Kingdom of Saudi Arabia Ministry of Education

Imam Abdulrahman bin Faisal University

Computer Science department

College of Science and Humanities

جامعة الإمام عبد الرحمن بن فيصل
IMAM ABDULRAHMAN BIN FAISAL UNIVERSITY

Language Theory & Finite Automata Project

Pharmacy Vending Machine

Milestone 3

CIS321 – Language Theory & Finite Automata Term 3

Group 3

Supervisor: Dr.Azza Ahmed Abdo Ali

Students name:

Student Name	ID
Amal Mohammed Alotaibi	2200001746
Asma Zaher Alshehri	2200000484
Manar Fahad Alenzi	2200006162
Shahad Emad Aljiaan	2200003861
Wadha Nayef Alsheddi	2200003246

1.Introduction

Pharmacy expansion has nearly become a necessity to serve people's emergent needs in places like airports, campuses, malls, and restaurants. People’s need for pharmacies in different places can be served by building a pharmacy vending machine.

Vending machines are electronic machines that can be found anywhere all around the world to serve various types of products which can serve people’s needs, for instance, ice cream, snacks, and train tickets. The mechanism of its work is to choose the desired products first and the process for payment. Pharmacy vending machines will be designed using finite automata. Finite State Automata are components of informatics that function like digital computers. Inputs are received, outputs are produced, temporary storage can be provided, and decisions may be made in the process of converting input to output. In essence, an automaton consists of a finite number of states, each containing information about a previous input [1].

1.1 Scope

In this project, we propose to design a vending machine using Finite State Machines, which will be presented in markets, restaurants, universities, and other places where people gather. The vending machine will sell medical products such as wound plasters, cotton, headache pills, and others for use when the user is suffering from an injury, wounds, or feeling unwell and cannot reach a pharmacy or hospital. Also, products related to skin care and oral care. This project involves the construction of a finite automata that contains three main sectors which are: medicines, skin care and oral care. The vending machine simulator can process products sales transactions by taking the selection of products by user as input and printing the amount of money is an output. The vending

machine provides payment methods to make it easier for the user to pay. However, if the user paid with cash. Only money from type 5 SR, 10 SR and 15SR is accepted, otherwise the process will be rejected.

2. Vending machine description

This section mainly discusses the benefits of the development of pharmacy vending machines besides the basic operation which will be performed.

2.1 Benefits

- Ease of its implementation.
- Serves people's emergent needs.
- Can be placed anywhere.
-Accepting two payment methods.
-Minimum management cost

Vending Machine States Table

Description	State
Select options	Q0
Select the skin care section	Q1
Select the medicine section	Q2
Select nail polish product	Q4
Select makeup remover product	Q5
Select sunscreen product	Q6
Select Panadol product	Q7
Select plaster product	Q8
Select alcohol pads product	Q9
Check availability for alcohol pads	Q19
Check availability for plaster	Q18
Check availability for Panadol	Q17
Check availability for sunscreen	Q15
Check availability for nail polish	Q13
Check availability for makeup remover	Q14
Product is not available.	Q32
I'm Select payment option	Q23
Product purchase successful	Q3
Product purchase successful	Q3
Product purchase not successful	Q34
Select card payment option	Q26
Select cash payment option	Q27
Select payment option	Q24
Product purchase successful	Q3
Product purchase successful	Q3
Product purchase not successful	Q37
Select card payment option	Q28

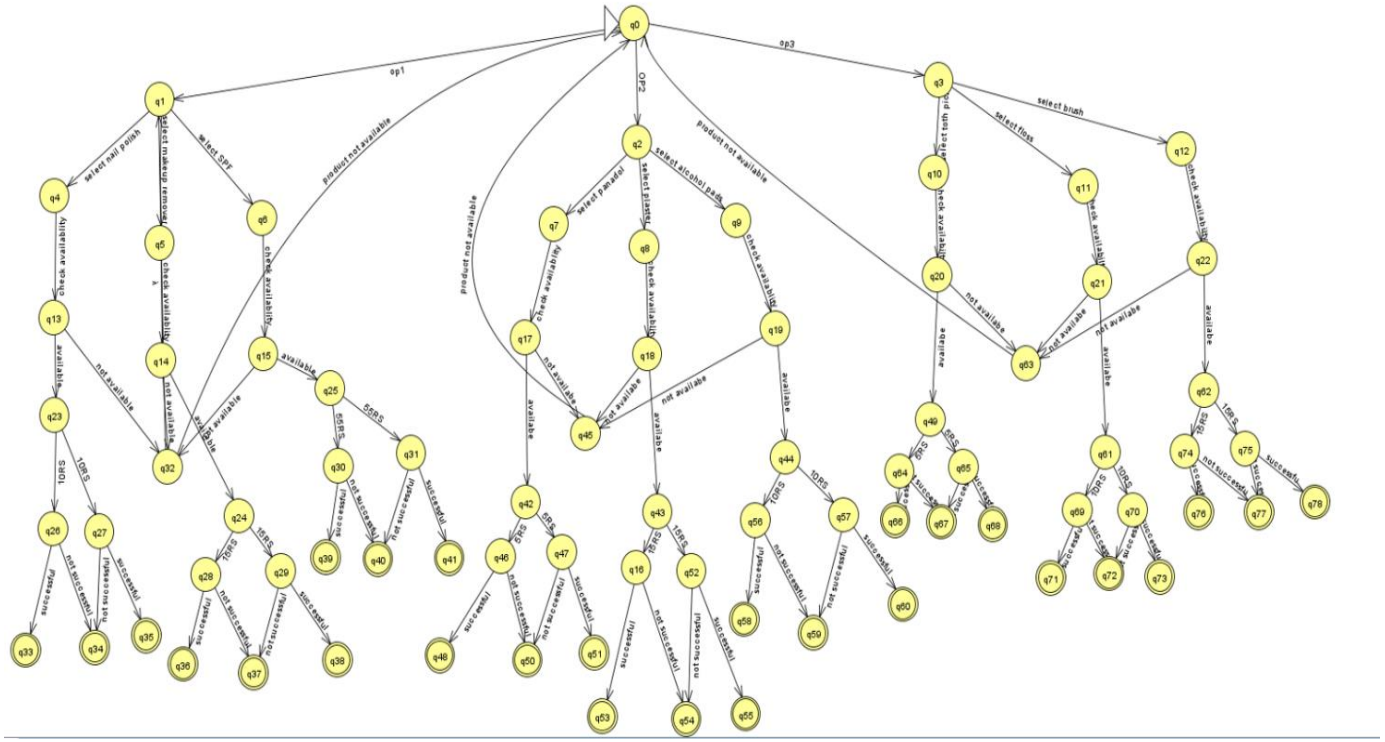
Select cash payment option	Q29
Select payment option	Q25
Product purchase successful	Q3
Product purchase successful	Q3
Product purchase not successful	Q40
Select card payment option	Q30
Select cash payment option	Q31
Product is not available.	Q45
Select payment option	Q42
Product purchase successful	Q3
Product purchase successful	Q3
Product purchase not successful	Q50
Select card payment option	Q46
Select cash payment option	Q47
Select payment option	Q43
Product purchase successful	Q3
Product purchase successful	Q3
Product purchase not successful	Q54
Select card payment option	Q16
Select cash payment option	Q52
Select payment option	Q44
Product purchase successful	Q3
Product purchase successful	Q3
Product purchase not successful	Q59
Select card payment option	Q56
Select cash payment option	Q57

Vending Machine Strings Table

String	abbreviation	Description
Op1	Op1	The skincare sector is selected by the user.
Op2	Op2	The medicine sector is selected by the user.
Select Nail polish	SelNP	The user selects Nail polish.
Select Makeup remover	SelMR	The user selects Makeup remover.
Select SPF	SelSPF	The user selects SPF.
Select Panadol	SelPn	The user selects Panadol.
Select Plaster	SelPl	The user selects Plaster.
Select Alcohol pads	SelC	The user selects Alcohol pads.
Check availability	CheckAv	To check whether the product is inside the machine or not.
“NO” Not available	NotAv	The product is not inside the machine.
Product not available	Pnot	The user will go back to the start state if the product was not inside the machine which is “q0”
“YES” Available	Av	The product is inside the machine.
10RS	10	The cost of the selected product
15RS	15	The cost of the selected product
5RS	5	The cost of the selected product
Successful	Succ	Successfully completed, the selected product has been delivered to the user.
Not Successful	Nots	User has not received the selected product due to an unsuccessful completion.

2.4 Finite Automata

Continuing in the previous section, we have a drawing of this finite automata after all that has been explained, all the states and strings that are going to be involved.



2.5 Scenario

Starting from the initial state (Q0) The user has 3 sector options to choose from:

1. (Q1) Makeup Section
2. (Q2) Pharmacy Section
3. (Q3) Oral care Section

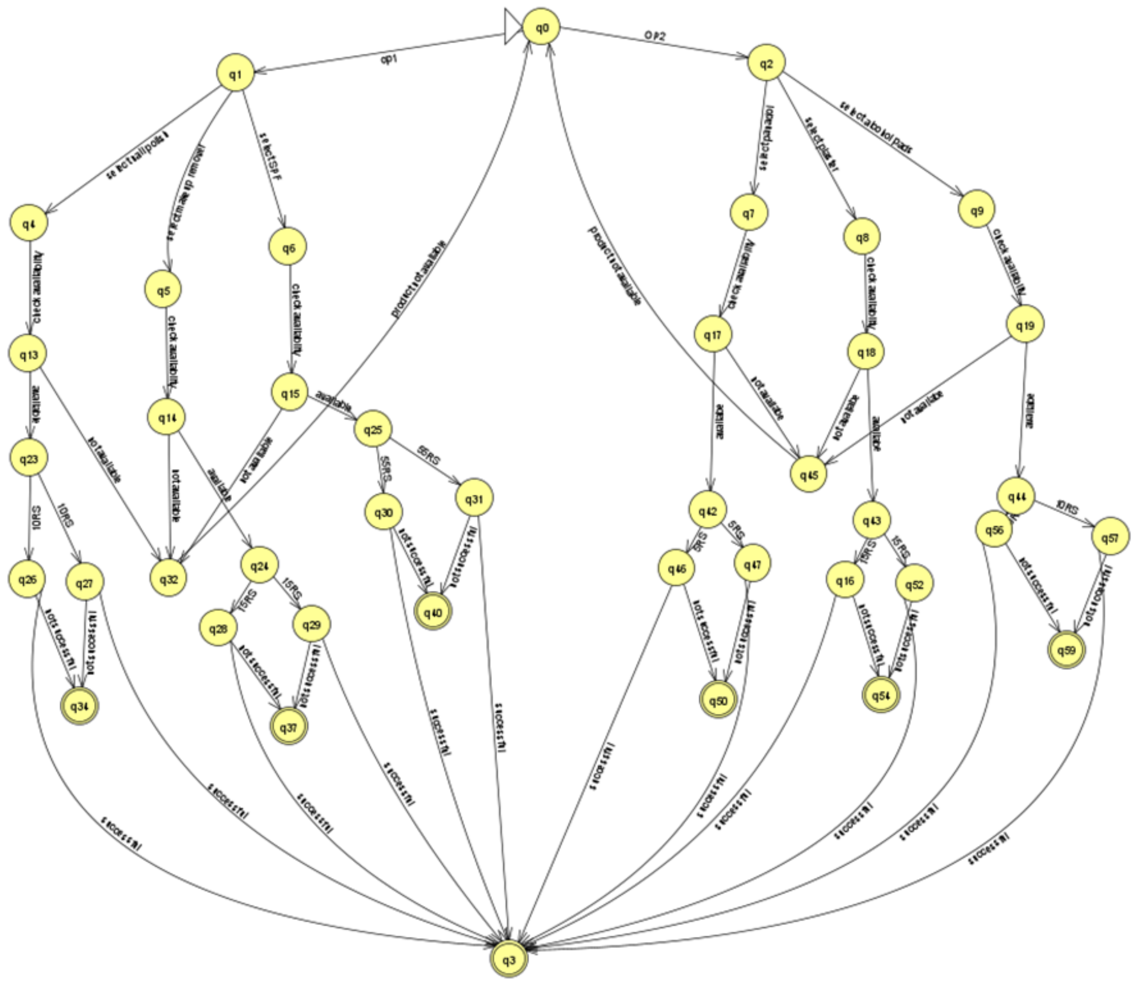
Each section has exactly 3 products with different prices, if the user selects (Q1) it will take them to the makeup section (Q1) which has 3 products to choose from:

(Q4) nail polish, (Q5) makeup remover, (Q6) SPF. So, when the user selects (Q6) for example, the machine will check first the availability of the product in state (Q15). The availability has two cases:

- If the product is available, the user will move to state (Q25). Then the user will have two payment methods either by credit card (Q30) or cash (Q31). The process of the two methods is the same for the two states. So, if user selects pay by credit card (Q30) the process is either successful (Q39) which is the final state, or not successful which is state (Q40) that is also a final state.
- If the product is not available, the state of the product will go back to the initial state and start the machine process.

The scenario applies the same for each product in the automata.

NFA



NFA State Transition Table

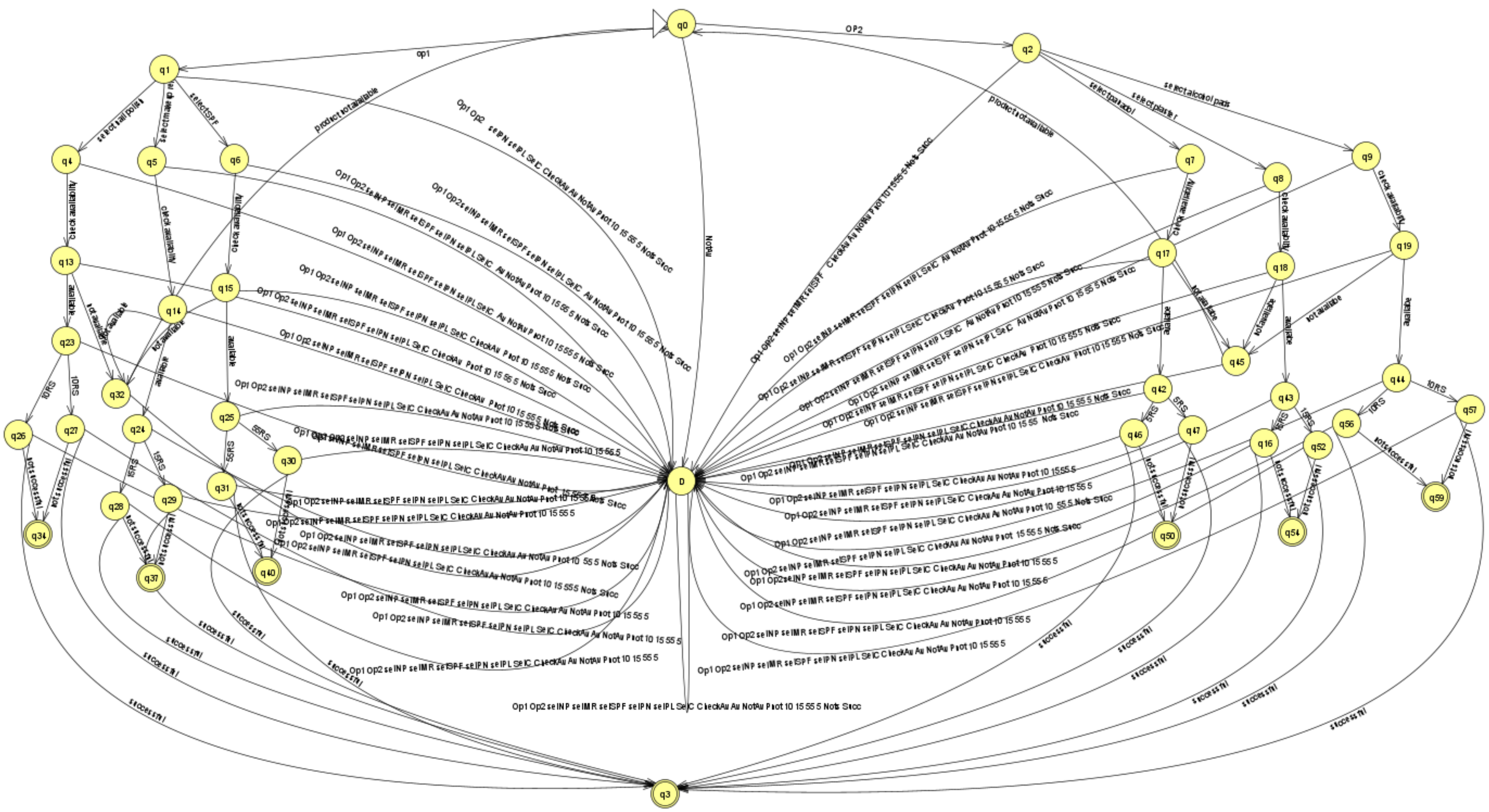
σ	Op1	Op2	SelNP	SelMR	SelSPF	SelPn	SelPl	SelC	CheckAv	Av	NotAv	Pnot	10	15	55	5	Nots	Succ
Q0	Q1	Q2	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q1	Φ	Φ	Q4	Q5	Q6	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q2	Φ	Φ	Φ	Φ	Φ	Q7	Q8	Q9	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q4	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q13	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q5	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q14	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q6	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q15	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q7	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q17	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q8	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q18	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q9	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q19	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q13	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q23	Q32	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q14	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q24	Q32	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q15	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q25	Q32	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q16	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q54	Q3
Q17	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q42	Q45	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q18	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q43	Q45	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q19	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q44	Q45	Φ	Φ	Φ	Φ	Φ	Φ	Φ
Q23	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q26,Q27	Φ	Φ	Φ	Φ	Φ
Q24	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q28,Q29	Φ	Φ	Φ	Φ
Q25	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q30,Q31	Φ	Φ	Φ
Q26	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q34	Q3
Q27	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q34	Q3
Q28	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q37	Q3
Q29	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q37	Q3
Q30	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q40	Q3
Q31	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q40	Q3
Q32	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q0	Φ	Φ	Φ	Φ	Φ	Φ
Q42	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q46,Q47	Φ	Φ
Q43	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q16,Q52	Φ	Φ	Φ	Φ
Q44	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q56,Q57	Φ	Φ	Φ	Φ	Φ
Q45	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q0	Φ	Φ	Φ	Φ	Φ	Φ
Q46	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q50	Q3
Q47	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q50	Q3
Q52	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q54	Q3
Q56	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q59	Q3
Q57	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Φ	Q59	Q3

DFA State Transition Table

σ	Op 1	Op 2	selNP	SelMR	SelSP F	SelP n	SelP l	Sel C	CheckAv	Av	NotAv	Pnot	10	15	55	5	Nots	Succ
Q0	Q1	Q2	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Q1	D	D	Q4	Q5	Q6	D	D	D	D	D	D	D	D	D	D	D	D	D
Q2	D	D	D	D	D	Q7	Q8	Q9	D	D	D	D	D	D	D	D	D	D
Q4	D	D	D	D	D	D	D	D	Q13	D	D	D	D	D	D	D	D	D
Q5	D	D	D	D	D	D	D	D	Q14	D	D	D	D	D	D	D	D	D
Q6	D	D	D	D	D	D	D	D	Q15	D	D	D	D	D	D	D	D	D
Q7	D	D	D	D	D	D	D	D	Q17	D	D	D	D	D	D	D	D	D
Q8	D	D	D	D	D	D	D	D	Q18	D	D	D	D	D	D	D	D	D
Q9	D	D	D	D	D	D	D	D	Q19	D	D	D	D	D	D	D	D	D
Q13	D	D	D	D	D	D	D	D	D	Q23	Q32	D	D	D	D	D	D	D
Q14	D	D	D	D	D	D	D	D	D	Q24	Q32	D	D	D	D	D	D	D
Q15	D	D	D	D	D	D	D	D	D	Q25	Q32	D	D	D	D	D	D	D
Q16	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q54	Q3
Q17	D	D	D	D	D	D	D	D	D	Q42	Q45	D	D	D	D	D	D	D
Q18	D	D	D	D	D	D	D	D	D	Q43	Q45	D	D	D	D	D	D	D
Q19	D	D	D	D	D	D	D	D	D	Q44	Q45	D	D	D	D	D	D	D
Q23	D	D	D	D	D	D	D	D	D	D	D	D	Q26,Q27	D	D	D	D	D
Q24	D	D	D	D	D	D	D	D	D	D	D	D	D	Q28,Q29	D	D	D	D
Q25	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Q26	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q34	Q3
Q27	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q34	Q3
Q28	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q37	Q3

Q29	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q37	Q3
Q30	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q40	Q3
Q31	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q40	Q3
Q32	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Q42	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q46, Q47	D	D
Q43	D	D	D	D	D	D	D	D	D	D	D	D	D	Q16,Q52	D	D	D	D
Q44	D	D	D	D	D	D	D	D	D	D	D	D	Q56,Q57	D	D	D	D	D
Q45	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D
Q46	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q50	Q3
Q47	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q50	Q3
Q52	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q54	Q3
Q56	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q59	Q3
Q57	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	Q59	Q3
D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

Build DFA


q0
OP2
op1
q1
q2
q3
q4
q5
q6
q7
q8
q9
q13
q14
q15
q16
q17
q18
q19
q23
q24
q25
q26
q27
q28
q29
q30
q31
q32
q34
q37
q40
q42
q43
q44
q45
q46
q47
q50
q52
q54
q56
q57
q59
D

Production Rules for NFA

q0 → Op1 q1 | Op2 q2

q1 → SelNP q4 | SelMR q5 | SelSPF q6

q2 → SelPn q7 | SelPl q8 | SelC q9

q4 → CheckAv q13

q5 → CheckAv q14

q6 → CheckAv q15

q7 → CheckAv q17

q8 → CheckAv q18

q9 → CheckAv q19

q13 → Av q23 | NotAv q32

q14 → Av q24 | NotAv q32

q15 → Av q25 | NotAv q32

q16 → Nots q54 | Succ q3

q17 → Av q42 | NotAv q45

q18 → Av q43 | NotAv q45

q19 → Av q44 | NotAv q45

q23 → 10 q26,q27

q24 → 15 q28,q29

q25 → 55 q30,q31

q26 → Nots q34 | Succ q3

q27→ Nots q34 | Succ q3

q28→ Nots q37 | Succ q3

q29 → Nots q37 | Succ q3

q30→ Nots q40 | Succ q3

q31→ Nots q40 | Succ q3

q32 → Pnot q0

q42 → 5 q46, q47

q43 → 15 q16,q52

q44 → 10 q56, 57

q45 → Pnot q0

q46 → Nots q50 | Succ q3

q47 → Nots q50 | Succ q3

q52 → Nots q54 | Succ q3

q56 → Nots q59 | Succ q3

q57 → Nots q59 | Succ q3

Production Rules for DFA

Q0 → Op1 Q1 | Op2 Q2 | SelNp D | Sel MR D | SelSpF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAv D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q1 → Op1 D | Op2 D | SelNp q4 | SelMR q5 | Selg SPF q5 | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q2 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm q7 | SelP1 q8 | SelC q9 |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q4 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv q13 | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q5 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv q14 | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q6 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv q15 | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q7 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |

CheckAv q17 | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q8 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv q18 | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q9 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv q19 | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q13 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av q23 | NotAV q32 | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q14 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av 24 | NotAV 32 | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q15 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av 25 | NotAV 32 | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q16 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS q54 | Succ q3

Q17 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |

CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q18 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q19 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av q24 | NotAV q45 | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q23 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 q26,q27 | 15 D | 55 D | 5 D | NotS D | Succ D

Q24 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 q28,q29 | 55 D | 5 D | NotS D | Succ D

Q25 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D| 55 D | 5 D | NotS D | Succ D

Q26 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS q34 | Succ q3

Q27 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |

CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS q34 | Succ q3

Q28 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS q37 | Succ q3

Q29 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS q37 | Succ q3

Q30 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS q40 | Succ q3

Q31 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS q40 | Succ q3

Q32 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 D | NotS D | Succ D

Q42 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D | 55 D | 5 q46,q47 | NotS D | Succ D

Q43 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |

CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 q16,q25 | 55 D | 5 D | NotS D | Succ D

Q44 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 q56,q57 | 15 D| 55 D | 5 D | NotS D | Succ D

Q45 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D| 55 D | 5 D | NotS D | Succ D

Q46 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D| 55 D | 5 D | NotS 50 | Succ 3

Q47 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D| 55 D | 5 D | NotS 5 | Succ 3

Q52 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D| 55 D | 5 D | NotS 54 | Succ 3

Q56 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |
CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D| 55 D | 5 D | NotS 59 | Succ 3

Q57 → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |

CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D| 55 D | 5 D | NotS 59 | Succ 3

D → Op1 D | Op2 D | SelNp D | SelMR D | Selg SPF D | SelPm D | SelP1 D | SelC D |

CheckAv D | Av D | NotAV D | Pmot D | 10 D | 15 D| 55 D | 5 D | NotS D | Succ D

Table for op1 new status name

We have to rewrite statue names so the CFG website can accept them :

Q0	S
Q1	a
Q4	b
Q5	c
Q6	e
Q14	g
Q15	h
Q32	y
Q23	i
Q24	k
Q25	l
Q26	m
Q27	n
Q28	o
Q29	p
Q37	u
Q34	x
Q13	f
Q31	q
Q30	r
Q40	v
Q3	t

OP1 CFG:

This is the CFG you have input above:

Start symbol: **S**
S → op1**A** | reset**W** | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 15D | 5D | notsD | succD
A → selnp**B** | selmr**C** | selspf**E** | op1D | checkD | yD | nD | pnotD | 10D | 15D | 5D | notsD | succD
W → ε
B → check**F** | op1D | selnpD | selmrD | selspfD | yD | nD | pnotD | 10D | 15D | 5D | notsD | succD
C → check**G** | op1D | selnpD | selmrD | selspfD | yD | nD | pnotD | 10D | 15D | 5D | notsD | succD
E → check**H** | op1D | selnpD | selmrD | selspfD | yD | nD | pnotD | 10D | 15D | 5D | notsD | succD
F → y**I** | n**J** | op1D | selnpD | selmrD | selspfD | checkD | pnotD | 10D | 15D | 5D | notsD | succD
G → y**K** | n**J** | op1D | selnpD | selmrD | selspfD | checkD | pnotD | 10D | 15D | 5D | notsD | succD
H → y**L** | n**J** | op1D | selnpD | selmrD | selspfD | checkD | pnotD | 10D | 15D | 5D | notsD | succD
I → 10**M** | 10**N** | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 15D | 5D | notsD | succD
J → pnot**S** | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | 10D | 15D | 5D | notsD | succD
K → 15**O** | 15**P** | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 5D | notsD | succD
L → 5**R** | 5**Q** | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 15D | notsD | succD
M → succ**T** | notsX | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 15D | 5D
N → succ**T** | notsX | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 15D | 5D
O → succ**T** | notsU | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 15D | 5D
P → succ**T** | notsU | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 15D | 5D
R → succ**T** | notsV | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 15D | 5D
Q → succ**T** | notsV | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | pnotD | 10D | 15D | 5D
T → ε
X → ε
U → ε
V → ε

Op1 strings:

Accepted Strings in OP1 examples :

Op1selspfchecky5succ

Op1selnpcheckmpnotreset

Op1selmrchecky15succ

Op1selnpchecky10succ

Rejected Strings in OP1 examples :

Op1selnp

100

Op1selmr500

Op2

Screenshots of OP1:

CFG Developer

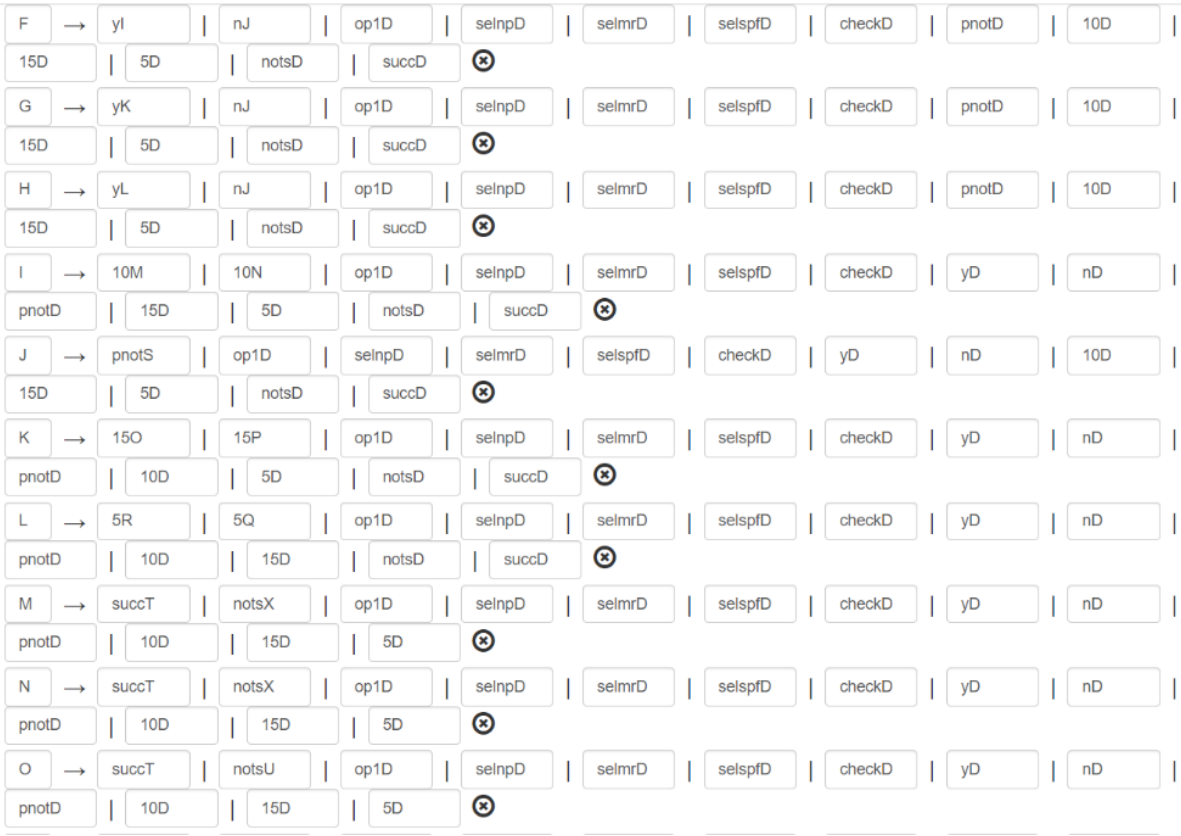
Create

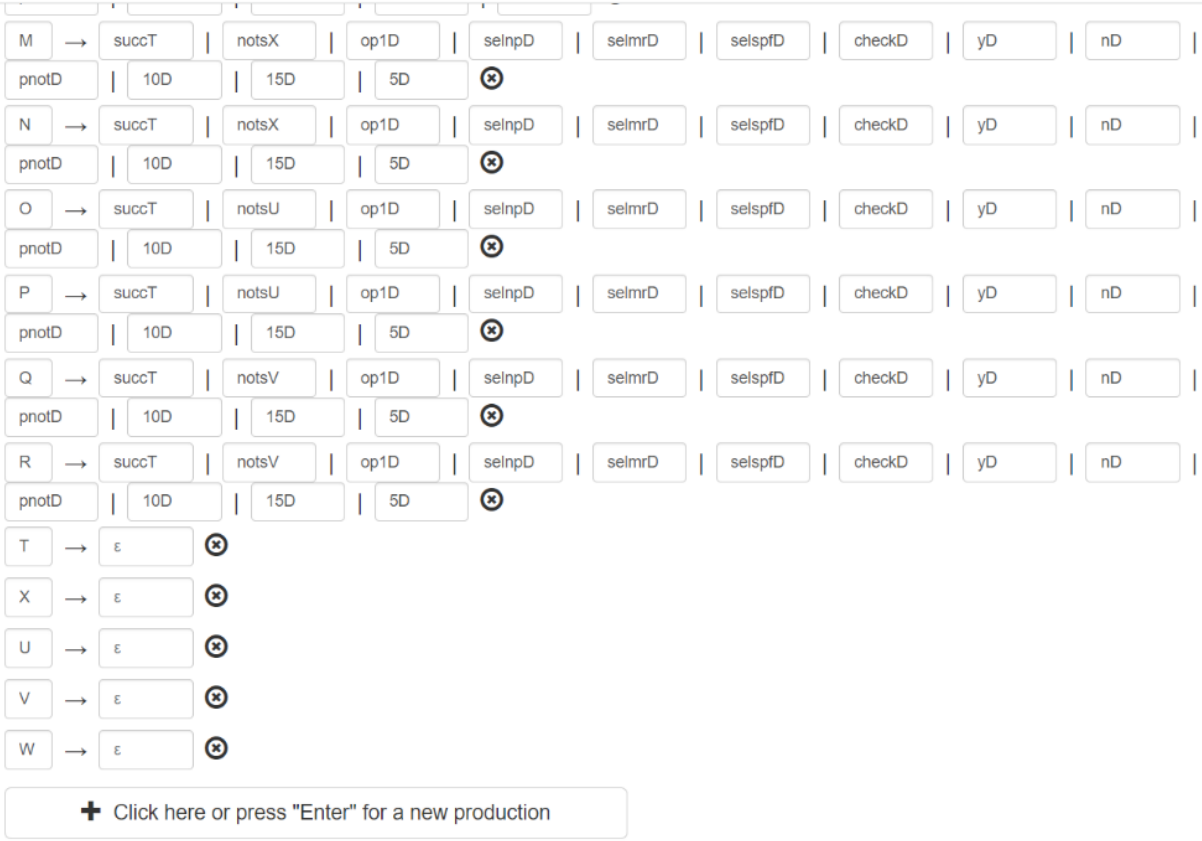
Input your context-free grammar (CFG) here. The start symbol has already been filled in for you.

- The left-hand nonterminal of each production must be filled in.
- [ε] - An empty text field corresponds to epsilon.
- [|] - For "or", use the standard pipe character that you use while coding.
- Input is case-sensitive. Whitespace is not ignored.

Reset Example

S →	op1A	resetW	selnpD	selmrD	selspfD	checkD	yD	nD	pnotD
10D	15D	5D	notsD	succD					
A →	selnpB	selmrC	selspfE	op1D	checkD	yD	nD	pnotD	10D
15D	5D	notsD	succD	⊗					
B →	checkF	op1D	selnpD	selmrD	selspfD	yD	nD	pnotD	10D
15D	5D	notsD	succD	⊗					
C →	checkG	op1D	selnpD	selmrD	selspfD	yD	nD	pnotD	10D
15D	5D	notsD	succD	⊗					
E →	checkH	op1D	selnpD	selmrD	selspfD	yD	nD	pnotD	10D
15D	5D	notsD	succD	⊗					
F →	yI	nJ	op1D	selnpD	selmrD	selspfD	checkD	pnotD	10D
15D	5D	notsD	succD	⊗					


F → yI | nJ | op1D | selnpD | selmrD | selspfD | checkD | pnotD | 10D |
15D | 5D | notsD | succD
G → yK | nJ | op1D | selnpD | selmrD | selspfD | checkD | pnotD | 10D |
15D | 5D | notsD | succD
H → yL | nJ | op1D | selnpD | selmrD | selspfD | checkD | pnotD | 10D |
15D | 5D | notsD | succD
I → 10M | 10N | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 15D | 5D | notsD | succD
J → pnotS | op1D | selnpD | selmrD | selspfD | checkD | yD | nD | 10D |
15D | 5D | notsD | succD
K → 15O | 15P | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 5D | notsD | succD
L → 5R | 5Q | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | notsD | succD
M → succT | notsX | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
N → succT | notsX | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
O → succT | notsU | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D


M → succT | notsX | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
N → succT | notsX | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
O → succT | notsU | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
P → succT | notsU | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
Q → succT | notsV | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
R → succT | notsV | op1D | selnpD | selmrD | selspfD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
T → ε
X → ε
U → ε
V → ε
W → ε
Click here or press "Enter" for a new production

Test

To test the CFG above, input test strings here, one per line. An empty line corresponds to the empty string. Results will be shown automatically.

```
op1selmrchecky15succ
op1selnp
op1selmrchecky15succ
op1selnpchecky10succ
op2
1000
```

Test Results for CFG

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
4	"op1selnpchecknpnotreset"	Yes	See Derivation
5	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
6	"op1selnp"	No	
7	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op1selnpchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
9	"op2"	No	
10	"1000"	No	
11	""	No	

Test

To test the CFG above, input test strings here, one per line. An empty line corresponds to the empty string. Results will be shown automatically.

```
op1selspfchecky5succ
op1selnpchecknpnotop1selspfchecky5succ
op1selmrchecky15succ
op1selnpchecknpnotreset
op1selmrchecky15succ
op1selmrchecky15succ
op1selnpchecky10succ
```

Test Results for CFG

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
4	"op1selnpchecknpnotreset"	Yes	See Derivation
5	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
6	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
7	"op1selnpchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2"	No	
9	"1000"	No	
10	""	No	

Test

To test the CFG above, input test strings here, one per line. An empty line corresponds to the empty string. Results will be shown automatically.

```
op1selspfchecky5succ
op1selnpchecknpnotop1selspfchecky5succ
op1selmrchecky15succ
op1selnpchecknpnotreset
op1selmrchecky15succ
op1selmrchecky15succ
op1selnpchecky10succ
```

Test Results for CFG

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op1A	S	op1A
A → selspfE	op1A	op1selspfE
E → checkH	op1selspfE	op1selspfcheckH
H → yL	op1selspfcheckH	op1selspfcheckyL
L → 5R	op1selspfcheckyL	op1selspfchecky5R
R → succT	op1selspfchecky5R	op1selspfchecky5succT
T → ε	op1selspfchecky5succT	op1selspfchecky5succ

2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
4	"op1selnpchecknpnotreset"	Yes	See Derivation
5	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
6	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
7	"op1selnpchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two

line corresponds to the empty string. Results will be shown automatically.

```
op1selspfchecky5succ
op1selnpchecknpnotop1selspfchecky5succ
op1selmrchecky15succ
op1selnpchecknpnotreset
op1selmrchecky15succ
op1selmrchecky15succ
op1selnpchecky10succ
```

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op1A	S	op1A
A → selnpB	op1A	op1selnpB
B → checkF	op1selnpB	op1selnpcheckF
F → nJ	op1selnpcheckF	op1selnpchecknJ
J → pnotS	op1selnpchecknJ	op1selnpchecknpnotS
S → op1A	op1selnpchecknpnotS	op1selnpchecknpnotop1A
A → selspfE	op1selnpchecknpnotop1A	op1selnpchecknpnotop1selspfE
E → checkH	op1selnpchecknpnotop1selspfE	op1selnpchecknpnotop1selspfcheckH
H → yL	op1selnpchecknpnotop1selspfcheckH	op1selnpchecknpnotop1selspfcheckyL
L → 5R	op1selnpchecknpnotop1selspfcheckyL	op1selnpchecknpnotop1selspfchecky5R
R → succT	op1selnpchecknpnotop1selspfchecky5R	op1selnpchecknpnotop1selspfchecky5succT
T → ε	op1selnpchecknpnotop1selspfchecky5succT	op1selnpchecknpnotop1selspfchecky5succ

3	"op1selmrchecky15succ"	Yes	Derivation One

line corresponds to the empty string. Results will be shown automatically.

```
op1selspfchecky5succ
op1selnpchecknpnotop1selspfchecky5succ
op1selmrchecky15succ
op1selnpchecknpnotreset
op1selmrchecky15succ
op1selmrchecky15succ
op1selnpchecky10succ
```

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op1A	S	op1A
A → selmrC	op1A	op1selmrC
C → checkG	op1selmrC	op1selmrcheckG
G → yK	op1selmrcheckG	op1selmrcheckyK
K → 15O	op1selmrcheckyK	op1selmrchecky15O
O → succT	op1selmrchecky15O	op1selmrchecky15succT
T → ε	op1selmrchecky15succT	op1selmrchecky15succ

4	"op1selnpchecknpnotreset"	Yes	See Derivation
5	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
6	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
7	"op1selnpchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2"	No	
9	"1000"	No	

line corresponds to the empty string. Results will be shown automatically.

```
op1selspfchecky5succ
op1selnpchecknpnotop1selspfchecky5succ
op1selmrchecky15succ
op1selnpchecknpnotreset
op1selmrchecky15succ
op1selmrchecky15succ
op1selnpchecky10succ
```

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
4	"op1selnpchecknpnotreset"	Yes	See Derivation

Rule	Application	Result
Start → S	Start	S
S → op1A	S	op1A
A → selnpB	op1A	op1selnpB
B → checkF	op1selnpB	op1selnpcheckF
F → nJ	op1selnpcheckF	op1selnpchecknJ
J → pnotS	op1selnpchecknJ	op1selnpchecknpnotS
S → resetW	op1selnpchecknpnotS	op1selnpchecknpnotresetW
W → ε	op1selnpchecknpnotresetW	op1selnpchecknpnotreset

5	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
6	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
7	"op1selnpchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2"	No	
9	"1000"	No	

line corresponds to the empty string. Results will be shown automatically.

```
op1selspfchecky5succ
op1selnpchecknpnotop1selspfchecky5succ
op1selmrchecky15succ
op1selnpchecknpnotreset
op1selmrchecky15succ
op1selmrchecky15succ
op1selnpchecky10succ
```

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
4	"op1selnpchecknpnotreset"	Yes	See Derivation
5	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op1A	S	op1A
A → selmrC	op1A	op1selmrC
C → checkG	op1selmrC	op1selmrcheckG
G → yK	op1selmrcheckG	op1selmrcheckyK
K → 15O	op1selmrcheckyK	op1selmrchecky15O
O → succT	op1selmrchecky15O	op1selmrchecky15succT
T → ε	op1selmrchecky15succT	op1selmrchecky15succ

6	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
7	"op1selnpchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2"	No	
9	"1000"	No	

line corresponds to the empty string. Results will be shown automatically.

```
op1selspfchecky5succ
op1selnpchecknpnotop1selspfchecky5succ
op1selmrchecky15succ
op1selnpchecknpnotreset
op1selmrchecky15succ
op1selmrchecky15succ
op1selnpchecky10succ
```

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
4	"op1selnpchecknpnotreset"	Yes	See Derivation
5	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
6	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op1A	S	op1A
A → selmrC	op1A	op1selmrC
C → checkG	op1selmrC	op1selmrcheckG
G → yK	op1selmrcheckG	op1selmrcheckyK
K → 15O	op1selmrcheckyK	op1selmrchecky15O
O → succT	op1selmrchecky15O	op1selmrchecky15succT
T → ε	op1selmrchecky15succT	op1selmrchecky15succ

7	"op1selnpchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2"	No	
9	"1000"	No	

line corresponds to the empty string. Results will be shown automatically.

```
op1selspfchecky5succ
op1selnpchecknpnotop1selspfchecky5succ
op1selmrchecky15succ
op1selnpchecknpnotreset
op1selmrchecky15succ
op1selmrchecky15succ
op1selnpchecky10succ
```

#	String	Matches	
1	"op1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op1selnpchecknpnotop1selspfchecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
4	"op1selnpchecknpnotreset"	Yes	See Derivation
5	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
6	"op1selmrchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
7	"op1selnpchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op1A	S	op1A
A → selnpB	op1A	op1selnpB
B → checkF	op1selnpB	op1selnpcheckF
F → yI	op1selnpcheckF	op1selnpcheckyI
I → 10M	op1selnpcheckyI	op1selnpchecky10M
M → succT	op1selnpchecky10M	op1selnpchecky10succT
T → ε	op1selnpchecky10succT	op1selnpchecky10succ

8	"op2"	No	
9	"1000"	No	

Table for op2 new status name

Q0	S
Q2	a
Q7	b
Q8	c
Q9	e
Q17	g
Q18	h
Q19	i
Q42	j
Q43	k
Q44	l
Q45	m
Q46	n
Q47	o
Q50	x
Q16	d
Q52	f
Q54	w
Q56	q
Q57	r
Q59	z
Q3	t

OP2 CFG:

This is the CFG you have input above:

Start symbol: **S**
S → op2**A** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D** | reset**V**
A → selpa**B** | selpl**C** | selc**E** | op2**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
D → ε
V → ε
B → check**G** | op2**D** | selpa**D** | selpl**D** | selc**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
C → check**H** | op2**D** | selpa**D** | selpl**D** | selc**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
E → check**I** | op2**D** | selpa**D** | selpl**D** | selc**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
G → y**J** | n**M** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
H → y**K** | n**M** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
I → y**L** | n**M** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
F → succ**T** | nots**W** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D**
T → ε
W → ε
J → 5**N** | 5**O** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | nots**D** | succ**D**
M → pnot**S** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
K → 15**P** | 15**F** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 5**D** | nots**D** | succ**D**
L → 10**Q** | 10**R** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 15**D** | 5**D** | nots**D** | succ**D**
N → succ**T** | nots**X** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
O → succ**T** | nots**X** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
P → succ**T** | nots**W** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
Q → succ**T** | nots**Z** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
R → succ**T** | nots**Z** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
X → ε
Z → ε

Op2 strings:

Accepted Strings in OP2 examples :

Op2selpachecky5succ

Op2selplchecky15succ

Op2selcchecky10succ

Op2selpachecknpnotreset

Rejected Strings in OP2 examples :

Op2selcchecky100

Op1

Op2selspf

Op2selpa200

Screenshots of OP2:

Create

Input your context-free grammar (CFG) here. The start symbol has already been filled in for you.

- The left-hand nonterminal of each production must be filled in.
- [ε] - An empty text field corresponds to epsilon.
- [|] - For "or", use the standard pipe character that you use while coding.
- Input is case-sensitive. Whitespace is not ignored.

Reset Example

S → op2A | selpaD | selplD | selcD | checkD | yD | nD | pnotD | 10D |
15D | 5D | notsD | succD | resetV

A → selpaB | selplC | selcE | op2D | checkD | yD | nD | pnotD | 10D |
15D | 5D | notsD | succD ⊗

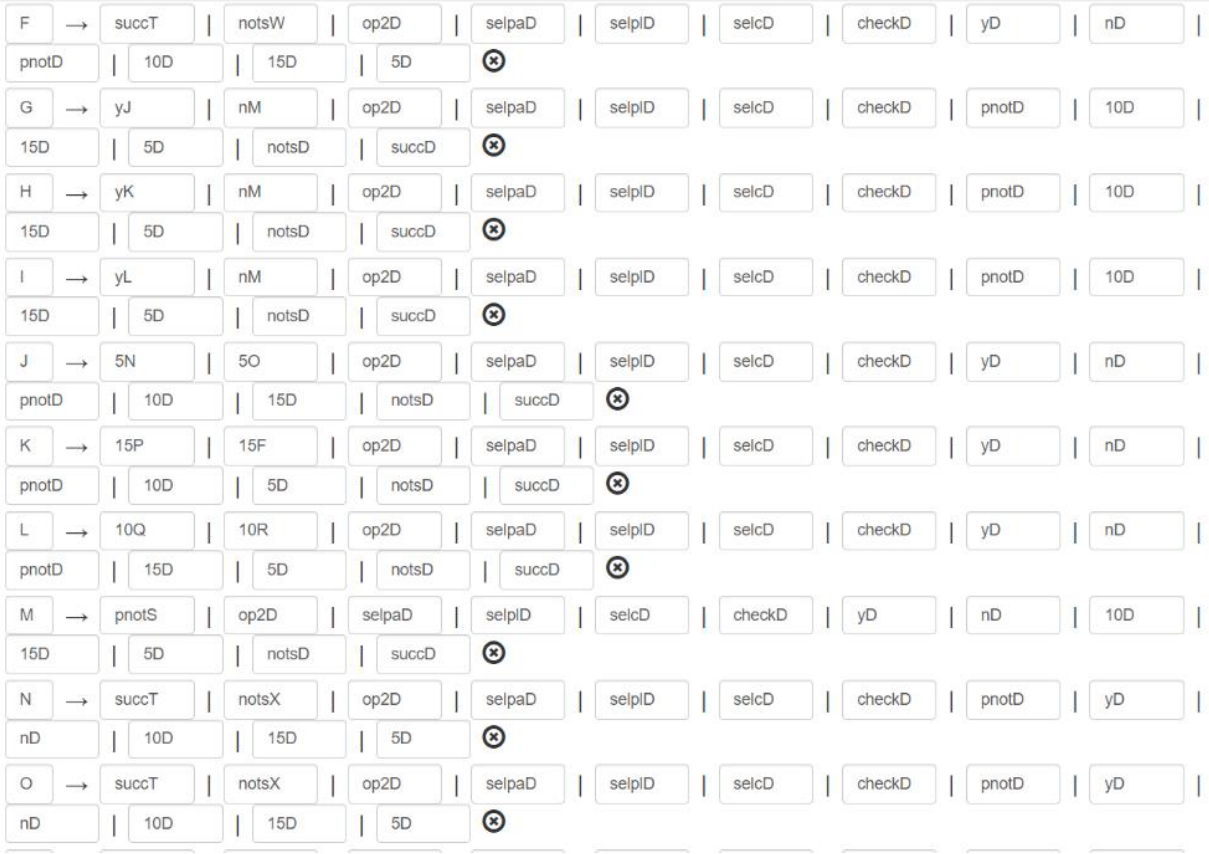
B → checkG | op2D | selpaD | selplD | selcD | yD | nD | pnotD | 10D |
15D | 5D | notsD | succD ⊗

C → checkH | op2D | selpaD | selplD | selcD | yD | nD | pnotD | 10D |
15D | 5D | notsD | succD ⊗

D → ε ⊗

E → checkI | op2D | selpaD | selplD | selcD | yD | nD | pnotD | 10D |
15D | 5D | notsD | succD ⊗

F → succT | notsW | op2D | selpaD | selplD | selcD | checkD | yD | nD |
pnotD | 10D | 15D | 5D ⊗


F → succT | notsW | op2D | selpaD | selplD | selcD | checkD | yD | nD |
pnotD | 10D | 15D | 5D
G → yJ | nM | op2D | selpaD | selplD | selcD | checkD | pnotD | 10D |
15D | 5D | notsD | succD
H → yK | nM | op2D | selpaD | selplD | selcD | checkD | pnotD | 10D |
15D | 5D | notsD | succD
I → yL | nM | op2D | selpaD | selplD | selcD | checkD | pnotD | 10D |
15D | 5D | notsD | succD
J → 5N | 5O | op2D | selpaD | selplD | selcD | checkD | yD | nD |
pnotD | 10D | 15D | notsD | succD
K → 15P | 15F | op2D | selpaD | selplD | selcD | checkD | yD | nD |
pnotD | 10D | 5D | notsD | succD
L → 10Q | 10R | op2D | selpaD | selplD | selcD | checkD | yD | nD |
pnotD | 15D | 5D | notsD | succD
M → pnotS | op2D | selpaD | selplD | selcD | checkD | yD | nD | 10D |
15D | 5D | notsD | succD
N → succT | notsX | op2D | selpaD | selplD | selcD | checkD | pnotD | yD |
nD | 10D | 15D | 5D
O → succT | notsX | op2D | selpaD | selplD | selcD | checkD | pnotD | yD |
nD | 10D | 15D | 5D


pnotD | 15D | 5D | notsD | succD
M → pnotS | op2D | selpaD | selplD | selcD | checkD | yD | nD | 10D |
15D | 5D | notsD | succD
N → succT | notsX | op2D | selpaD | selplD | selcD | checkD | pnotD | yD |
nD | 10D | 15D | 5D
O → succT | notsX | op2D | selpaD | selplD | selcD | checkD | pnotD | yD |
nD | 10D | 15D | 5D
P → succT | notsW | op2D | selpaD | selplD | selcD | checkD | pnotD | yD |
nD | 10D | 15D | 5D
Q → succT | notsZ | op2D | selpaD | selplD | selcD | checkD | pnotD | yD |
nD | 10D | 15D | 5D
R → succT | notsZ | op2D | selpaD | selplD | selcD | checkD | pnotD | yD |
nD | 10D | 15D | 5D
T → ε
W → ε
X → ε
Z → ε
V → ε
Click here or press "Enter" for a new production

Verify

This is the CFG you have input above:

Start symbol: **S**
S → op2**A** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D** | reset**V**
A → selpa**B** | selpl**C** | selc**E** | op2**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
D → ε
V → ε
B → check**G** | op2**D** | selpa**D** | selpl**D** | selc**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
C → check**H** | op2**D** | selpa**D** | selpl**D** | selc**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
E → check**I** | op2**D** | selpa**D** | selpl**D** | selc**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
G → y**J** | n**M** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
H → y**K** | n**M** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
I → y**L** | n**M** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
F → succ**T** | nots**W** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | 5**D**
T → ε
W → ε
J → 5**N** | 5**O** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 15**D** | nots**D** | succ**D**
M → pnot**S** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D** | nots**D** | succ**D**
K → 15**P** | 15**F** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 10**D** | 5**D** | nots**D** | succ**D**
L → 10**Q** | 10**R** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | y**D** | n**D** | pnot**D** | 15**D** | 5**D** | nots**D** | succ**D**
N → succ**T** | nots**X** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
O → succ**T** | nots**X** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
P → succ**T** | nots**W** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
Q → succ**T** | nots**Z** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
R → succ**T** | nots**Z** | op2**D** | selpa**D** | selpl**D** | selc**D** | check**D** | pnot**D** | y**D** | n**D** | 10**D** | 15**D** | 5**D**
X → ε
Z → ε

Some strings from the language of this grammar:

```
op2selpachecky55
op2selpaselpa
op2succ
selc
10
op2pnot
```

Test

To test the CFG above, input test strings here, one per line. An empty line corresponds to the empty string. Results will be shown automatically.

```
op2selpachecky10
op2selplchecknpnotreset
op2selplchecknpnotreset
op1
op2selspf
100
55
```

Test Results for CFG

#	String	Matches	
1	"op2selpachecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op2selplcheck5"	Yes	See Derivation
4	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
5	"op2selpachecknpnotreset"	Yes	See Derivation
6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2selccheckpnotop2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
9	"op2selpacheck10"	Yes	See Derivation
10	"op2selplchecknpnotreset"	Yes	See Derivation
11	"op2selplchecknpnotreset"	Yes	See Derivation
12	"op1"	No	
13	"op2selspf"	No	
14	"100"	No	

Test

To test the CFG above, input test strings here, one per line. An empty line corresponds to the empty string. Results will be shown automatically.

op2selpachecky10
op2selplcheckpnotreset
op2selplcheckpnotreset
op1
op2selspf
100
55

Test Results for CFG

#	String	Matches	
1	"op2selpachecky5succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op2A	S	op2A
A → selpaB	op2A	op2selpaB
B → checkG	op2selpaB	op2selpacheckG
G → yJ	op2selpacheckG	op2selpacheckyJ
J → 5N	op2selpacheckyJ	op2selpachecky5N
N → succT	op2selpachecky5N	op2selpachecky5succT
T → ε	op2selpachecky5succT	op2selpachecky5succ

2	"op2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op2selplcheck5"	Yes	See Derivation
4	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
5	"op2selpachecknpnotreset"	Yes	See Derivation
6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2selccheckpnotop2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two

Test

To test the CFG above, input test strings here, one per line. An empty line corresponds to the empty string. Results will be shown automatically.

op2selpacheck10
op2selplchecknpnotreset
op2selplchecknpnotreset
op1
op2selspf
100
55

Test Results for CFG

#	String	Matches	
1	"op2selpachecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op2A	S	op2A
A → selplC	op2A	op2selplC
C → checkH	op2selplC	op2selplcheckH
H → yK	op2selplcheckH	op2selplcheckyK
K → 15P	op2selplcheckyK	op2selplchecky15P
P → succT	op2selplchecky15P	op2selplchecky15succT
T → ε	op2selplchecky15succT	op2selplchecky15succ

3	"op2selplcheck5"	Yes	See Derivation
4	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
5	"op2selpachecknpnotreset"	Yes	See Derivation
6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2selccheckpnotop2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two

Test

To test the CFG above, input test strings here, one per line. An empty line corresponds to the empty string. Results will be shown automatically.

op2selpachecky10
op2selplcheckpnotreset
op2selplcheckpnotreset
op1
op2selspf
100
55

Test Results for CFG

#	String	Matches	
1	"op2selpachecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op2selplcheck5"	Yes	See Derivation
4	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op2A	S	op2A
A → selcE	op2A	op2selcE
E → checkI	op2selcE	op2selccheckI
I → yL	op2selccheckI	op2selccheckyL
L → 10Q	op2selccheckyL	op2selccheckyl0Q
Q → succT	op2selccheckyl0Q	op2selccheckyl0succT
T → ε	op2selccheckyl0succT	op2selccheckyl0succ

5	"op2selpachecknpnotreset"	Yes	See Derivation
6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2selccheckpnotop2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two

Test

To test the CFG above, input test strings here, one per line. An empty line corresponds to the empty string. Results will be shown automatically.

op2selpacheck10
op2selplchecknpnotreset
op2selplchecknpnotreset
op1
op2selspf
100
55

Test Results for CFG

#	String	Matches	
1	"op2selpachecky5succ"	Yes (ambiguously)	Derivation One Derivation Two
2	"op2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
3	"op2selplcheck5"	Yes	See Derivation
4	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
5	"op2selpachecknpnotreset"	Yes	See Derivation
6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op2A	S	op2A
A → selcE	op2A	op2selcE
E → checkI	op2selcE	op2selccheckI
I → yL	op2selccheckI	op2selccheckyL
L → 10Q	op2selccheckyL	op2selcchecky10Q
Q → succT	op2selcchecky10Q	op2selcchecky10succT
T → ε	op2selcchecky10succT	op2selcchecky10succ

8	"op2selccheckpnotop2selplchecky15succ"	Yes	Derivation One

6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2selccheckpnotop2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two

Rule	Application	Result
Start → S	Start	S
S → op2A	S	op2A
A → selcE	op2A	op2selcE
E → checkI	op2selcE	op2selccheckI
I → nM	op2selccheckI	op2selccheckpM
M → pnotS	op2selccheckpM	op2selccheckpnotS
S → op2A	op2selccheckpnotS	op2selccheckpnotop2A
A → selplC	op2selccheckpnotop2A	op2selccheckpnotop2selplC
C → checkH	op2selccheckpnotop2selplC	op2selccheckpnotop2selplcheckH
H → yK	op2selccheckpnotop2selplcheckH	op2selccheckpnotop2selplcheckyK
K → 15P	op2selccheckpnotop2selplcheckyK	op2selccheckpnotop2selplchecky15P
P → succT	op2selccheckpnotop2selplchecky15P	op2selccheckpnotop2selplchecky15succT
T → ε	op2selccheckpnotop2selplchecky15succT	op2selccheckpnotop2selplchecky15succ

6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2selccheckpnotop2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
9	"op2selpacheck10"	Yes	See Derivation

Rule	Application	Result
Start → S	Start	S
S → op2A	S	op2A
A → selpaB	op2A	op2selpaB
B → checkG	op2selpaB	op2selpacheckG
G → 10D	op2selpacheckG	op2selpacheck10D
D → ε	op2selpacheck10D	op2selpacheck10

10	"op2selplchecknpnotreset"	Yes	See Derivation
11	"op2selplchecknpnotreset"	Yes	See Derivation
12	"op1"	No	
13	"op2selspf"	No	
14	"100"	No	
15	"55"	No	
16	""	No	

6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2selccheckpnotop2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
9	"op2selpacheck10"	Yes	See Derivation
10	"op2selplcheckpnotreset"	Yes	See Derivation

Rule	Application	Result
Start → S	Start	S
S → op2A	S	op2A
A → selplC	op2A	op2selplC
C → checkH	op2selplC	op2selplcheckH
H → nM	op2selplcheckH	op2selplchecknM
M → pnotS	op2selplchecknM	op2selplchecknpnotS
S → resetV	op2selplchecknpnotS	op2selplchecknpnotresetV
V → ε	op2selplchecknpnotresetV	op2selplchecknpnotreset

11	"op2selplcheckpnotreset"	Yes	See Derivation
12	"op1"	No	
13	"op2selspf"	No	
14	"100"	No	
15	"55"	No	
16	""	No	

6	"op2selccheckpnotreset"	Yes	See Derivation
7	"op2selcchecky10succ"	Yes (ambiguously)	Derivation One Derivation Two
8	"op2selccheckpnotop2selplchecky15succ"	Yes (ambiguously)	Derivation One Derivation Two
9	"op2selpacheck10"	Yes	See Derivation
10	"op2selplcheckpnotreset"	Yes	See Derivation
11	"op2selplcheckpnotreset"	Yes	See Derivation

Rule	Application	Result
Start → S	Start	S
S → op2A	S	op2A
A → selplC	op2A	op2selplC
C → checkH	op2selplC	op2selplcheckH
H → nM	op2selplcheckH	op2selplchecknM
M → pnotS	op2selplchecknM	op2selplcheckpnotS
S → resetV	op2selplcheckpnotS	op2selplcheckpnotresetV
V → ε	op2selplcheckpnotresetV	op2selplcheckpnotreset

12	"op1"	No
13	"op2selspf"	No
14	"100"	No
15	"55"	No
16	""	No

Conclusion

The implementation of pharmacy vending machines can be achieved through the application of finite automata. Finite State Automata, which function similarly to digital computers, are integral components of informatics. These automata receive inputs, generate outputs, store temporary information, and make decisions during the process of transforming input into output. A pharmacy vending machine, designed using finite automata, would consist of a finite number of states, each containing information about previous inputs.
By incorporating the principles of finite automata into the design of pharmacy vending machines, the aim is to create a seamless and user-friendly experience for individuals seeking immediate access to essential pharmaceutical products. These machines have the potential to revolutionize the accessibility and convenience of pharmacies in diverse settings, ensuring that people's emergent needs are efficiently met wherever they may be.[2]

In conclusion, the utilization of pharmacy vending machines represents a promising solution to the growing demand for pharmacies in various locations. By leveraging the functionality of finite automata, these machines can effectively cater to people's urgent requirements while offering a convenient and accessible alternative to traditional pharmacy services. As technology continues to advance, the implementation of innovative solutions like pharmacy vending machines holds great potential for improving healthcare accessibility and meeting the emergent needs of individuals worldwide.[1]

References

[1]:A. Alrehily, R. Fallatah, and V. Thayananthan, “Design of vending machine using finite state machine and visual automata simulator,” *International Journal of Computer Applications*, vol. 115, no. 18, pp. 37–42, 2015.

[2] Introduction to Automata Theory, Languages, and Computation, Pearson New International Edition, 2014 (3/E), John E. Hopcroft, Rajeev Motwani, Jeffrey D. Ullman, ISBN-10: 1292039051 ISBN-13: 9781292039053.

[3] A. Monga, “Finite State Machine based vending machine controller with auto-billing features,” International Journal of VLSI Design & Communication Systems, vol. 3, no. 2, pp. 19–28, 2012.