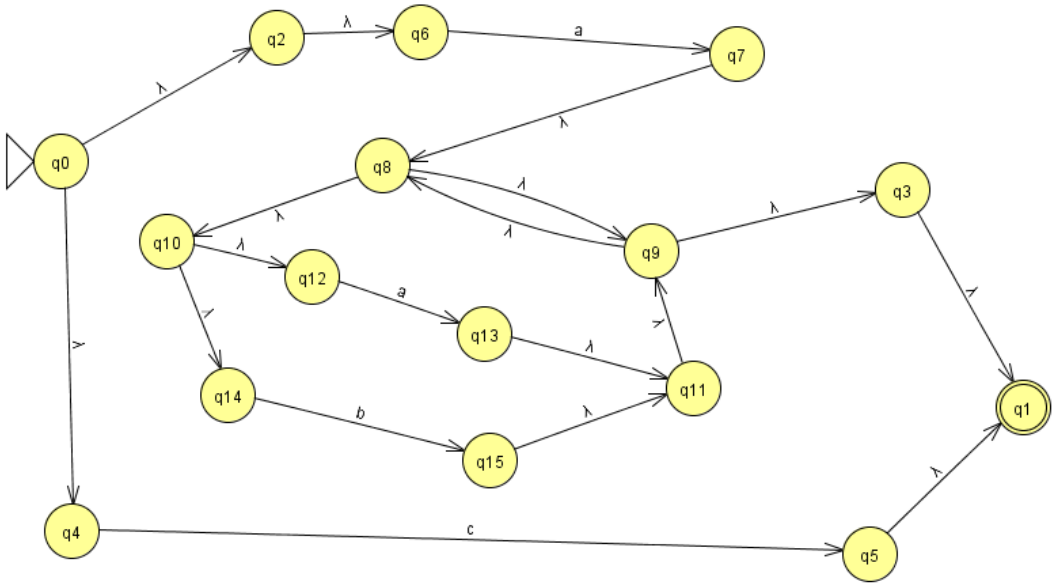
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Stop the program.

PROCEDURE:

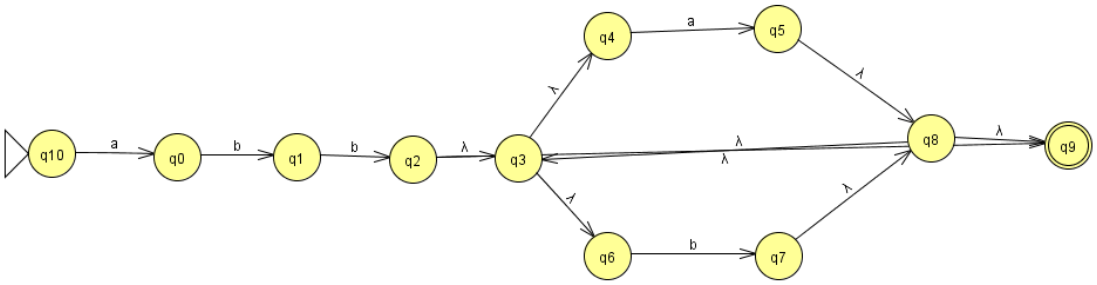
- Start the JFLAP8_beta.jar file.
- Click Regular Expression button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

SCREENSHOTS:

1. a(a+b)*+c



2. abb.(a/b)*



RESULT:

Thus, the ε - NFA has been designed successfully from the given regular express

EX NO: 8

PUSHDOWN AUTOMATON

AIM:

To construct a pushdown automata.

ALGORITHM:

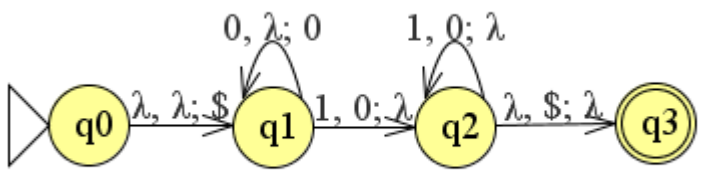
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Stop the program.

PROCEDURE:

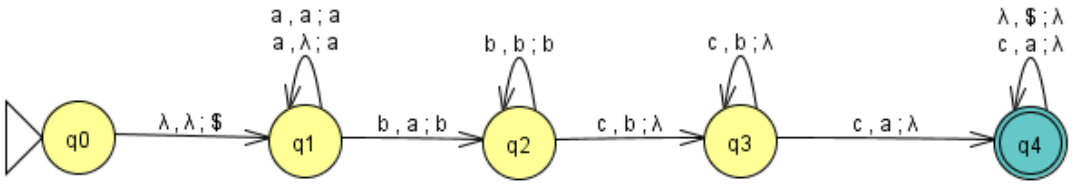
- Start the JFLAP8_beta.jar file.
- Click Pushdown Automaton button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

SCREENSHOTS:

PDA that accepts L = $\{0^n 1^n \mid n \geq 0\}$



PDA that accepts L={anbncm/m,n>=1}



RESULT:

Thus, the given PDA is constructed successfully.

EX NO: 9 **PUMPING LEMMA**

AIM:

To draw the transition diagram of a NFA from given transition table.

ALGORITHM:

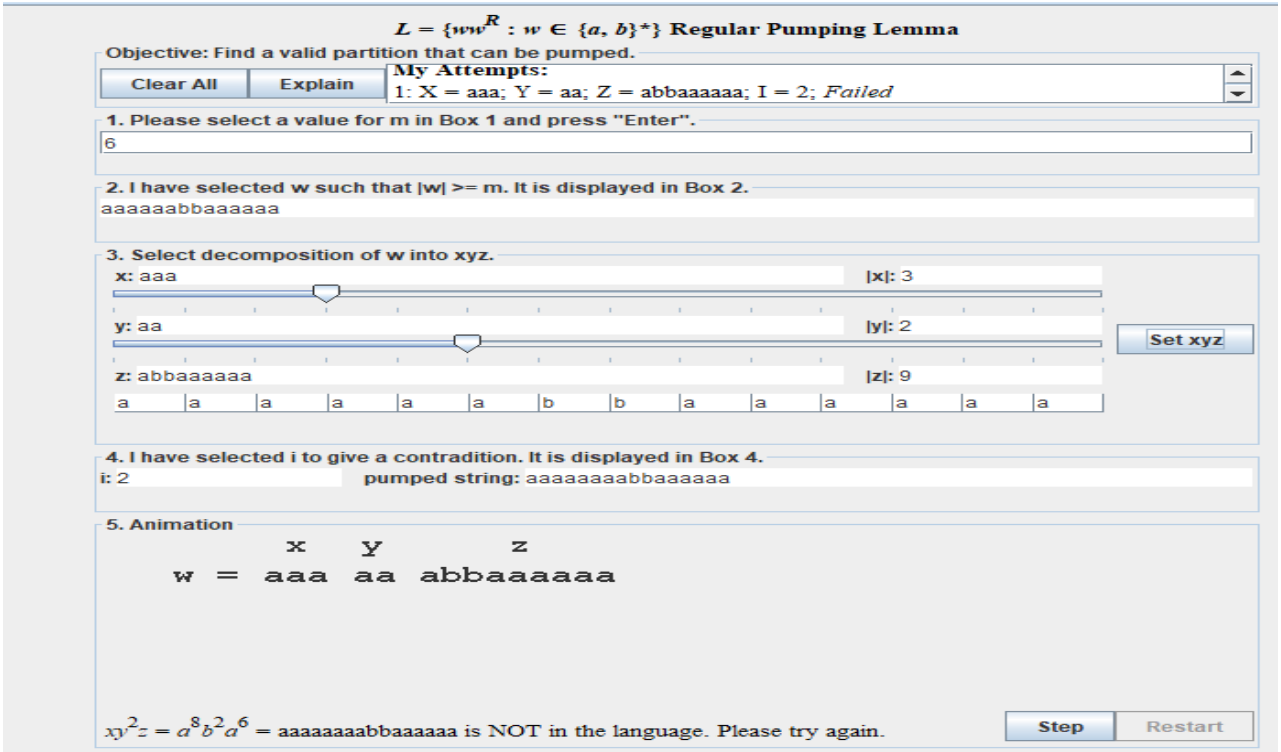
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Stop the program.

PROCEDURE:

- Start the JFLAP8_beta.jar file.
- Click Pumping lemma button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

SCREENSHOTS:

1.



RESULT:

Thus, pumping lemma for given language is proved successfully.

EX NO: 10 **TURING MACHINE**

AIM:

To construct a Turing Machine that accepts $0^n1^n \mid n > 1$**.**

.

ALGORITHM:

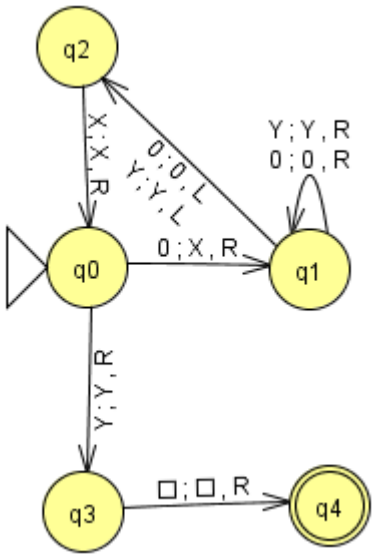
- Start the JFLAP8_beta.jar file.
- Add required states.
- Make required transitions.
- Denote starting and final state.
- Save the project.
- Stop the program.

PROCEDURE:

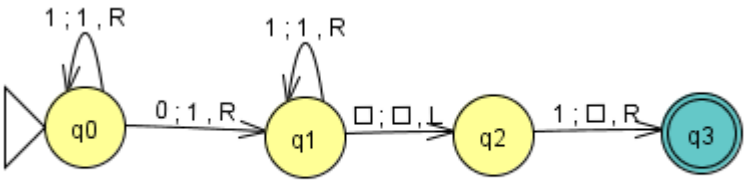
- Start the JFLAP8_beta.jar file.
- Click Turing machine button.
- Add required states to the Automaton editor.
- Make required transitions as per the problem.
- Denote starting and final state.
- Save the project.
- Test the automaton for few input values.
- Stop the program.

SCREENSHOTS:

Turing machine for L={0n1m/n>=1}



Turing machine for a+b



RESULT:

Thus, the given Turing Machine is designed successfully.