МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МІСЬКОГО ГОСПОДАРСТВА імені О. М. БЕКЕТОВА

О. О. Петрова

ДОСЛІДЖЕННЯ РОБОТИ МАШИНИ ТЮРІНГА

НАВЧАЛЬНО-МЕТОДИЧНИЙ ПОСІБНИК

2-ге видання, перероблене і доповнене

Харків
ХНУМГ ім. О. М. Бекетова
2021

УДК 510.582 (076)
П30

Автор

Петрова Олена Олександрівна, кандидат технічних наук, доцент кафедри комп'ютерних наук та інформаційних технологій Харківського національного університету міського господарства імені О. М. Бекетова

Рецензенти:

Чуб Ігор Андрійович, доктор технічних наук, професор, начальник кафедри ПІНП Національного університету цивільного захисту України;

Костенко Олександр Борисович, кандидат фізико-математичних наук, доцент кафедри комп'ютерних наук та інформаційних технологій Харківського національного університету міського господарства імені О. М. Бекетова.

Рекомендовано до друку Вченою радою ХНУМГ ім. О. М. Бекетова, протокол № 12 від 01.07.2020.

Петрова О. О.

П30 Дослідження роботи машини Тюрінга : навч.-метод. посібник / О. О. Петрова ; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – 2-ге вид., перероб. і доп. – Харків : ХНУМГ ім. О. М. Бекетова, 2021. – 75 с.

Матеріал 1-го видання навчально-методичного посібника «Дослідження роботи машини Тьюринга» з дисципліни «Теорія алгоритмів» (затверджено науково-методичною радою Харківського національного університету будівництва та архітектури, протокол № 8 від 28.05.2015) доповнено та перероблено вступ, допрацьовані задачі з прикладами їхнього розв'язання. Розглянуто питання формального уточнення поняття алгоритму за допомогою машини Тюрінга; викладено принципи роботи машини Тюрінга як абстрактної математичної моделі алгоритмів та універсального визначення поняття алгоритму; розглянуто абстрактний виконавець, призначений для формалізації поняття алгоритму.

Навчально-методичний посібник вміщує теоретичний матеріал, який стосується автоматного програмування. Для закріплення знань пропонуються детальні пояснення функціональних схем машин Тюрінга та варіанти індивідуальних завдань до практичних занять та самостійного вивчення тем.

Призначено для студентів спеціальностей 122 – Комп'ютерні науки, 126 – Інформаційні системи та технології, а також усіх тих, хто цікавиться цими питаннями.

УДК 510.582 (076)

ЗМІСТ

ВСТУП………………………………………………………………… 4
1 ОСНОВНІ ПОНЯТТЯ ТЕОРІЇ АЛГОРИТМІВ……………………….. 6
1.1 Визначення алгоритму…………………………………………….. 6
1.2 Властивості машини Тюрінга як алгоритму…………...………. 7
1.3 Формальне визначення алгоритму………………………………… 8
2 МОДЕЛЮВАННЯ РОБОТИ МАШИНИ ТЮРІНГА ………………. 9
2.1 Структура машини Тюрінга……………………………………….. 9
2.2 Конфігурація та протокол машини Тюрінга ………………….. 12
2.3 Представлення машини Тюрінга …………............................... 13
2.3.1 Представлення машини Тюрінга сукупністю команд……… 13
2.3.2 Представлення машини Тюрінга у вигляді таблиці відповідності……………………………………………………… 14
2.3.3 Представлення машини Тюрінга у вигляді графа………….. 16
2.4 Відмінність машини Тюрінга від сучасного комп’ютера………... 16
3 ПРОГРАМНИЙ ІНТЕРПРЕТАТОР ALGO 2000………………… 18
3.1 Інтерфейс машини Тюрінга, реалізованої в програмному інтерпретаторі ALGO 2000…………………………....................... 18
3.2 Основні прийоми роботи в програмному інтерпретаторі ALGO 2000.. 19
4 ПРАКТИЧНІ ЗАНЯТТЯ………………………………………………… 21
4.1 Практичне заняття 1 Розв’язання простих задач за допомогою машини Тюрінга з використанням програмного інтерпретатора ALGO 2000……………………………………………………...…. 21
4.2 Практичне заняття 2 Моделювання роботи машини Тюрінга в програмному інтерпретаторі ALGO 2000……………………………... 35
4.3 Практичне заняття 3 Обробка масивів та символьних даних у програмному інтерпретаторі ALGO 2000………………………… 59
СПИСОК ДЖЕРЕЛ………………………………………………....... 74

ВСТУП

Поняття алгоритму в його загальному вигляді належить до основних, невизначених понять математики. Сьогодні найбільш поширеними є такі строго математичні уточнення понятті алгоритму:

− загальнорекурсивні функції, які пов'язують поняття алгоритму з найбільш традиційними поняттями математики: обчисленнями й числовими функціями;

− машини Тюрінга, які засновані на уявленні про алгоритм як про деякий детермінований пристрій, здатний виконувати в кожен окремий момент часу лише елементарні операції. Таке уявлення не залишає сумнівів в однозначності алгоритму й елементарності його кроків. Крім того, евристика цих моделей є близькою до електронно-обчислювальної машини (ЕОМ);

− нормальні алгоритми, які засновані на перетворенні слів у довільних алфавітах, в яких елементарними операціями є підстановки, тобто заміни частини слова на інше слово. Переваги цього типу полягають в його максимальній абстрактності та можливості застосовувати поняття алгоритму до об'єктів довільної природи [1].

Доведено, що всі ці уточнення еквівалентні між собою, тому на основі тези Тюрінга «для будь-якого алгоритму в деякому алфавіті існує алгоритм, який дає при однакових початкових даних той же результат, що і вихідний алгоритм», дозволяють виконувати обчислення згідно з обраним планом подібно до машини Тюрінга (МТ).

Формалізовані підходи до алгоритмів мають високу освітню цінність. Абстрактні машини Тюрінга є прекрасними засобами освоєння алгоритмізації при безмашинному та машинному варіанті викладання дисципліни «Теорія алгоритмів».

У 1936 році Алан Тюрінг (Turing) установив зв'язок загально рекурсивних функцій із загальними ідеями «обчислювальності» та «вивідності». Використовуючи схему абстрактної машини, Тьюринг довів низку важливих для кібернетики положень у галузі математичної логіки.

Завершений Тюрінгом в 1946 році проєкт «автоматичного обчислювального пристрою» (Automatic Computing Engine – ACE) розглядається зараз як розробка, що вміщувала перший докладний опис комп'ютера в сучасному розумінні цього слова.

Визначення МТ було запропоновано математиками як точне строго математичне уточнення поняття алгоритму: алгоритм – це машина Тюрінга.

МТ передбачає строгу математичну побудову, математичний апарат (аналогічний, наприклад, до апарата диференціальних рівнянь), створений для розв'язання певних задач, який був названий «машиною» через те, що за описом його складових і за своїм функціонуванням він схожий на обчислювальну машину [1].

Принципова відмінність МТ від обчислювальних машин полягає в тому, що її запам'ятовувальний пристрій становить нескінченну стрічку: реальні обчислювальні машини можуть мати доволі великий запам'ятовувальний пристрій, який обов'язково має бути кінцевим. Машину Тюрінга не можна реалізувати саме через нескінченність її стрічки. У цьому сенсі вона є потужнішою за будь-яку іншу обчислювальну машину.

МТ потрібно пізнавати за такими визначальними характеристиками:

1) вона є першоджерелом математики, у якому введено поняття алгоритму. Згідно з тезою Тюрінга, будь-який алгоритм може бути реалізований відповідною машиною. Ця теза є формальним визначенням алгоритму, який дозволяє будувати докази щодо існування або не існування алгоритмів, виходячи з опису відповідних машин;

2) вона є першоджерелом програмування, у якому вводиться поняття алгоритму;

3) Тюрінг довів існування задач, які алгоритмічно нерозв'язні [2].

Теоретичний матеріал дисципліни «Теорія алгоритмів» достатньо абстрактний та складний для розуміння студентами, тому потрібно вивчати його за побудовою великої кількості машин Тюрінга для конкретних функцій та задач, що дозволить студентам розуміти сутності машин Тюрінга.

Однією з цілей дисципліни є побудова строго математичного уточнення поняття алгоритму, а метою цього навчально-методичного посібника є вивчення принципів роботи МТ.

У навчально-методичному посібнику розглянуто програмне моделювання роботи МТ. Теоретичний матеріал кожного розділу супроводжується методикою розв'язання задач, прикладами та завданнями для виконання практичних занять.

У першому розділі розглянуто основні поняття теорії алгоритмів та вказано на необхідність їхнього математичного визначення.

У другому розділі наведено структуру МТ, розглянуто способи її представлення та операції, які вона виконує. Автором наведено теоретичні положення, структуру машини Тюрінга, дано визначення конфігурації (миттєвого зображення машини), описані докладно способи опису машини Тюрінга.

У третьому розділі розглянуто інтерфейс та прийоми роботи в програмному інтерпретаторі ALGO 2000.

1 ОСНОВНІ ПОНЯТТЯ ТЕОРІЇ АЛГОРИТМІВ

1.1 Визначення алгоритму

Інтуїтивне визначення алгоритму не дозволяє розглядати властивості алгоритмів як властивості формальних об’єктів. Тому математичне визначення алгоритму необхідно з таких причин:

1) тільки за наявності формального визначення алгоритму можна зробити висновок про розв’язання або нерозв’язання проблеми;

2) воно дає можливість порівнювати алгоритми, призначені для розв’язання однакових задач;

3) воно дає можливість порівнювати різні проблеми за складністю алгоритмів їхнього вирішення.

Одна з причин розпливчастості інтуїтивного визначення алгоритму полягає в тому, що об’єктом алгоритму може виявитися будь-що. Тому природно було б почати з формалізації поняття об’єкта. Об’єктами роботи в обчислювальних алгоритмах є числа, алгоритмів шахової гри – фігури й позиції, об’єктами ж алгоритмізації виробничих процесів слугують, наприклад, показання приладів. Однак алгоритми оперують не об’єктами реального світу, а їхніми зображеннями. Тому алгоритмами в сучасній математиці прийнято називати відповідності, що конструктивно задаються, між зображеннями об’єктів в абстрактних алфавітах.

Абстрактним алфавітом називається будь-яка кінцева сукупність об’єктів, званих літерами або символами конкретного алфавіту, до того ж природа цих об’єктів нас зовсім не цікавить. Символом абстрактних алфавітів можна вважати літери алфавіту мови, цифри, будь-які знаки і навіть слова деякої конкретної мови. Основною вимогою до алфавіту є його скінченність. Таким чином, можна стверджувати, що алфавіт – це кінцева множина різних символів. Алфавіт, як будь-яка множина, задається переліком його елементів.

Отже, об’єкти реального світу можна зображувати словами в різних алфавітах, що дозволяє вважати, що об’єктами роботи алгоритмів можуть бути тільки слова. Тоді можна сформулювати таке визначення: алгоритм є чіткою кінцевою системою правил для перетворення слів деякого алфавіту на слова цього самого алфавіту.

Слово, до якого застосовується алгоритм, називається вхідним. Слово, що утворюється в результаті застосування алгоритму, називається вихідним.

Сукупність слів, до яких застосовується цей алгоритм, називається областю застосованості цього алгоритму.

Формальні визначення алгоритму з’явилися в 30–40-х роках XX століття. Можна виділити три основні типи універсальних алгоритмічних моделей, що розрізняються вихідними евристичними міркуваннями стосовно того, що таке алгоритм.

Перший тип пов’язує поняття алгоритму з обчисленнями і числовими функціями. Найбільш розвинена і вивчена модель цього типу – рекурсивні функції – є історично першою формалізацією поняття алгоритму. Ця модель заснована на функціональному підході і розглядає поняття алгоритму з погляду того, що можна обчислити за його допомогою.

Другий тип заснований на представленні алгоритму як деякого детермінованого пристрою, здатного виконувати в кожен окремий момент деякі примітивні операції або інструкції. Таке уявлення не залишає сумнівів в однозначності алгоритму й елементарності його кроків. Основною теоретичною моделлю цього типу, яка створена в 30-х роках XX сторіччя, є машина Тюрінга, що становить автоматну модель, в основі якої лежить аналіз процесу виконання алгоритму як сукупності набору інструкцій.

Третій тип алгоритмічних моделей – це перетворення слів у довільних алфавітах, у яких елементарними операціями є підстановки, тобто заміни частини слова (підслова) на інше слово. Перевага цього типу полягає в його максимальній абстрактності та можливості застосувати поняття алгоритму до об'єктів довільної природи.

1.2 Властивості машини Тюрінга як алгоритму

На прикладі МТ добре простежуються властивості алгоритмів:

1. *Дискретність*. МТ може перейти до $(k+1)$-кроку тільки після виконання k-го кроку, тому що саме k-й крок визначає, яким буде $(k+1)$-й крок.

2. *Детермінованість*. У кожній клітині таблиці машини Тюрінга записаний лише один варіант дії. На кожному кроці результат визначається однозначно, отже, послідовність кроків розв'язання задачі визначається однозначно, тобто якщо МТ на вхід подають одне й те саме слово, то вихідне слово кожен раз буде одним і тим самим.

3. *Результативність*. Змістовно результати кожного кроку і всієї послідовності кроків визначаються однозначно, отже, машина Тюрінга, яка правильно розроблена, за кінцевого числа кроків перейде в стан q_k, тобто за кінцевого числа кроків буде отримано відповідь на питання, поставлене в задачі.

4. *Масовість*. Кожна машина Тюрінга визначена над усіма допустимими словами з алфавіту, у цьому і полягає властивість масовості. Кожна МТ призначена для розв'язання одного класу задач, тобто для кожної задачі розробляється своя (нова) машина Тюрінга.

5. *Зрозумілість*. На кожному кроці в клітину записується символ з алфавіту, автомат виконує один рух (вліво – Л, Left, вправо – П, Right, є нерухомим – Н, Stop), і МТ переходить в один з описаних станів [3].

1.3 Формальне визначення алгоритму

Основні властивості алгоритму: дискретність, детермінізм, масовість та результативність, дозволяють представити процес обчислення будь-якої числової функції за допомогою математичної машини. Ця машина за кінцеве число кроків за вихідними даними дозволяє обчислити шуканий числовий результат відповідно до заданих правил.

Така модель алгоритму була запропонована англійським математиком, логіком, інженером Аланом Тюрінгом наприкінці 1930-х років, що майже на два десятиліття випередило появу ЕОМ і стало їхнім теоретичним прообразом.

А. Тюрінг у 1936 р. описав схему деякої гіпотетичної (абстрактної) машини та формалізував правила виконання дій за допомогою опису роботи цієї машини, що можна розглядати як одне з перших формальних визначень алгоритму.

Поняття машини Тюрінга виникло в результаті прямої спроби розкласти відомі обчислювальні процедури на елементарні операції. Тюрінг навів низку аргументів на користь того, що повторення його елементарних операцій було б достатньо для проведення будь-якого можливого обчислення. Машина Тюрінга – це один із найвідоміших ідеальних пристроїв для обчислення функцій та один з перших апаратів, використовуваних у дослідженнях із теорії алгоритмів.

МТ є абстракцією, яку не можна реалізувати практично. Тому алгоритми для машини Тюрінга повинні виконуватися іншими засобами.

Основним наслідком формалізації алгоритмів із використанням машини Тюрінга є можливість доведення існування або не існування алгоритмів розв’язання різних задач.

Описуючи різні алгоритми для машин Тюрінга і доводячи реалізованість різноманітних композицій алгоритмів, Тюрінг переконливо довів різноманітність можливостей запропонованої ним конструкції, що дозволило йому виступити з такою тезою: «Будь-який алгоритм може бути реалізований відповідною машиною Тюрінга».

Довести тезу Тюрінга неможливо, оскільки в його формулюванні не визначено поняття «будь-який алгоритм». Його можна тільки обґрунтувати, подаючи різні алгоритми у вигляді машин Тюрінга. Було доведено, що клас функцій, обчислювальних на цих машинах, збігається з класом частково рекурсивних функцій.

МТ є найбільш загальною математичною моделлю «детермінованого перетворювача слів», тобто моделлю, за допомогою якої може бути обчислена будь-яка функція з множини слів в одному алфавіті в множину слів в іншому алфавіті.

2 МОДЕЛЮВАННЯ РОБОТИ МАШИНИ ТЮРІНГА

2.1 Структура машини Тюрінга

Алгоритмічний процес – це робота, виконувана деякою «абстрактною обчислювальною машиною». Кожна окрема МТ здатна виконувати тільки один алгоритм, для визначення якого можна користуватись терміном «програма МТ» – набір інструкцій, які спрощені до однотипної схеми. Всі МТ відрізняються за своїми програмами [4, 5].

МТ можна класифікувати у такий спосіб (рис. 2.1).

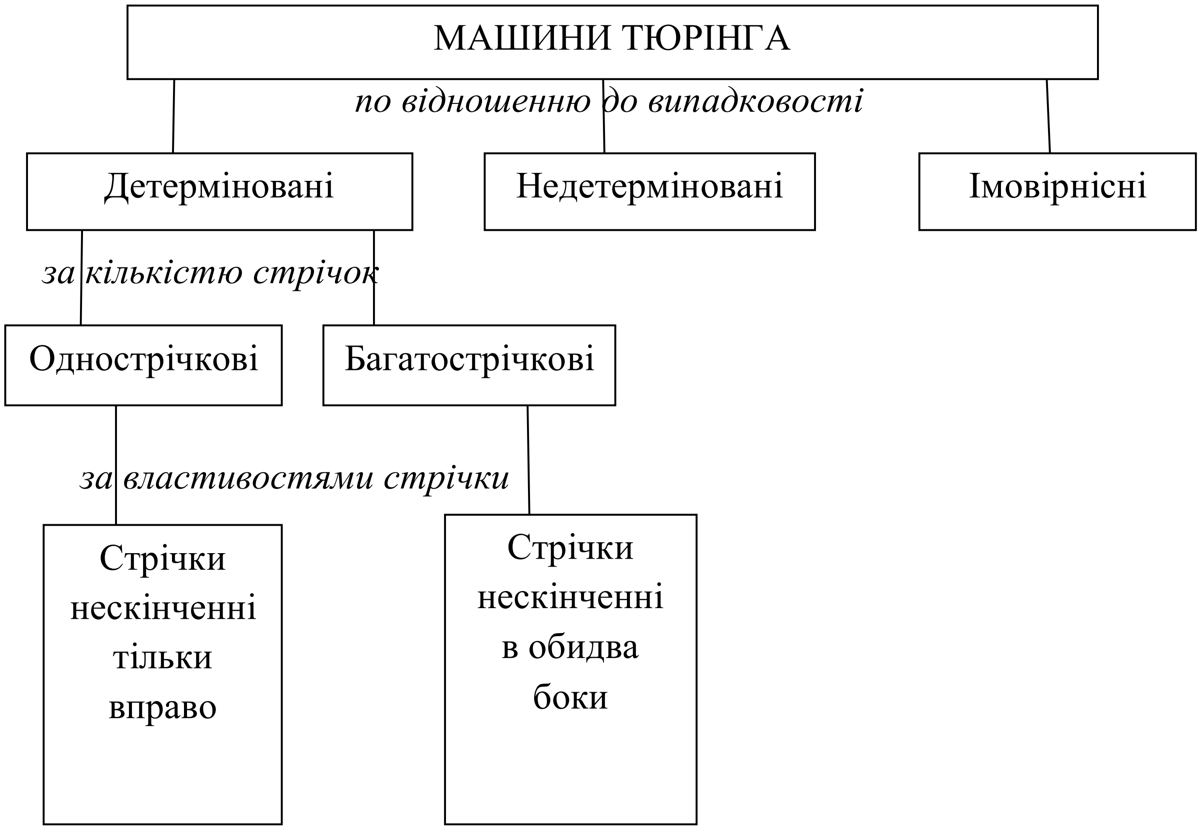


Рисунок 2.1 – Схема класифікації МТ

Моделлю процесу обчислення слугує детермінована одностріччкова машина Тюрінга (ДМТ), яка схематично зображена на рисунку 2.2

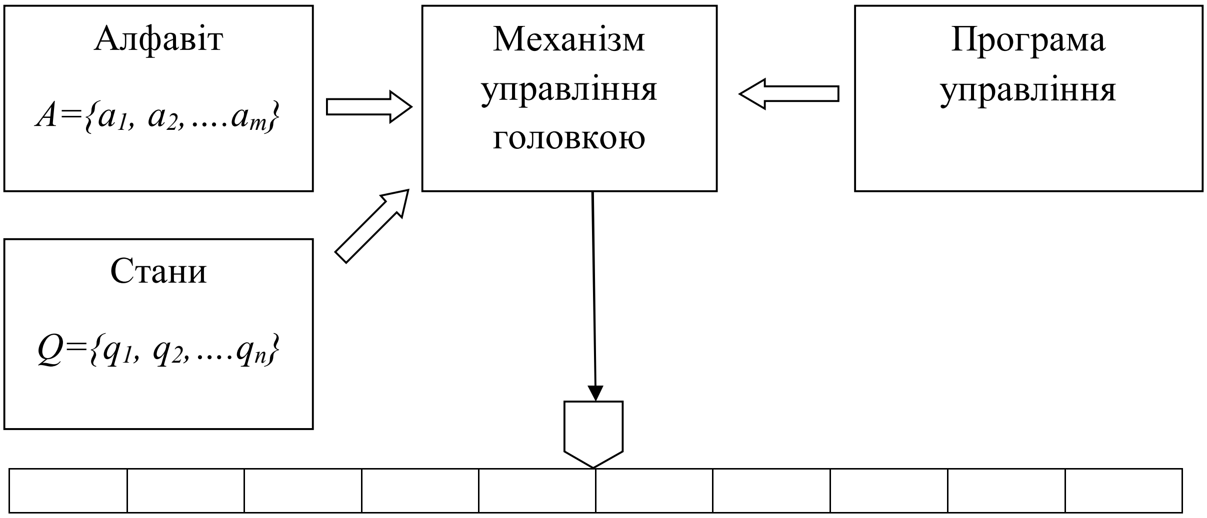


Рисунок 2.2 – Схематичне зображення детермінованої однострічкової машини Тюрінга

ДТМ складається з керувального пристрою з кінцевим числом станів, головки, яка може зчитувати та записувати символи, та необмеженої в обидва боки стрічки, яка розділена на нескінченну кількість однакових клітин, пронумерованих цілими числами: ……, –2, –1, 0, 1, 2, 3….

Інформаційна стрічка нескінченної довжини становить послідовність клітин, у кожну з яких записаний тільки один символ із множини символів алфавіту: $A=\{a_1, a_2,\ldots,a_m\}$. Послідовність символів на стрічці формує слово. Проміжок між словами також є символом множини *A*, наприклад, # □ *A*. У формальних граматиках множину *A* називають множиною термінальних символів. Інформаційна стрічка виконує функції зовнішньої пам’яті МТ.

Головка на кожному кроці роботи машини вказує на одну з клітинок. Крім цього, існує кінцева множина станів головки, до того ж на кожному кроці роботи машини головка перебуває в одному зі станів. У переході між кроками роботи головка машини може змінювати свій стан і разом із тим або записувати в клітинку, на яку вона вказує, символ алфавіту, або переміщатися вліво або вправо по стрічці на одну клітинку.

Головка оглядає тільки одну клітинку інформаційної стрічки, передає інформацію про її вміст в керувальний пристрій і за вказівкою останнього зберігає або ж змінює вміст клітинки.

Керувальний пристрій становить механізм, який на кожному кроці обчислення знаходиться в одній із множин станів: $Q = \{q_1, q_2,\ldots, q_n\}$. Залежно від стану q_i та зчитаного символу a_j керувальний пристрій надає команду на стирання або запис символу в клітинку, що оглядається, переводу управляючого пристрою в новий стан і переміщення головки на сусідню клітинку інформаційної стрічки. Тому стани керувального пристрою називають «пам’яттю машини Тюрінга», оскільки машина пам’ятає всі проміжні стани, які перевели її зі стану q_0 у стан q_i.

Із позиції формальних граматик множина символів, які описують стани керувального пристрою, є множиною нетермінальних символів. Серед усіх станів керувального пристрою варто виділити два: q_o – початковий стан («старт») і q_k – кінцевий стан («стоп») (у цьому випадку під *k* розуміється не числова змінна, а мнемонічний знак кінця), що полегшує складання протоколів машин Тюрінга, а також композицію декількох МТ. Для опису переміщень головки відносно інформаційної стрічки вводиться додатковий алфавіт: D={П, Л, С}, де П – означає переміщення головки вправо на одну клітинку інформаційної стрічки, Л – вліво на одну клітинку та С – зупинка переміщення головки.

МТ має в розпорядженні кінцеве число знаків (символів): $a_1, a_2, \ldots, a_m$, що утворюють зовнішній алфавіт, у якому кодуються відомості, що подаються в машину, а також ті, які виробляються в ній. У МТ обробка інформації, як і в комп'ютері, виконується в логічному блоці, який може перебувати в одному з кінцевої кількості станів: $q_1, q_2, \ldots q_n$. Блок має два вхідних канали: через один із них на кожній стадії роботи машини (у кожному такті) надходить знак із клітинки, яку оглядають, через інший – знак q_i того стану, який приписується блоку на певний такт. Через вихідний канал блок надсилає в клітинку, яку оглядає, відповідний «перепрацьований» знак a_j, який є однозначною функцією від сигналів $a_j q_i$, поданих на вхід (рис. 2.3).

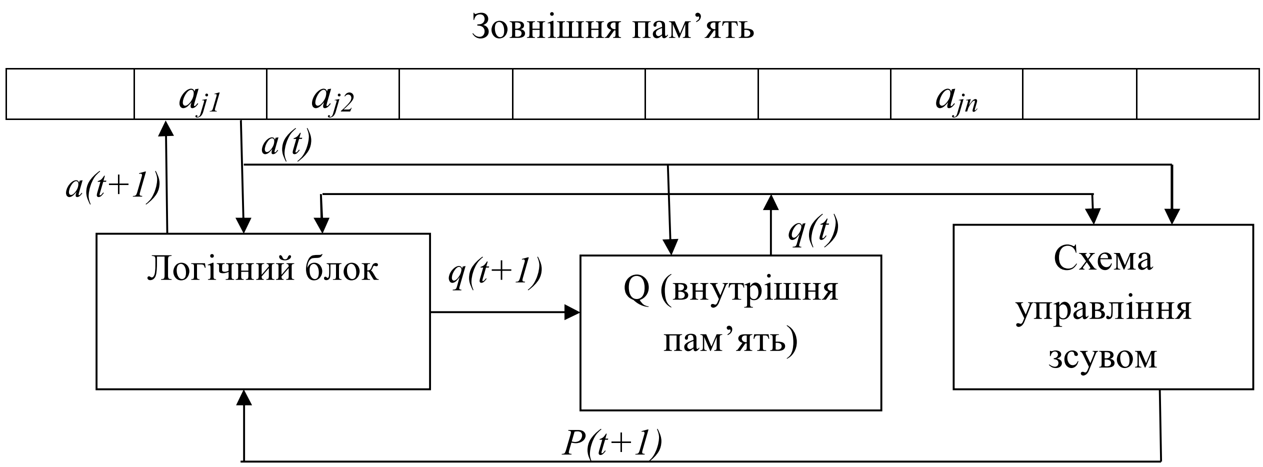


Рисунок 2.3 – Обробка інформації в МТ

Логічний блок реалізує функцію, яка ставить у залежність кожній парі знаків: a_i, q_n (кількість таких пар складає $k*m$) трійку знаків: a_j, D, q_t. Таку логічну функцію зручно подавати у вигляді прямокутної таблиці, стовпчики якої занумеровані знаками стану, а рядки – знаками зовнішнього алфавіту. У кожній клітинці таблиці записано відповідну вихідну трійку знаків. Таку таблицю можна називати функціональною схемою машини [4].

2.2 Конфігурація та протокол машини Тюрінга

Описом МТ є послідовність символів на інформаційній стрічці, положення зчитувальної/записувальної головки щодо клітинки інформаційної стрічки і поточний стан керувального пристрою. Такий опис називають конфігурацією машини Тюрінга:

$$K = \alpha\, q_i\, \beta,$$

де α – слово (або послідовність символів), розташоване зліва від головки;

β – слово (або послідовність символів), розташоване під і праворуч від головки;

q_i – поточний стан МТ.

Символ «α», що знаходиться в клітинці безпосередньо під головкою, є першим символом слова β.

Іншими словами, конфігурацією МТ називається її поточний стан, поточний стан інформаційної стрічки та місце розташування головки. Під час роботи МТ у кожному такті виконується зміна конфігурацій, тобто роботою МТ можна називати перехід від однієї конфігурації до іншої за допомогою команд цієї машини.

До незаключної конфігурації може бути застосована тільки одна команда, яка переводить машину в нову конфігурацію. У такий спосіб реалізується дискретність і детермінізм алгоритмічного процесу.

Робота алгоритму виконується дискретними кроками, на кожному кроці виконується одна з команд, які складають алгоритм. Кожному такому кроку відповідає цілком визначена активна клітинка на інформаційній стрічці.

Робота МТ полягає в багаторазовому повторенні наступного циклу елементарних дій:

– дія перша полягає у зчитуванні символу a_j, що знаходиться під зчитувальною головкою;

– дія друга полягає в пошуку команди, яка відповідає поточному стану керувального пристрою q_i і зчитаному символу a_j , тобто

$$q_i a_j \rightarrow q_{i+1} a_m\, D;$$

– дія третя полягає у виконанні знайденої команди, тобто записі в клітинку, що оглядається, символу a_m, переході керувального пристрою в стан q_{i+1} та переміщенні головки на сусідню клітинку інформаційної стрічки: $D=\{П, Л, С\}$.

Ці три дії становлять одну елементарну команду, яку можна озвучити у такий спосіб: що потрібно записати в поточну клітинку, у який стан повинна перейти головка, куди повинна переміститись головка. Послідовність команд для реалізації процесу обчислення становить програму алгоритмічного процесу або протокол машини Тюрінга. Варто зазначити, що будь-які дві команди не можуть мати однакову пару поточного стану q_i і зчитуваного символу a_j, тобто пари ($q_i a_j$). МТ зупиняється тільки в тому випадку, якщо на черговому кроці керувальний пристрій генерує стан q_k. Результатом роботи МТ буде заключне слово на інформаційній стрічці.

Говорять, що машина здатна вирішувати деякий клас задач, якщо вона застосована лише до інформації, що відображує в визначеному коді умови будь-якої окремої задачі певного типу та переробляє її в інформацію, яка відображує в тому самому коді розв’язок цієї задачі.

2.3 Представлення машини Тюрінга

Роботу МТ зручно описувати за допомогою протоколу МТ (програмою алгоритмічного процесу), таблицею відповідності (функціональною схемою машини) та графом [5].

2.3.1 Представлення машини Тюрінга сукупністю команд

Кожна команда машини містить не більше однієї дії. Фактично команди виглядають так:

$a_i \to q_j$ – у клітинку, що оглядається, записати символ a_i, переміститися вправо (до наступної клітинки) та перейти в стан q_j;

$a_i \leftarrow q_j$ – у клітинку, що оглядається, записати символ a_i, переміститися вліво (до наступної клітинки) та перейти в стан q_j;

$a_i S\, q_j$ – у клітинку, що оглядається, записати символ a_i, зупинитися та перейти в стан q_j.

Наприклад, конфігурацію із внутрішнім станом q_i, у якій на стрічці записано слово $abcde$, а головка вказує на символ d, можна записати у вигляді: $abc\mathrm{q_i}de$. Стандартною початковою конфігурацією назвемо конфігурацію вигляду $q_1\alpha$, тобто конфігурацію, що містить початковий стан, у якому головка вказує на крайній лівий символ слова, написаного на стрічці. Аналогічно стандартною заключною конфігурацією назвемо конфігурацію вигляду $q_k\,\beta$.

Для прикладу розглянемо таку систему команд машини: $q_2a_5 \to q_3a_4R$ і $q_3a_1L \to q_4a_2$.

У ході виконання програми буде реалізовано такий ланцюжок команд: $q_2a_5a_1a_2 \to a_4q_3a_1a_2 \to q_4a_4a_2a_2$ і отримано такий результат: $q_2a_5a_1a_2 \to q_4a_4a_2a_2$.

Далі наведемо пояснення щодо роботи прикладу.

1. На інформаційній стрічці в трьох клітинках записано слово $a_5\ a_1\ a_2$.

2. Ліва частина першої команди: $q_2a_5 \to q_3a_4R$, означає, що головка вказує на клітинку, в якій записано символ a_5, якщо головка перебуває в стані q_2 (q_2a_5). У ході виконання команди в стані q_3 в цю саму клітинку замість символа a_5 буде записано символ a_4, після чого головка переміститься вправо і вкаже на символ a_1.

3. Друга команда $q_3a_1L \rightarrow q_4a_2$, означає, що за перебування в стані q_3 головка вказує на клітинку з символом a_1, який за перебування головки в стані q_4 замінюється на символ a_2, після чого головка переміщується вліво.

У результаті виконання послідовності команд отримано таку конфігурацію МТ: $q_4a_4a_2a_2$.

Як другий приклад розглянемо сукупність команд МТ, яка інвертує вхідний ланцюжок, записаний із використанням нулів і одиниць.

Нехай алфавіт машини Тюрінга задано множиною: $A = \{0, 1, \varepsilon\}$, де символ ε відповідає порожній клітинці, а число станів пристрою керування задано в вигляді множини: $Q = \{q_0, q_1, q_k\}$.

Якщо, наприклад, початкова конфігурація має вигляд $q_0$110011, то кінцева конфігурація після завершення операції інвертування повинна мати такий вигляд q_k001100. Для розв’язання задачі машиною буде створена така послідовність команд:

$q_0 1 \rightarrow q_0 0R$, $q_1 0 \rightarrow q_1 0L$,

$q_0 0 \rightarrow q_0 1R$, $q_1 1 \rightarrow q_1 1L$,

$q_0 \varepsilon \rightarrow q_1 \varepsilon L$, $q_1 \varepsilon \rightarrow q_k \varepsilon R$.

У стандартній початковій конфігурації головка стоїть над першим символом зліва, пристрій управління знаходиться в початковому стані. На наступному такті МТ, не змінюючи свого стану, замінює символ 0 на символ 1 або 1 на 0 і зсувається вправо на один символ. Після перегляду всього ланцюжка виявиться, що головка вказує на символ, який означає порожню клітинку. У цьому випадку машина Тюрінга переходить у новий стан і зсувається вліво на один символ. На подальших тактах керувальний пристрій не змінює свого стану, залишає без зміни символ, на який вказує головка і переміщується вліво доти, поки не натрапить на порожню клітинку, після чого МТ переходить у кінцевий стан і переміщується вправо на один символ, переходячи в стандартну заключну конфігурацію.

2.3.2 Представлення машини Тюрінга у вигляді таблиці відповідності

Сама по собі МТ ніякої роботи не виконує. Для її роботи потрібно написати програму, яка записується в вигляді таблиці, де зліва в першій колонці перераховуються всі стани, у яких може перебувати автомат, а в першому рядку зверху – усі символи, які автомат може розпізнавати на стрічці. Потрібно зауважити, що в розглянутій у цьому виданні програмі – інтерпретаторі ALGO 2000 таблиця відповідності є транспонованою. Символи і стани, які потрібно вказувати в таблиці, визначає автор програми. На перетинах (у клітинках) вказуються такти (дії), які повинен виконувати автомат під час його перебування у відповідному стані і розпізнавати на стрічці відповідний символ.

За табличного опису кожен рядок має ім’я поточного і початкового станів машини, а стовпчик – ім’я символу зовнішньої пам’яті, а отже, елементами таблиці є праві частини команд q_ja_iD (табл. 2.1) [6].

Таблиця 2.1 – Таблиця відповідності

Ім'я поточного та початкового станів машини / ім'я символу зовнішньої пам'яті	Символи алфавіту: $A=\{a_1, a_2, \ldots, a_m\}$			
	a_i	a_k		a_m
q_1	$q_1\ a_j D$	..	..	..
q_2	..			..
...				
q_i	..	$q_j\ a_i D$	..	
..	..			..
q_n	..			$q_k a_z D$

У цілому таблиця визначає дії МТ за всіх можливих конфігурацій і тим самим повністю визначає її поводження. Описати алгоритм у вигляді МТ означає надати таку таблицю.

Часто МТ визначають як таку, що складається зі стрічки, автомату та програми, тому за різних програм отримують різні МТ. Можна вважати, що МТ одна, але вона може виконувати різні програми [7].

Таблична форма опису машини є більш компактною, тому дозволяє застосувати матричні методи аналізу для оптимізації структури алгоритму.

Розглянемо створення таблиці відповідності на прикладі сукупності команд МТ, яка інвертує вхідний ланцюжок, записаний з використанням символів нулів і одиниць, який наведено в підрозділі 2.3.1 (табл. 2.2).

Таблиця 2.2 – Таблиця відповідності для операції інвертування програмою МТ

Символи/стан	0	1	ε
q_0	$q_0 1R$	$q_0 0R$	$q_0 \varepsilon L$
q_1	$q_1 0L$	$q_1 1L$	$q_к\ \varepsilon L$

Цю таблицю називають функціональною схемою цієї машини, оскільки вона відіграє роль інструкції (програми) для МТ. З таблиці, зокрема, видно якими є зовнішній і внутрішній алфавіти машини.

2.3.3 Представлення машини Тюрінга у вигляді графа

За представлення МТ у вигляді графа вершинами є стани керувального пристрою, а дугами – переходи в ті стани, які передбачені командою (рис. 2.4) [7].

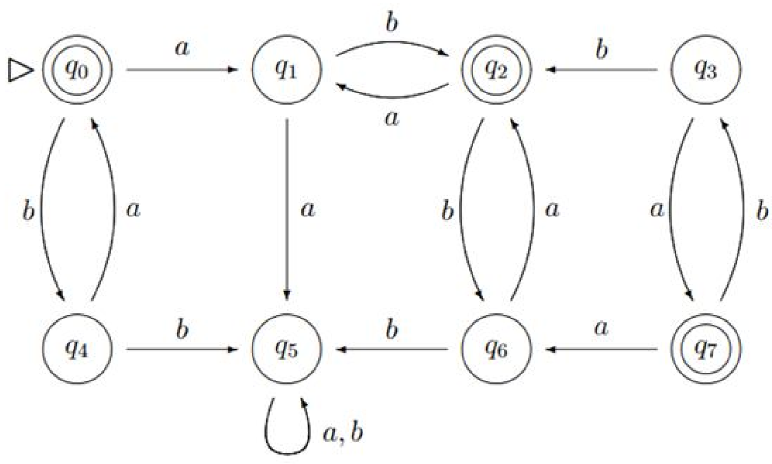


Рисунок 2.4 – Представлення МТ у вигляді графа

Початкова та кінцева вершини графа зазвичай виділяються; на рисунку 2.4 вони позначені чорним кружечком. Якщо на черговому такті роботи машини Тюрінга символ на стрічці не змінюється, то в правій частині команди його можна не писати.

2.4 Відмінність машини Тюрінга від сучасного комп’ютера

До сьогоднішнього дня всі обчислювальні машини в деякому розумінні базуються на ідеї Тюрінга: їхня пам’ять фізично складається з бітів, кожен із яких містить або 0 або 1. Крім того, мікропрограмне управління успадкувало від цих абстрактних машин і програму, вміщену в «постійну пам’ять», і значною мірою структуру команди.

Кожен комп’ютер може моделювати машину Тюрінга (операції перезапису клітинок, порівняння і переходу до іншої сусідньої клітинки з урахуванням зміни стану машини). Отже, комп’ютер може моделювати алгоритми в будь-якому формалізмі, що свідчить про те, що всі комп’ютери (незалежно від потужності, архітектури тощо) є еквівалентними з погляду принципової можливості розв’язання алгоритмічних задач.

Принципова різниця між МТ і обчислювальними машинами полягає в тому, що її запам’ятовувальний пристрій становить нескінченну стрічку, яка не знає можливості її фізичної реалізації. Стрічка розділена на клітинки, в кожній із яких може бути записаний один із символів кінцевого алфавіту: $A = \{a_0, a_1, \dots, a_m\}$, який називають вхідним алфавітом МТ. У процесі роботи МТ може бути заповнене кінцеве число клітинок. Зчитувальна головка в кожен момент часу оглядає клітинку стрічки, залежно від символу в ній та стану керувального пристрою, записує в клітинку новий символ або залишає його без зміни, переміщується на клітинку вліво, вправо або залишається на місці. Водночас керувальний пристрій переходить у новий стан або залишається в попередньому. Серед станів керувального пристрою виділено початковий – q_0 і заключний – $q_к$ стани.

Таким чином, за один такт роботи МТ може зчитати символ, записати замість нього новий або залишити його без зміни і перемістити головку на одну клітинку вліво або вправо, або залишити її на місці.

Можна констатувати такі відмінності МТ порівняно з ЕОМ:

1. Розкладання процесу в МТ на прості елементарні операції доведено в відомому сенсі до граничної можливості. Наприклад, операція додавання, яка існує в ЕОМ як одинична елементарна операція, у МТ сама розкладається на ланцюжок ще більш простих операцій. Це значною мірою подовжує процес, який реалізовується в МТ, але разом із тим логічна структура процесу спрощується і набуває придатного для теоретичних досліджень стандартного вигляду.

2. Зовнішня пам’ять в МТ зображується нескінченою стрічкою, яка розбита на клітинки. Жодна реально існуюча машина не може мати нескінченну стрічку, і в цьому сенсі МТ становить лише ідеалізовану схему, яка відображає потенційну можливість збільшення обсягу пам’яті.

Така ідеалізація виправдовується зв’язком між поняттям алгоритму та поняттям машини з потенційно необмеженою пам’яттю [4].

Систему команд МТ можна інтерпретувати і як опис роботи конкретного механізму, і як програму. Природно поставити завдання побудови МТ, яка б реалізувала алгоритм відтворення роботи МТ – універсальної машини *U*. Універсальна МТ за заданим кодом довільної машини Тюрінга *T* та кодуванням вхідного ланцюжка *x* буде моделювати поведінку машини *T* із вхідним ланцюжком *x*. Існування універсальної машини Тюрінга означає, що систему команд будь-якої машини Тюрінга можна уявити двояко: або як опис роботи конкретного пристрою машини *T*, або як програму для універсальної машини *U*. Для інженера, який проєктує систему управління, будь-який алгоритм управління може бути реалізований або апаратурною побудовою відповідної схеми, або у вигляді програми для універсальної управляючої ЕОМ.

МТ є розширенням моделі кінцевого автомату та здатна імітувати (за наявності відповідної програми) будь-яку машину, дія якої полягає в переході від одного дискретного стану до іншого.

Машини Тюрінга вміють виконувати будь-які обчислення, якими б складними вони не були, незважаючи на простоту конструкції МТ. На відміну від Бебіджа та Лейбніца, які створили реальні машини для вирішення вузького класу задач, Тюрінг запропонував просту та зрозумілу теоретичну модель обчислювальної машини, яка здатна виконувати будь-які обчислення.

Головна мета вивчення та використання МТ – вивчення властивостей алгоритмів і проблеми алгоритмічної вирішувальності задачі.

3 ПРОГРАМНИЙ ІНТЕРПРЕТАТОР ALGO 2000

Програмний інтерпретатор ALGO 2000 – це інтерпретатор машини Поста та машини Тюрінга [8]. Під поточною машиною розуміється машина Поста або машина Тюрінга залежно від того, яка з них є активною в конкретний момент часу. Файл машини Поста має розширення **.pst*, а файл МТ –**.tur*

Порядок роботи МТ задається в вигляді таблиці, у яку в кожний стовпчик верхнього рядка заносяться символи внутрішнього алфавіту, а в кожний рядок першого стовпчика – символи зовнішнього алфавіту. У клітинках на перетині інших стовпчиків та рядків розміщуються команди. Якщо на перетині будь-якого рядка та будь-якого стовпчика отримано пусту клітинку, то це означає, що в конкретному внутрішньому стані цей символ зустрітися не може.

3.1 Інтерфейс машини Тюрінга, реалізованої в програмному інтерпретаторі ALGO 2000

ALGO 2000 – навчальна модель універсального виконавця (абстрактної обчислювальної машини), яка запропонована Тюрінгом як уточнення поняття алгоритму. Уточнення поняття алгоритму машиною Тюрінга полягає в тому, що алгоритмічні процеси – це процеси, які може здійснити відповідно влаштована машина.

До складу інтерфейсу ALGO 2000 входять твкі елементи (рис. 3.1):

1) рядок меню;
2) панель інструментів;
3) поле задання умови задачі;
4) інформаційна стрічка;
5) поле задання зовнішнього алфавіту;
6) таблиця відповідності МТ;
7) поле коментарів щодо розв’язання задачі.

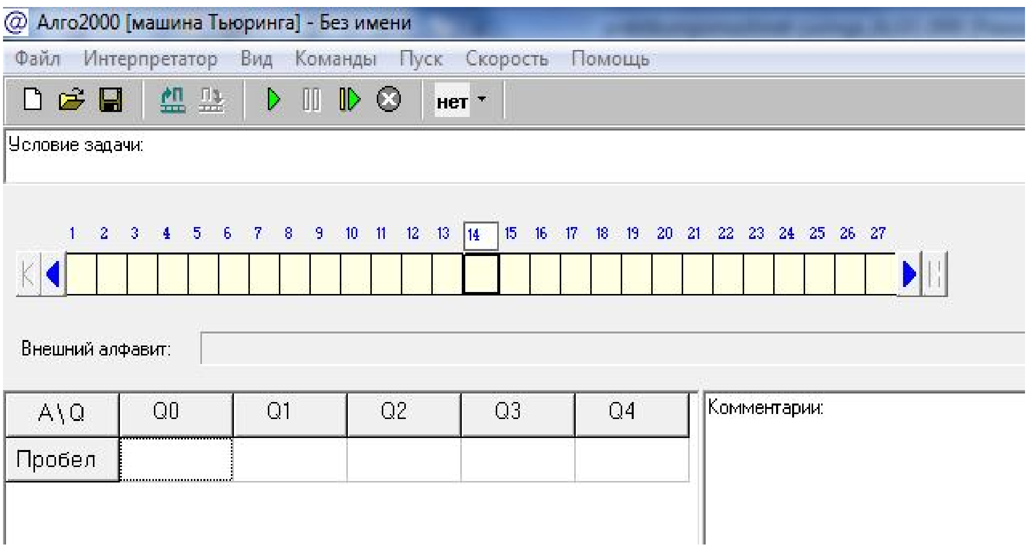


Рисунок 3.1 – Інтерфейс МТ

Зовнішній алфавіт – це редактор для вводу, відображення та зберігання символів зовнішнього алфавіту МТ, який необхідно заповнювати перш за все під час написання алгоритму МТ. Кожен символ можна навести тільки один раз. Між символами можуть бути пробіли. Символ «пробіл» у зовнішньому алфавіті завжди існує за замовчуванням. Для відображення редактора «зовнішній алфавіт» необхідно подати команду ВИД/ЗОВНІШНІЙ АЛФАВІТ.

Після закінчення вводу зовнішнього алфавіту формується перший стовпчик таблиці, який заповнюється символами зовнішнього алфавіту в такому самому порядку, як у редакторі «зовнішній алфавіт».

Стрічка складається з 999 клітинок із нумерацією від 1 до 999. Клітинка з жирною рамкою, що знаходиться в центрі стрічки, називається кареткою.

У таблицю МТ безпосередньо записується алгоритм розв’язання задачі. Перед вводом алгоритму необхідно в клітинки першого стовпчика записати символи зовнішнього алфавіту, а в клітинки верхнього рядка записати символи внутрішнього алфавіту: q_1, $q_2 \ldots q_n$.

Стандартною початковою конфігурацією є конфігурація вигляду $q_1\alpha$, тобто конфігурація, що вміщує початковий стан, в якому головка вказує на крайній лівий символ слова, написаного на стрічці.

Стандартна заключна конфігурація вигляду $q_k\beta$, у якій головка вказує на правий крайній символ слова.

Робота МТ розпочинається зі стандартної початкової конфігурації та закінчується стандартною кінцевою конфігурацією.

3.2 Основні прийоми роботи в програмному інтерпретаторі ALGO 2000

Під час представлення МТ у вигляді таблиці відповідності в клітинках розташовуються команди МТ. Формат команд МТ має такий вигляд: *aKq*,
де *a* – новий зміст поточної клітинки (новий символ зовнішнього алфавіту, який заноситься в поточну клітинку);
K – команда механізму протягування стрічки МТ («вліво», «вправо», «зупинитися»);
Q – новий внутрішній стан МТ.

Робота МТ полягає в тому, що каретка переміщується вздовж стрічки та друкує або стирає символи. Для того щоб МТ почала працювати, необхідно задати зовнішній алфавіт, тобто вказати символи, які до нього входять, а також задати програму та деякий стан машини, тобто необхідно розташувати символи зовнішнього алфавіту на клітинах стрічки, зокрема можна всі клітини залишити пустими та поставити каретку навпроти однієї з клітинок.

Робота машини на основі заданої програми здійснюється у такий спосіб.

Припустимо, що МТ в конкретний момент часу перебуває у внутрішньому стані q_i, а в клітинці, що оглядається, знаходиться символ a_j. У результаті машина переходить до виконання команди aKq у клітинці, на перетині стовпчика q_i та рядка a_j:

1) у поточну клітинку стрічки заноситься новий символ a;

2) виконується зсув каретки вліво або вправо, або ж зупинення роботи машини;

3) машина переходить у новий стан q.

Для вводу команди в клітинку необхідно:

1) ввести символ зовнішнього алфавіту;

2) ввести одну з команд стрічкопротяжного механізму:

– символ «<» означає переміщення каретки вліво;

– символ «>» означає переміщення каретки вправо;

– символ підкреслення «_» означає пробіл, тобто пусту клітинку;

– символ знак оклику «!» означає зупинення програми (зображується у вигляді ) ;

3) ввести номер нового внутрішнього стану (вводиться тільки цифра, літера Q не вводиться).

У випадку неправильного вводу команди в клітинці жодного запису не з’явиться. Для видалення або додавання стовпчиків чи рядків необхідно використовувати відповідні пункти головного меню або піктограми панелі інструментів. Запуск програми МТ починається зі стану q_0 [9–10].

Автомат машини Тюрінга в процесі роботи виконує такі дії:

– записує символ зовнішнього алфавіту в клітину (і в пусту клітину), змінюючи символ, який знаходиться в цій клітині;

– переміщується на одну клітину вліво або вправо;

– змінює свій внутрішній стан.

Користь машини Тюрінга проявляється в наглядній демонстрації суті алгоритмічних обчислень. Розроблені практичні заняття допоможуть студентам зрозуміти, що машина Тюрінга здатна імітувати всіх виконавців за допомогою задання правил переходу, які реалізують процес покрокового обчислення, кожен крок якого є елементарним.

4 ПРАКТИЧНІ ЗАНЯТТЯ

4.1 Практичне заняття 1
Розв'язання простих задач за допомогою машини Тюрінга з використанням програмного інтерпретатора ALGO 2000

Мета заняття – формування знань та вмінь щодо складання програм для МТ.

Завдання на виконання практичного заняття

1. Побудувати МТ, зовнішній алфавіт якої складається з символів «*», «# », «&» так, щоб уведені символи «# » та «&» перетворювались на символи «*» [9]. Водночас послідовність повинна починатися з символів «*», а закінчуватися символами «&».

2. Побудувати МТ, яка починає рух вліво від довільної клітинки, знаходить першу за такого переміщення клітинку, що містить символ 1 (якщо такий зустрінеться на стрічці), і зробивши один крок вправо, зупиниться на сусідній клітині. Вміст інформаційної стрічки не змінюється.

3. Побудувати МТ, яка починає роботу від довільної клітинки, що містить символ 1, переміщується вліво доти, поки не пройде підряд п'ять символів 0. Головка повинна зупинитися на першій клітинці зліва після п'яти символів 0, надрукувавши в цій клітинці символ 1. Вміст інших клітинок стрічки залишається незмінним.

4. Скласти функціональну схему МТ, яка зможе розташувати в порядку зростання цифри, введені в три клітинки інформаційної стрічки в довільному порядку. Каретка розташована на крайній лівій цифрі.

Загальні положення

Будь-яка МТ пов'язана з двома кінцевими алфавітами: алфавітом вхідних символів A і алфавітом станів Q. Алфавіт A називається зовнішнім, а алфавіт Q – внутрішнім. З різними машинами Тюрінга можуть бути пов'язані різні алфавіти A і Q.

Вхідне слово розміщується на стрічці – по одному символу в розташованих підряд клітинках. Ліворуч і праворуч від вхідного слова знаходяться тільки порожні клітинки. До алфавіту A завжди входить порожній символ «пробіл» – ознака того, що клітинка є порожньою.

Автомат може рухатися вздовж стрічки вліво або вправо, читати вміст клітинок і записувати в клітинки символи.

Робота МТ полягає в повторенні такого циклу дій, які є спільними для будь-якої МТ:

1) зчитування символу з клітинки, яку оглядає головка;

2) відшукування застосовуваної команди;

3) виконання відшуканої команди.

Разом із тим вважається, що в програмі не існує двох таких команд, перші два символи яких були б однаковими.

МТ зупиняється в тому і тільки в тому випадку, коли жодна з команд її програми не є застосовуваною.

Результатом роботи МТ після її зупинення є слово, записане на стрічці. Сама ж машина Тюрінга стає алгоритмічною системою, вона обробляє заздалегідь записане на стрічці слово або нескінченно довго, або зупиняється після скінченного числа кроків. У першому випадку вважається, що алгоритм, який виконується машиною, є незастосовуваним до вхідного слова *p*, а в другому випадку слово, яке залишається на стрічці після зупинення машини, приймається як вихідне, до якого машина перетворює задане вхідне слово *p*.

Порядок виконання практичного заняття

Приклад 1

Написати програму (функціональну схему), яка надає можливість МТ збільшувати на один кількість символів «*» у послідовності символів «*», що введені на інформаційну стрічку. Разом із тим потрібно врахувати, що головка для зчитування розташована на символі «*», що міститься справа.

1.1 Подати після запуску програми команду меню ІНТЕРПРЕТАТОР/МАШИНА ТЮРІНГА.

1.2 Задати зовнішній алфавіт у вигляді символу «*». Для цього ввести в поле завдання зовнішнього алфавіту символ «*», натиснути в таблиці МТ клітинку з назвою «Пробіл» (рис. 4.1).

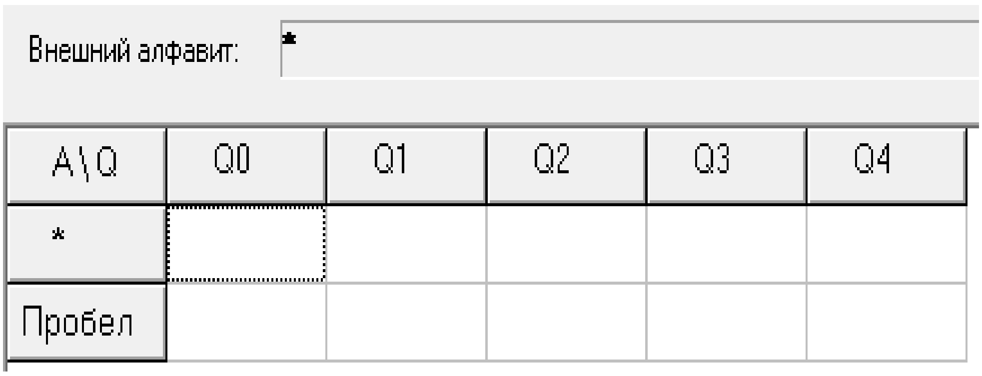


Рисунок 4.1 – Уведення символу зовнішнього алфавіту

1.3 Заповнити таблицю МТ командами (рис. 4.2).

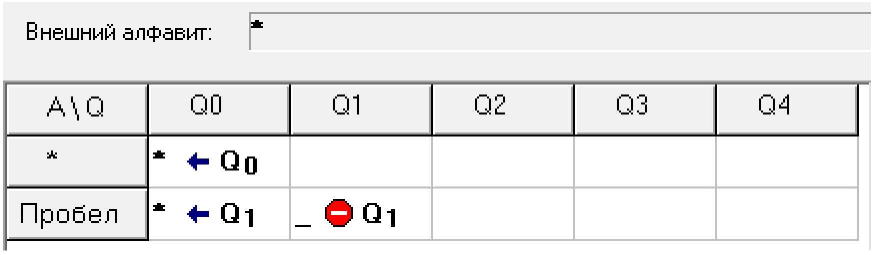


Рисунок 4.2 – Таблиця відповідності

1.4 Задати на інформаційній стрічці послідовність, що складається з чотирьох символів «*» (рис. 4.3).

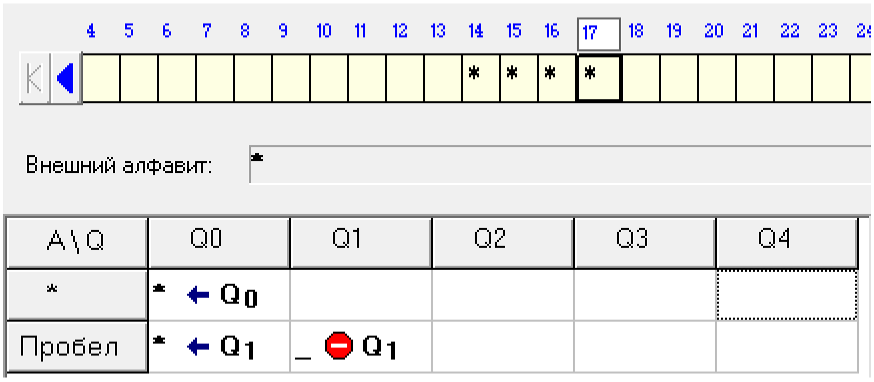


Рисунок 4.3 – Уведена послідовність символів та інструкцій для МТ

1.5 Запустити програму на виконання командою ПУСК/ЗАПУСТИТИ.

У вигляді послідовностей рисунків зобразимо процес зміни інформації на стрічці під час роботи МТ, яка описана функціональною схемою, зображеною у вигляді таблиці відповідності (рис. 4.3).

На першому кроці логічний блок перебуває в стані q_0 та оглядає праву клітинку, в яку записано один із символів «*». Згідно з таблицею відповідності в цю клітинку записується символ «*» (тобто в цьому випадку не виконується заміна вмісту клітинки, що оглядається), виконується зміщення блоку вліво та перехід в такий самe стан q_0 (рис. 4.4).

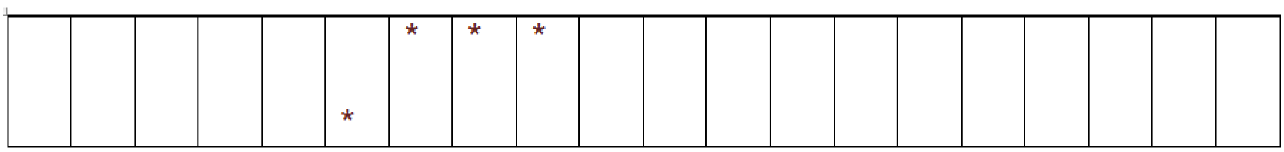


Рисунок 4.4 – Перший крок роботи логічного блоку

Аналогічні дії буде виконано ще три рази, в результаті чого логічний блок досягне пустої клітинки (рис. 4.5).

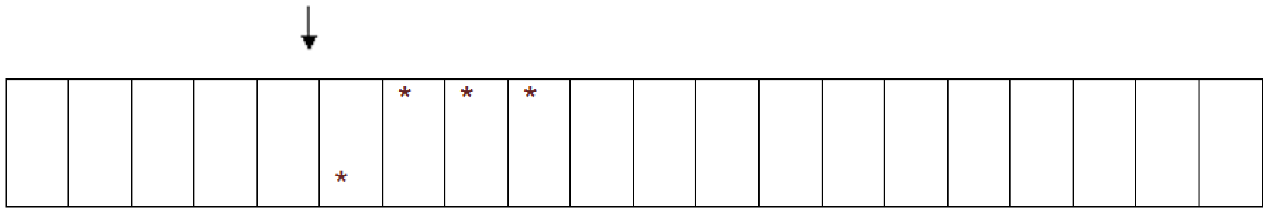


Рисунок 4.5 – Перегляд уведених символів на інформаційній стрічці без зміни вмісту кожної з клітинок

Разом із тим логічний блок оглядає клітинку з символом «Пробіл», записуючи в неї символ «*» (згідно з програмою). Після чого МТ припиняє свою роботу (рис. 4.6).

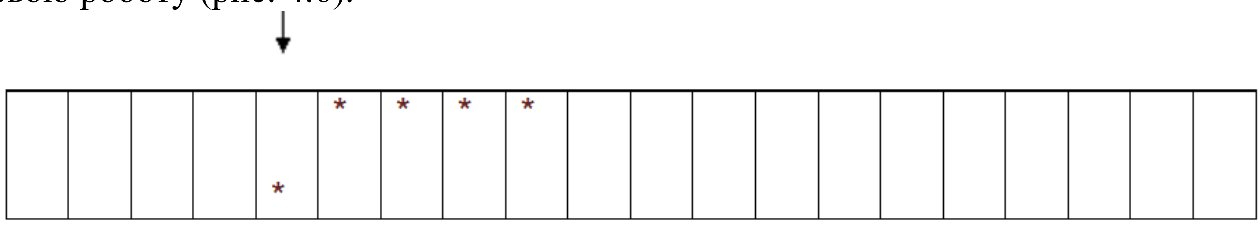


Рисунок 4.6 – Запис у пусту клітинку ще одного символу «*»

У результаті роботи МТ кількість символів «*» збільшилася на один, про що свідчить результат роботи програми (рис. 4.7).

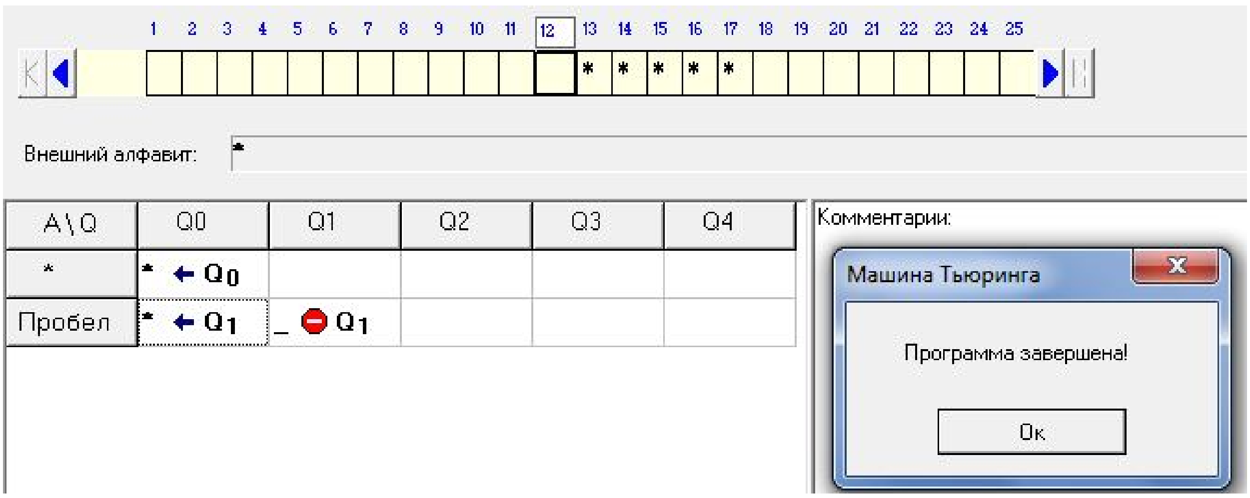


Рисунок 4.7 – Додання символу «*» до існуючої послідовності символів

1.6 Запустити програму з крайньої лівої клітинки та пояснити результат роботи, виконаної нею в такому випадку.

Зі структури наведеної програми видно, що вона приписує до послідовності введених символів ще один символ, тобто обчислює функцію інкремента.

Приклад 2

Написати програму, яка замінює у введеній послідовності з декількох символів «+» кожен із символів «+», розташованих на непарному місці, на символ «–», якщо введена парна кількість символів «+», і кожен із символів «+», розташованих на парному місці замінюється на символ «–», якщо введена непарна кількість символів «+» [10].

Водночас потрібно враховувати, що головка для зчитування розташована на першому лівому символі «+».

2.1 Подати команду меню ІНТЕРПРЕТАТОР/МАШИНА ТЮРІНГА.

2.2 Задати зовнішній алфавіт у вигляді символу «+» та символу «–». Для цього ввести в поле задання зовнішнього алфавіту символ «+», натиснути в таблиці МТ на клітинку з назвою «Пробіл», після чого ввести символ «–» (рис. 4.8).

Внешний алфавит: +-

A\Q	Q0	Q1	Q2	Q3	Q4
+					
-					
Пробел					

Рисунок 4.8 – Уведення символів зовнішнього алфавіту

2.3 Заповнити таблицю МТ командами (рис. 4.9).

Внешний алфавит: +-

A\Q	Q0	Q1	Q2	Q3	Q4
+	+ → Q_0	+ ← Q_2	- ← Q_1		
-	- → Q_0				
Пробел	_ ← Q_1	_ ⛔ Q_2	_ ⛔ Q_2		

Рисунок 4.9 – Таблиця відповідності

Команди стану q_0 дозволяють проглянути введену послідовність символів із зовнішнього алфавіту з послідовним переміщенням вправо: $+q_0 \to + \ q_0$, а після досягнення першої пустої клітинки розпочати рух вліво: $\_ \leftarrow q_1$.

За допомогою команд $+\leftarrow \ q_2$ та $\_\leftarrow \ q_1$ виконується заміна кожного першого символу послідовності «+» на символ «–».

Команда «пустий символ $q_1 \to$ пустий символ зупинка q_2» виконує зупинення програми.

2.4 Задати на інформаційній стрічці послідовність, що складається з парної кількості символів «+», наприклад, «++++++» (рис. 4.10).

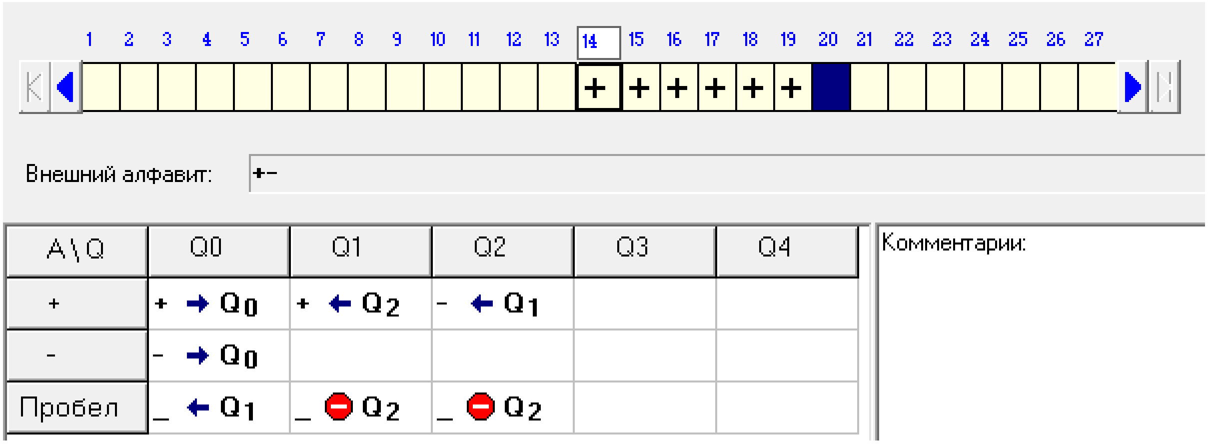


Рисунок 4.10 – Варіант уведення послідовності символів «+»

2.5 Запустити програму для виконання командою ПУСК/ЗАПУСТИТИ. Результат виконання програми наведено на рисунку 4.11.

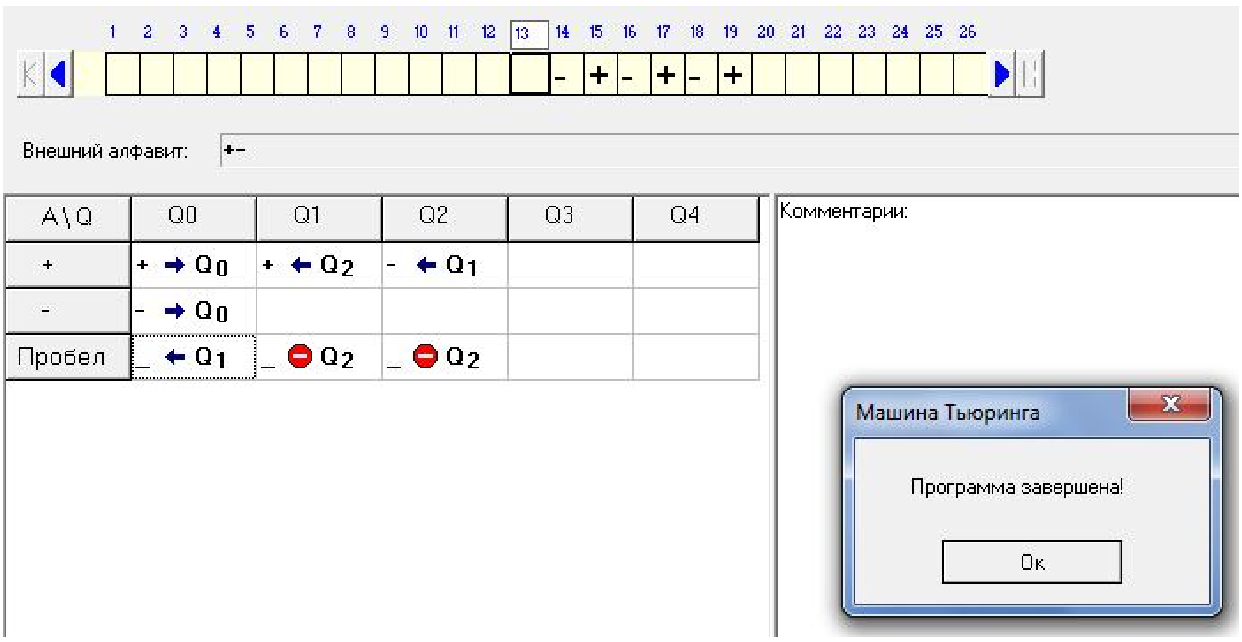


Рисунок 4.11 – Заміна кожного символу «+», розташованого на непарному місці, на символ «–»

2.6 Задати на інформаційній стрічці послідовність, що складається з непарної кількості символів «+», наприклад, « +++++».

Проаналізувати отримані результати.

2.7 Запустити програму з крайньої правої клітинки з символом «+».

Проаналізувати отримані результати.

Приклад 3

Написати програму, яка б замінювала введені на інформаційній стрічці одиниці на зірочки, врахувавши при цьому те, що головка для зчитування розташована на першій введеній одиниці.

3.1 Задати зовнішній алфавіт у вигляді цифри 1 та символу «*».

3.2 Заповнити таблицю МТ командами.

Для цього необхідно спочатку записати команди в вигляді:

1) $1q0 \to *q1$ – знаходження початку числа в стані q_0 та виконання заміни цифри 1 на символ «*». Каретка зрушується вправо;

2) $1q1 \to *q1$ – знаходження кінця числа в стані q_1 та заміна всіх цифр 1 на символ «*». Каретка зрушується вправо;

3) пустий символ $q0 \to$ пустий символ $q1$ – знаходження пустого символу, після чого на наступній пустий клітинці буде виконано зупинку програми. На цьому кроці виконується досягнення кінця послідовності. Каретка зрушується вправо;

4) пустий символ $q1 \to$ пустий символ стоп $q1$ – зупинення програми.

На рисунку 4.12 наведено заповнену таблицю для МТ у програмі ALGO 2000.

A\Q	Q0	Q1	Q2	Q3	Q4
1	* → Q1	* → Q1			
*					
Пробел	_ → Q1	_ ⛔ Q1			

Рисунок 4.12 – Таблиця відповідності для МТ

3.3 Задати на інформаційній стрічці послідовність, що складається з цифр 1, наприклад, 1111 (рис. 4.13).

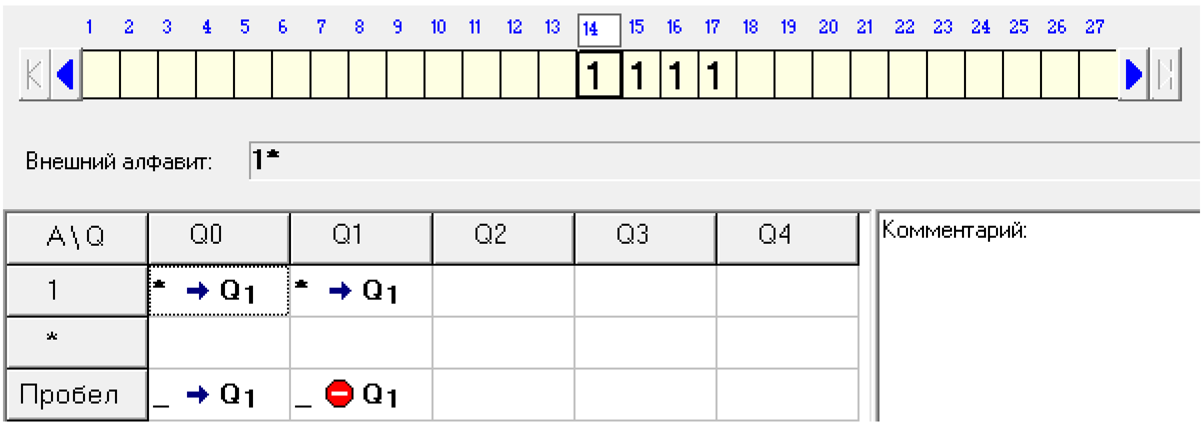


Рисунок 4.13 – Уведена послідовність одиниць

3.4 Запустити програму для виконання командою ПУСК/ЗАПУСТИТИ. Результат виконання програми наведено на рисунку 4.14.

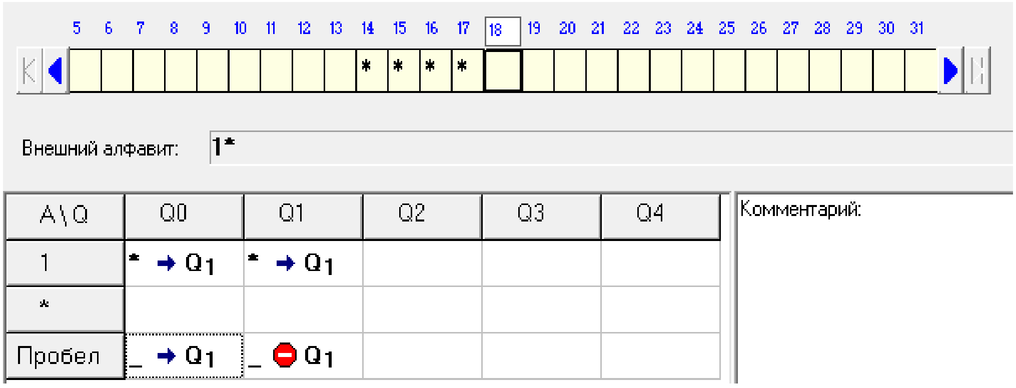


Рисунок 4.14 – Результат заміни введеної послідовності 1 на символи «*»

3.5 Увести іншу послідовність одиниць.
Проаналізувати отриманий результати.

Приклад 4

Написати програму, яка в кінцевому наборі одиниць ставить символ «*» на місця першої і останньої одиниць, враховуючи, що головка для зчитування розташована на першій лівій одиниці.

4.1 Задати зовнішній алфавіт у вигляді цифри 1 та символу «*».

4.2 Заповнити таблицю МТ командами для введеного на інформаційній стрічці набору одиниць «1111». Для цього необхідно спочатку записати команди у вигляді:

1) $1q0 \rightarrow *q1$ – знаходження в стані q_0 на початку числа та виконання заміни цифри 1 на символ «*»;

2) $1q1 \rightarrow *q1$ – знаходження в стані q_1 в кінці числа та заміна всіх цифр 1 на символ «*»;

3) $1q1 \rightarrow *\text{стоп}q2$ – знаходження останньої одиниці в стані $q2$ та заміна її на символ «*». На наступній пустій клітинці буде виконано зупинення програми.

На рисунку 4.15 наведено заповнену МТ у програмі ALGO 2000.

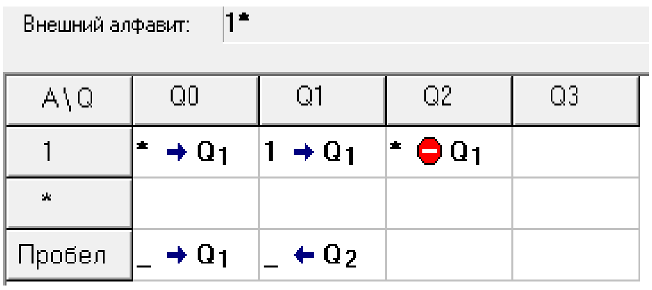


Рисунок 4.15 – Програма для МТ

Результат виконання наведено на рисунку 4.16.

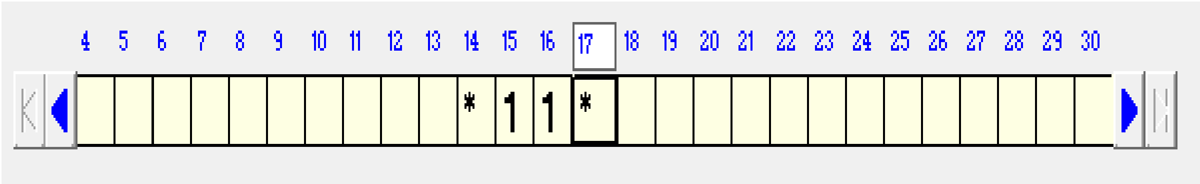


Рисунок 4.16 – Результат заміни першої та останньої одиниць на символ «*»

4.3 Увести інший набір одиниць та проаналізувати отримані результати.

Приклад 5

Написати програму, що моделює роботу МТ, яка стирає всі цифри введеного числа. Головка може бути розташована на будь-якій цифрі введеної послідовності.

5.1 Задати зовнішній алфавіт у вигляді послідовності цифр: *0 1*.

5.2 Заповнити таблицю МТ командами та інформаційну стрічку цифрами 10100, як це наведено на рисунку 4.17.

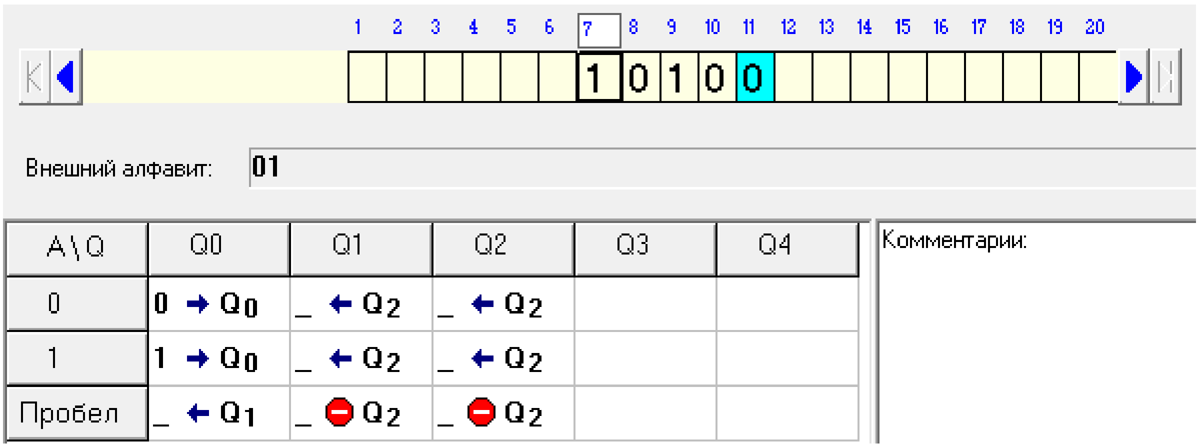


Рисунок 4.17 – Уведені дані

5.3 Запустити програму для виконання. Результати виконання наведено на рисунку 4.18.

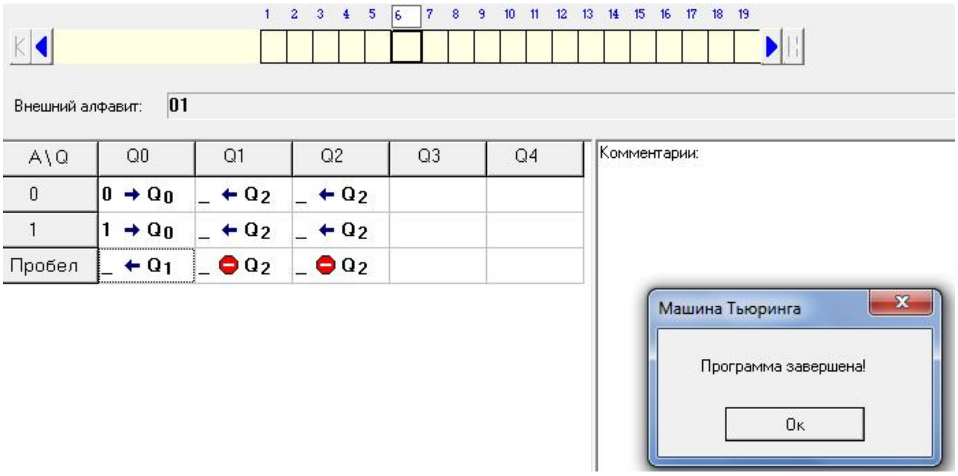


Рисунок 4.18 – Заміна введених цифр на пусті клітинки

Приклад 6

Написати програму, що моделює роботу машини Тюрінга, яка інвертує вхідний ланцюжок, записаний з використанням цифр 0 та 1. Вхідний алфавіт МТ задано множиною $A=\{0,\ 1,\ \varepsilon\}$, де символ ε відповідає пустій клітинці, а кількість станів пристрою управління задано у вигляді множини $Q = \{q_0,\ q_1,\ q_к\}$.

Якщо, наприклад, початкова конфігурація в стані q_0 має вигдяд 110011, то кінцева конфігурація після завершення операції інвертування повинна мати вигляд $q_к$ 001100.

6.1 Задати зовнішній алфавіт у вигляді послідовності цифр: *0 1*.

6.2 Заповнити таблицю МТ командами, як це наведено на рисунку 4.19.

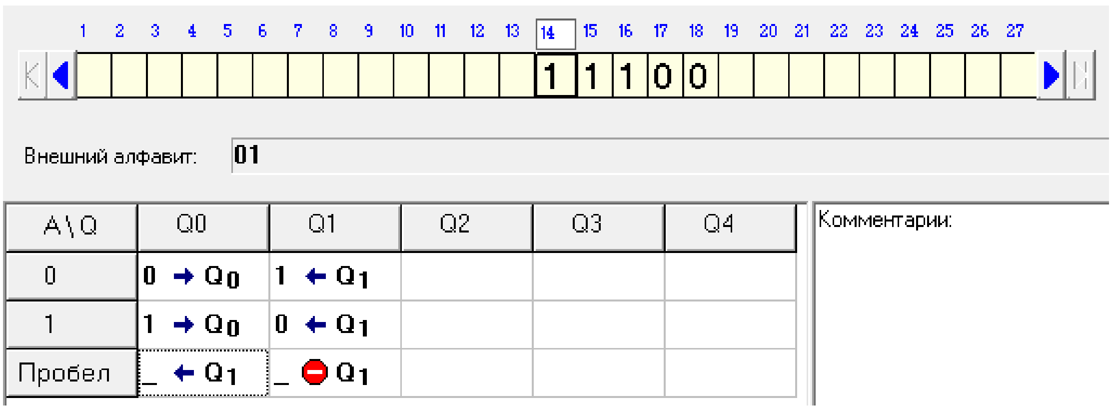


Рисунок 4.19 – Уведена для інвертування послідовність

Для розв'язання задачі машиною буде створено таку послідовність команд:

$q_0 1 \rightarrow q_0 0$*Вправо*,

$q_0 0 \rightarrow q_0 1$*Вправо*,

$q_0\, \varepsilon \rightarrow q_1\, \varepsilon$ *Вліво*,

$q_1 0 \rightarrow q_1 0$*Вліво*,

$q_1 1 \rightarrow q_1 1$*Вліво*,

$q_1 \varepsilon \rightarrow q_к\, \varepsilon R$,

де ε – пустий_символ.

З використанням же напрямку переміщення, заданого буквами англійського алфавіту, послідовність команд буде виглядати так:

$q_0 1 \rightarrow q_0 0R,$ $\qquad$ $q_1 0 \rightarrow q_1 0L,$

$q_0 0 \rightarrow q_0 1R,$ $\qquad$ $q_1 1 \rightarrow q_1 1L,$

$q_0\, \varepsilon \rightarrow q_1 \varepsilon L,$ $\qquad$ $q_1 \varepsilon \rightarrow q_к\, \varepsilon R.$

У стандартній початковій конфігурації головка стоїть над першим символом ліворуч, а пристрій управління знаходиться в початковому стані. На наступному такті машина Тюрінга, не змінюючи свого стану, замінює цифру 0 на цифру 1, або навпаки, і зсувається вправо на один символ. Після перегляду всього ланцюжка під головкою виявиться символ, що вказує на порожню клітинку.

У цьому випадку МТ переходить у новий стан і переміщується вліво на один символ. На наступних тактах керувальний пристрій не змінює свого стану, залишає без зміни символ під головкою і переміщується вліво доти, поки не натрапить на порожню клітинку, після чого МТ переходить у заключний стан і переміщується вправо на один символ, переходячи в стандартну заключну конфігурацію.

Результат виконання програми наведено на рисунку 4.20.

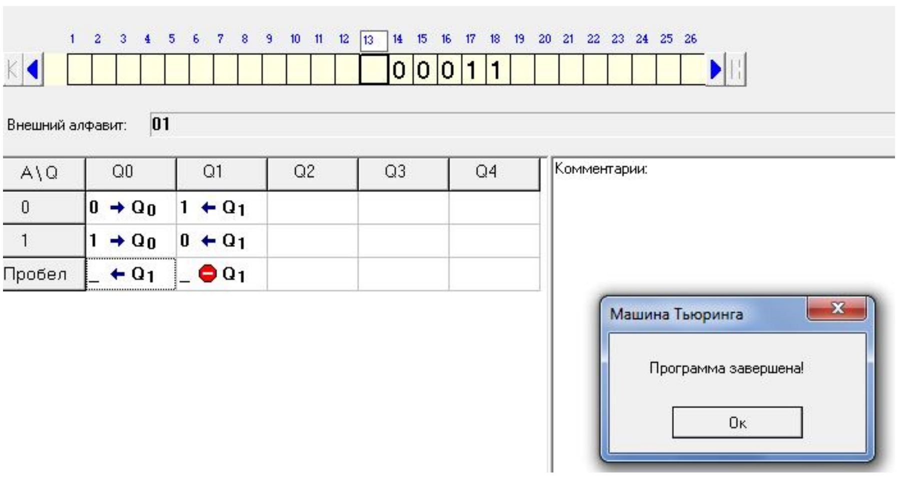


Рисунок 4.20 – Результат інвертування введеної послідовності

6.3 Увести іншу вхідну послідовність та проаналізувати отримані результати.

Приклад 7

Написати програму, що моделює роботу МТ, яка в заданій послідовності символів «*» та символів «+» залишає на інформаційній стрічці тільки символи «*» , виконуючи групування символів «*».

7.1 Задати зовнішній алфавіт у вигляді послідовності символів: * + .

7.2 Заповнити таблицю МТ командами та ввести на інформаційну стрічку послідовність «**+++**+++», як це наведено на рисунку 4.21.

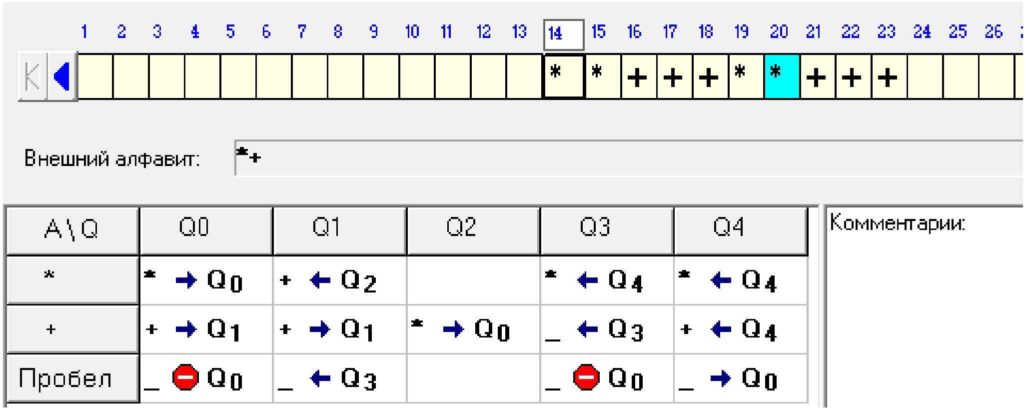


Рисунок 4.21 – Уведені дані для групування символів «*»

7.3 Запустити програму для виконання.

Отримані результати наведено на рисунку 4.22.

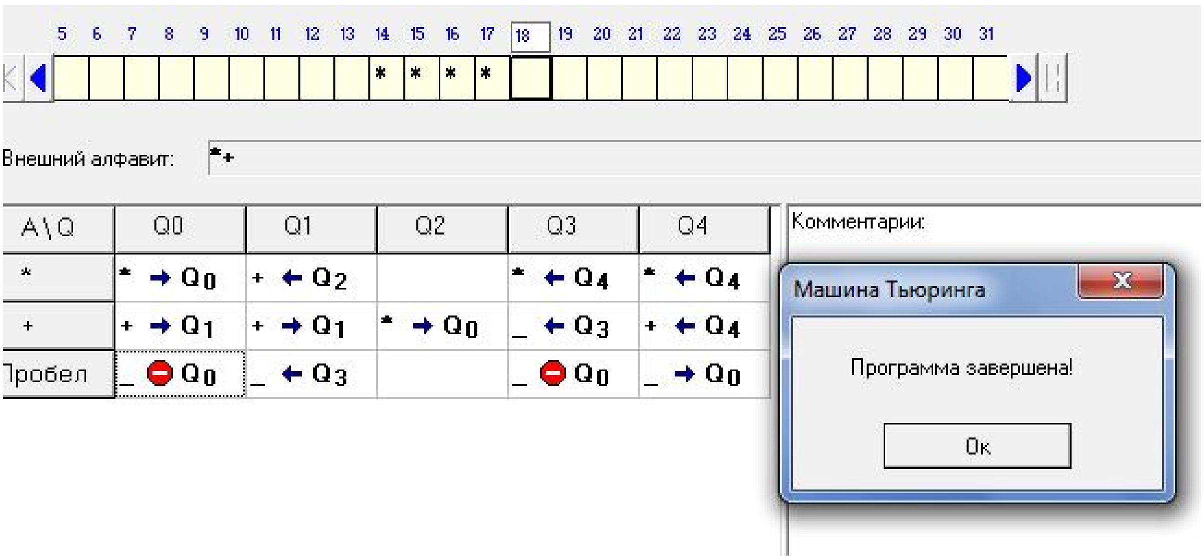


Рисунок 4.22 – Результати групування символів «*»

7.4 Увести на інформаційну стрічку послідовність «++*+*++****». Проаналізувати отримані результати.

7.5 Змінити програму так, щоб на інформаційній стрічці залишались згрупованими тільки символи «+».

Приклад 8

Розробити машину Тюрінга, яка б видаляла пари взаємних дужок, розташованих підряд «( )».

8.1 Задати зовнішній алфавіт у вигляді дужок, що відкриваються та закриваються.

8.2 Заповнити таблицю МТ командами та ввести на інформаційну стрічку послідовність «) (( ) (( )», як це наведено на рисунку 4.23.

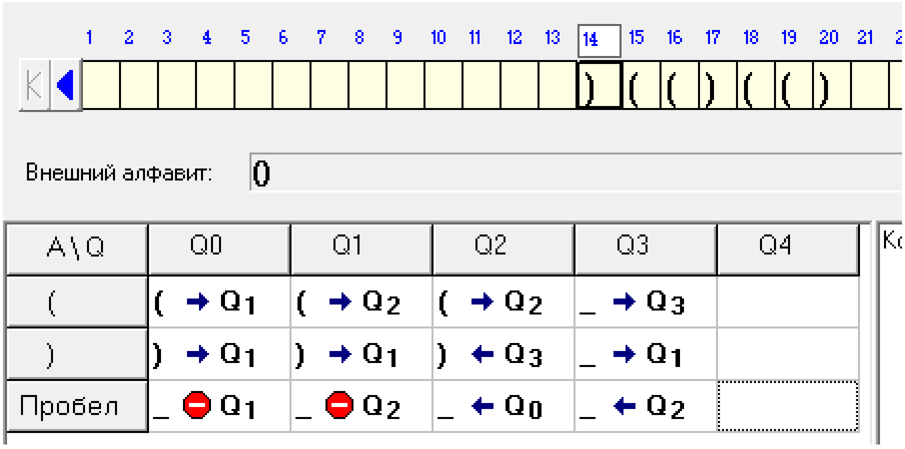


Рисунок 4.23 – Початкова конфігурація

8.3 Виконати задану програму. Автомат, що перебуває в стані q_1, оглядає крайній лівий символ рядка.

Результат роботи програми наведено на рисунку 4.24.

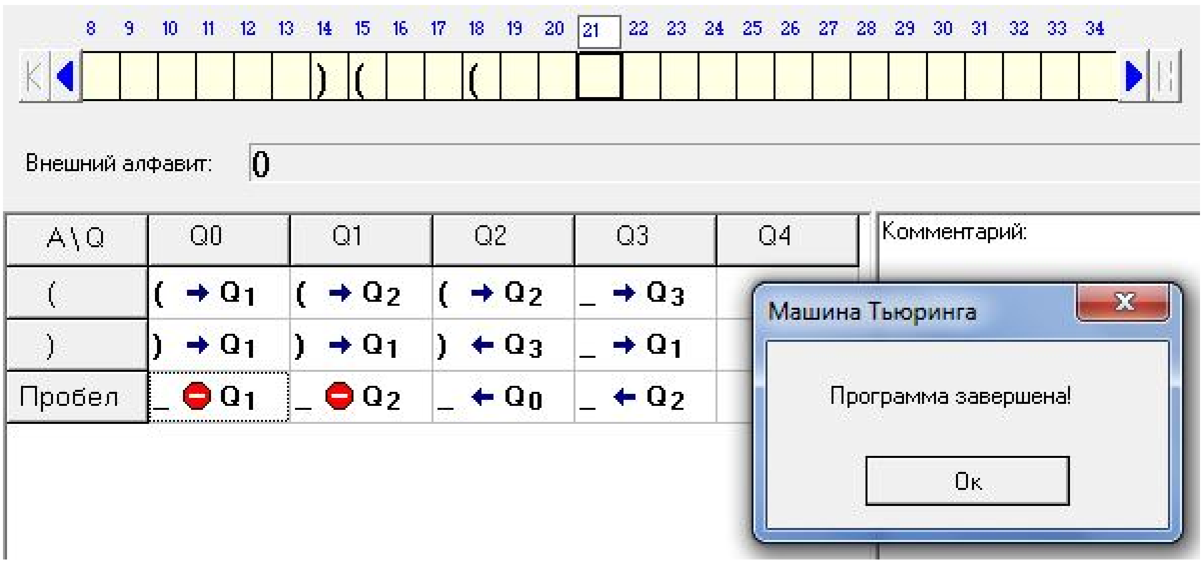


Рисунок 4.24 – Отримана послідовність у результаті вводу послідовності дужок «) (( ) (( )»

8.4 Увести іншу послідовність дужок, наприклад «)) ( ) ( ))» [11–13].

Результат роботи програми наведено на рисунку 4.25.

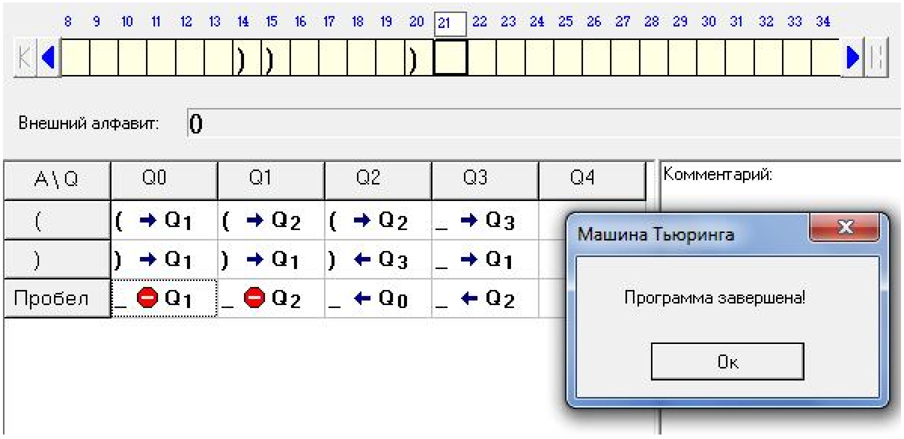


Рисунок 4.25 – Отримана послідовність у результаті вводу послідовності дужок «)) ( ) ( ))»

8.5 Удосконалити роботу програми.

Контрольні питання

1. Дати визначення МТ.
2. За якими ознаками можна класифікувати машини Тюрінга?
3. Якими способами можна представляти МТ?
4. Із яких частин складається МТ?
5. Перелічити та дати пояснення команд МТ.
6. Пояснити порядок роботи МТ.
7. Пояснити, що собою являє таблиця відповідності МТ.
8. Як буде виконуватися програма, наведена на рисунку 4.3, якщо встановити головку на порожню ліву клітинку?
9. Чому для таблиці відповідності МТ, наведеної на рисунку 4.10, не можна вводити на інформаційну стрічку символ «–»?
10. Як буде виконуватися програма (рис. 4.13), якщо керувальна головка буде розташована на лівій порожній клітинці від першої цифри 1?
11. Як буде виконуватися програма (рис. 4.13), якщо керувальна головка буде розташована не на клітинці з номером 13 (тобто лівіше за клітинку, з якої починається запис послідовності одиниць), а на клітинці з номером 12?
12. Як буде виконуватися програма (рис. 4.15), якщо керувальна головка для зчитування буде розташована на пустій правій клітинці від останньої одиниці?
13. Як буде виконуватися програма (рис. 4.15), якщо керувальна головка буде розташована не на клітинці, де введена перша 1, а на клітинці, що розташована лівіше (тобто на пустій клітинці)?
14. Як буде виконуватися програма (рис. 4.19), якщо керувальна головка для зчитування буде розташована на пустій лівій клітинці від першої лівої цифри?
15. Як буде виконуватися програма (рис. 4.21), якщо розташувати головку на пустій клітинці, лівіше першого введеного символу?

4.2 Практичне заняття 2
Моделювання роботи машини Тюрінга
в програмному інтерпретаторі ALGO 2000

Мета завдання – розробка алгоритмів функціонування МТ для виконання простих функцій за допомогою програмного інтерпретатора ALGO 2000.

Завдання на виконання практичного завдання

1. Розробити МТ, яка б виконувала операцію (x div 2) і мала вхідний алфавіт $A = \{0, 1, \varepsilon\}$, де символ ε – це пустий символ (пробіл). Операція (x div 2) реалізується зсувом ланцюжка вправо на один розряд.
2. Розробити МТ, яка б виконувала операцію конкатенації двох ланцюжків, заданих у вхідному алфавіті $A = \{0, 1, *, \varepsilon\}$.
3. Розробити МТ, яка б виконувала операцію копіювання вхідного ланцюжка, заданого в алфавіті $A = \{1, *, \varepsilon\}$, де символ «*» використовується як розділювач двох ланцюжків.
4. Розробити МТ, яка б виконувала множення числа в десятковій системі числення на 11.
5. Розробити МТ, яка б знаходила додаток числа в десятковій системі числення та числа в трійковій системі числення, наприклад, 576 + 100. Додаток потрібно подати в десятковій системі числення. Каретка розташована на крайній правій цифрі правого числа.
6. Розробити МТ, яка б могла виступати як двійково-вісімковий дешифратор.
7. Розробити МТ, яка б установлювала співвідношення між двома натуральними числами m та n, які представлено в унарній системі числення. Між числами m та n стоїть знак «?», який замінюють залежно від співвідношення між числами на один із відповідних знаків « >», « <», « =».

Загальні положення

Говорять, що МТ обчислює функцію $f(x_1, x_2, \ldots, x_n)$, якщо виконуються такі умови:

1) для будь-яких $x_1, x_2, \ldots, x_n$, які належать до області визначення функції, МТ з початкової конфігурації, маючи на стрічці подання аргументів, переходить у заключну конфігурацію, маючи на стрічці результат (подання функції);

2) для будь-яких $x_1, x_2, \ldots, x_n$, які не належать до області визначення функції, МТ, починаючи з початкової конфігурації, працює нескінченно.

Якщо початкова та заключна конфігурації МТ є стандартними, то говорять, що МТ правильно обчислює функцію.

Функція називається обчислюваною за Тюрінгом, якщо існує машина Тюрінга, що обчислює її.

Для доведення обчислювальності функції, а надалі й існування алгоритму, необхідно побудувати МТ, реалізація якої на практиці найчастіше становить трудомістке завдання. У зв’язку з цим виникає необхідність розбиття алгоритму на окремі задачі, кожна з яких буде розв’язуватись окремою машиною Тюрінга. Якщо об’єднати програми цих машин, то вийде нова програма, яка буде дозволяти вирішувати вихідну задачу.

Машини Тюрінга можуть обчислювати шукану функцію з відновленням і без відновлення. Обчислення функції з відновленням означає, що машини Тюрінга працюють із збереженням початкових даних: спочатку отримати результат на стрічці, а потім початкові дані. Обчислення функції без відновлення означає, що машини Тюрінга працюють без збереження початкових даних.

Справедливим є твердження про те, що будь-яка правильно обчислювана функція є правильно обчислюваною, якщо вона обчислюється з відновленням.

Обчислювана функція використовується для уточнення поняття алгоритму.

Обчислювальна функція – це будь-яка числова функція (зазвичай від цілочислових аргументів та з цілочисловою областю значень), значення якої може бути обраховане за допомогою деякого алгоритму.

Математично точно описана множина числових функцій, які співпадають з сукупністю всіх обчислювальних функцій з широким розумінням алгоритму, отримала назву рекурсивних функцій.

Тезис Черча формулюється так: клас усіх рекурсивних функцій співпадає з класом скрізь визначених обчислювальних функцій [14–15].

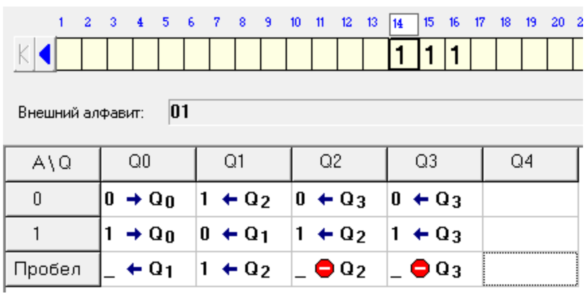
Порядок виконання практичного заняття

Приклад 1

Побудувати машину Тюрінга, яка правильно обчислює функцію $f(x) = x+1$ за правилами двійкового додавання.

1.1 Обрати вхідний алфавіт $A = \{0, 1, \varepsilon\}$, де ε – пустий символ (пробіл).

1.2 Представити МТ у вигляді таблиці відповідності (рис. 4.26).



A\Q	Q0	Q1	Q2	Q3	Q4
0	0 → Q_0	1 ← Q_2	0 ← Q_3	0 ← Q_3	
1	1 → Q_0	0 ← Q_1	1 ← Q_2	1 ← Q_3	
Пробел	_ ← Q_1	1 ← Q_2	_ Q_2	_ Q_3	

Рисунок 4.26 – Уведені дані

1.3 Записати програму побудованої МТ для випадку, коли вхідний ланцюжок на стрічці дорівнює двійковому числу 111.

Зліва від кожної команди наведено представлення вхідного ланцюжка на стрічці до виконання цієї команди. Символ, який знаходиться під головкою, будемо позначати підкресленням.

Оскільки на стрічці написано число 111, то команди з цифрою 0 не використовуються.

1) $q_0 1 \rightarrow q_0 1R$ $\varepsilon\underline{1}11\varepsilon$ (у початковому стані q_0 головка знаходиться на першій клітинці з цифрою 1 та згідно з командою не змінює вмісту клітинки, а тільки переміщується вправо);

2) $q_0 1 \rightarrow q_0 1R$ $\varepsilon 1\underline{1}1\varepsilon$ (стан q_0 не змінюється і головка далі переміщується вправо на наступну клітинку з цифрою 1);

3) $q_0 1 \rightarrow q_0 1R$ $\varepsilon 11\underline{1}\varepsilon$ (стан q_0 не змінюється і головка далі переміщується вправо на наступну клітинку з цифрою 1);

4) $q_0\varepsilon \rightarrow q_1\varepsilon L$ $\varepsilon 111\underline{\varepsilon}$ (стан q_0 змінюється, всі одиниці числа на інформаційній стрічці закінчилися, а отже головка зустрічає символ ε (пуста клітинка), переходить в стан q_1, згідно з яким головка зміщується вліво, разом із тим у клітинці зліва повинна виконатися заміна установленої цифри 1 на цифру 0, що демонструють пункти 5 і 6);

5) $q_1 1 \rightarrow q_1 0L$ $\varepsilon 11\underline{1}\varepsilon$;

6) $q_1 1 \rightarrow q_1 0L$ $\varepsilon 11\underline{0}\varepsilon$;

7) $q_1 1 \rightarrow q_1 0L$ $\varepsilon 1\underline{0}0\varepsilon$ (у подальшому виконується така ж сама команда і залишається такий самий стан: стан q_1, заміна цифри 1 на цифру 0 і переміщення головки вліво);

8) $q_1 1 \rightarrow q_1 0L$ $\varepsilon\underline{0}00\varepsilon$ (в подальшому виконується така ж сама команда і залишається такий самий стан: стан q_1, заміна цифри 1 на цифру 0 і переміщення головки вліво);

9) $q_1\varepsilon \rightarrow q_2 1L$ $\underline{\varepsilon}000\varepsilon$ (стан q_1 змінюється, виконана заміна всіх трьох цифр 1 на цифру 0, а отже, головка зустрічає символ ε, переходить у стан q_2, згідно з яким головка переміщується вліво та замінює символ ε на цифру 1);

10) $q_2\,\varepsilon \rightarrow q_к\,\varepsilon R$ $\underline{\varepsilon}1000\varepsilon$.

Із прикладу видно, що МТ із стандартної початкової конфігурації, маючи на стрічці число 111, після виконання сукупності команд (п.п. 1–10) перейшла в стандартний заключний стан, маючи на стрічці результат 1000. Дійсно $111_2 + 1_2 = 1000_2$.

Результат роботи МТ наведено на рисунку 4.27.

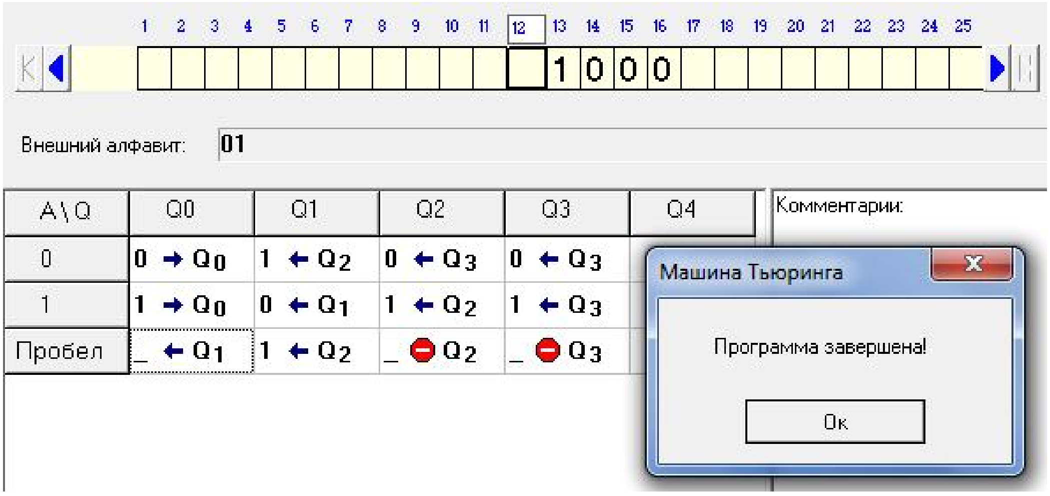
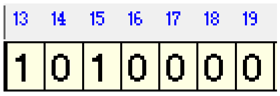


Рисунок 4.27 – Результат роботи функції *f(x)* = *x*+*1*

1.4 Увести двійкове число 1001111 на стрічку.

Отриманий результат є наступний: 

Приклад 2

Написати програму, яка б розв'язувала задачу «подільності на 4».

У процесі стандартного кодування ціле число *n* представляється словом з нулей та одиниць, тобто двійковим записом цього числа. Додатне ціле число ділиться на 4 тоді і тільки тоді, коли дві останні цифри двійкового запису даного числа є нулями.

2.1 Обрати вхідний алфавіт *A* ={0, 1}.

2.2 Представити МТ у вигляді таблиці відповідності (рис. 4.28).

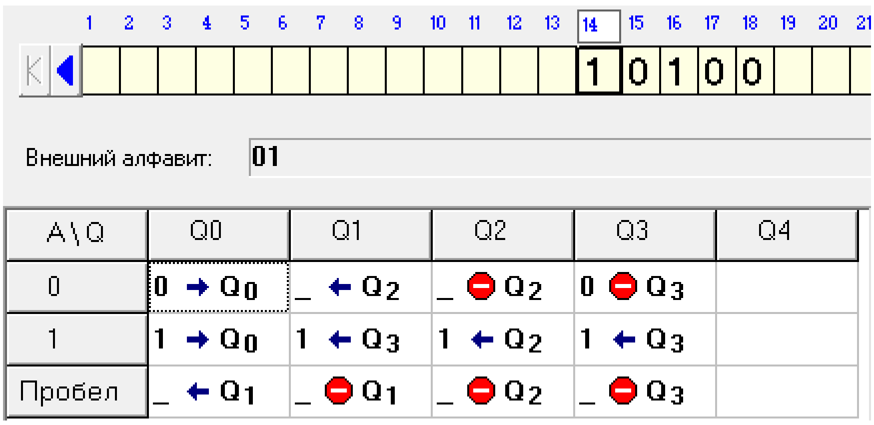


Рисунок 4.28 – МТ для розв'язання задачі «подільності на 4»

Розроблена програма обчислює функцію $f_M : \{0,1\}^* \to \{0,1,b\}^*$, яка відображає кожне слово $x \in \{0,1\}^*$ у слово $f_M(x)$, яке отримується з x знищенням двох крайніх справа символів. Якщо $|x| < 2$, то програма як $f_M(x)$ видає пусте слово.

На рисунку 4.29 наведено демонстрацію роботи алгоритму програми для розв'язання задачі «подільності на 4» числа x=10100, яке в десятковій системі числення є числом 20.

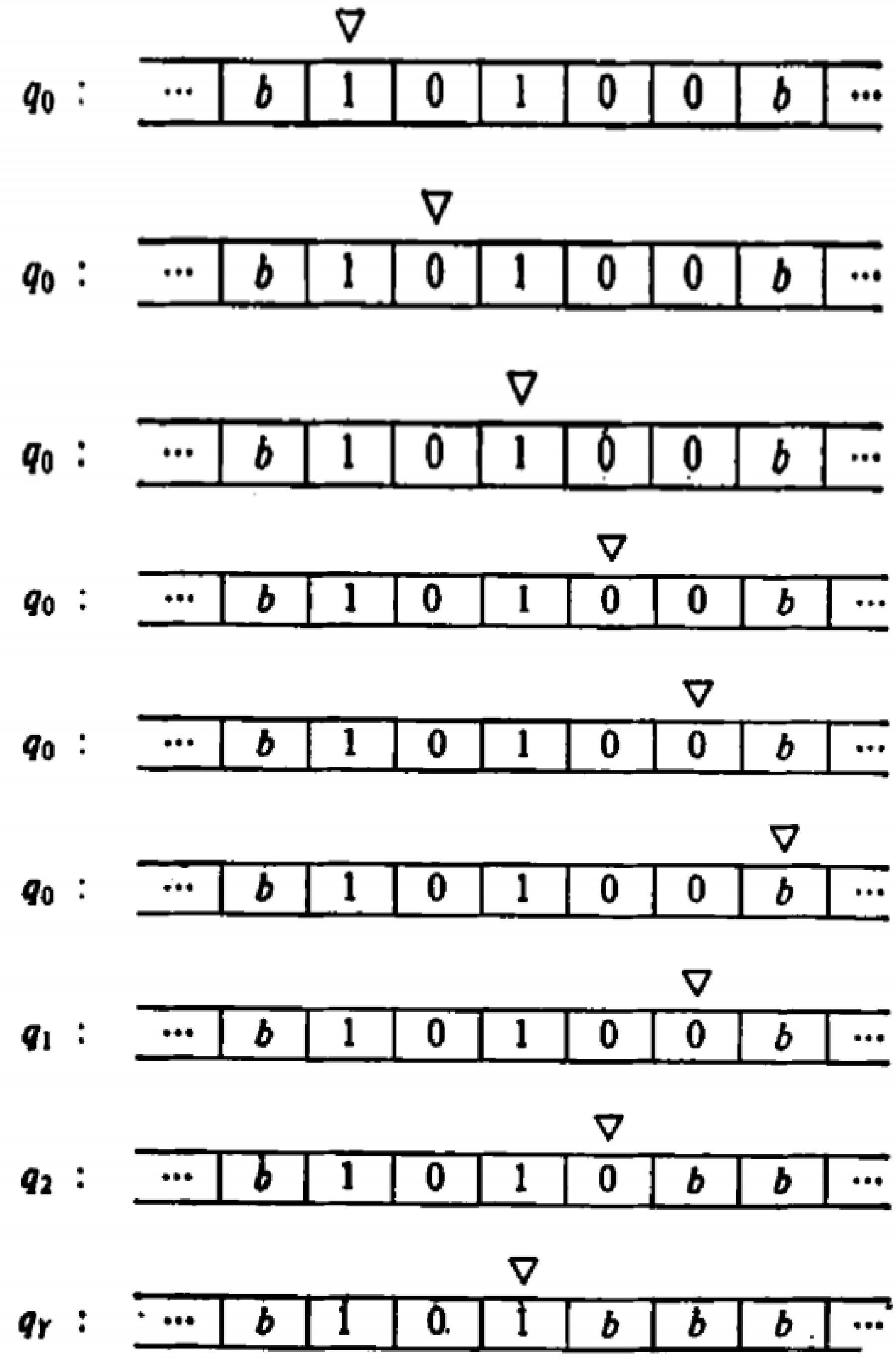


Рисунок 4.29 – Візуалізація роботи наведеної програми

2.3 Запустити програму для виконання. Отриманий результат наведено на рисунку 4.30.

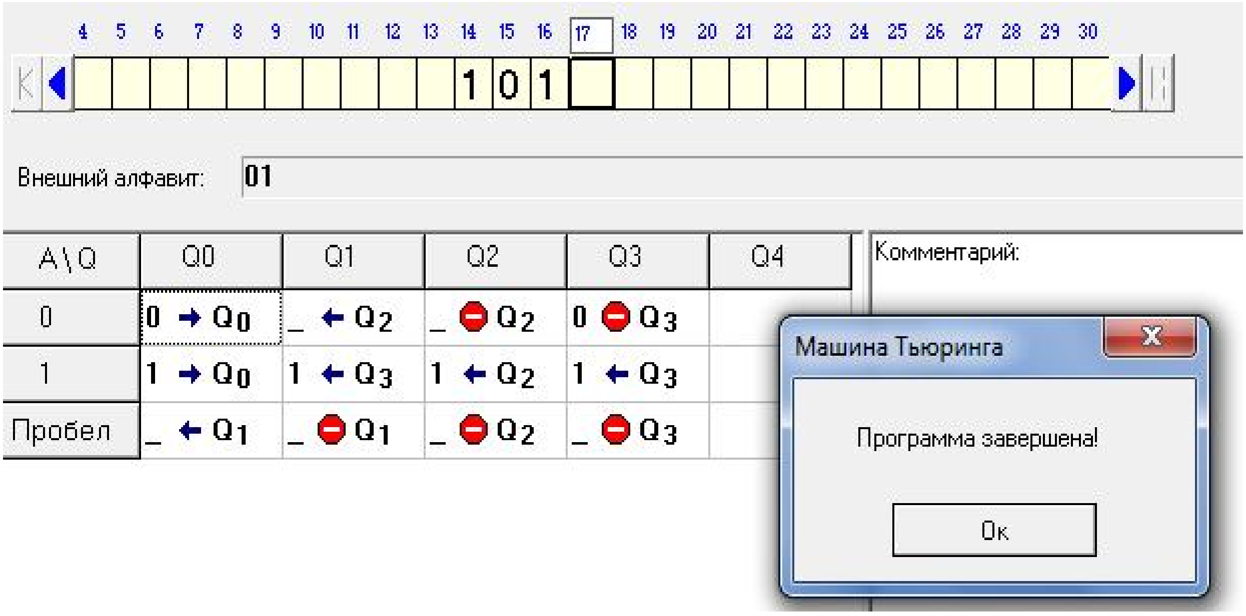


Рисунок 4.30 – Результат роботи програми «подільності на 4»

У десятковій системі числення число 101 відповідає числу 5.

Наведена програма видалила з уведеного числа дві останні цифри – нулі, тим самим підтверджуючи подільність введеного числа на 4.

У випадку, якщо двома останніми цифрами уведеного числа є не нулі, програма не вносить жодних змін, а це означає, що введене число є неподільним на 4.

2.4 Запустити програму для виконання, якщо введеное число є 1010000, що відповідає числу 80 в десятковій системі числення. Перевірити правильність роботи програми.

2.5 Запустити програму для виконання, якщо введене число є 1010011, що відповідає числу 83 в десятковій системі числення. Перевірити правильність роботи програми.

Приклад 3

Розробити машину Тюрінга, яка б збільшувала на 1 задане в вісімковій системі числення число *n*.

Вісімкова система числення – це позиційна цілочисельна система числення з основою 8, для представлення чисел в якій використовуються цифри від 0 до 7.

Вісімкова система найчастіше використовується в областях, пов’язаних із цифровими пристроями, та характеризується легким переведенням вісімкових чисел у двійкові і назад шляхом заміни вісімкових чисел на триплети двійкових. Ця система числення раніше широко використовувалася в програмуванні і взагалі в комп’ютерній документації, проте наразі майже повністю витіснена шістнадцятирічною системою.

3.1 Задати зовнішній алфавіт у вигляді послідовності цифр: *0 1 2 3 4 5 6 7*.

3.2 Заповнити таблицю МТ командами, як це наведено на рисунку 4.31.

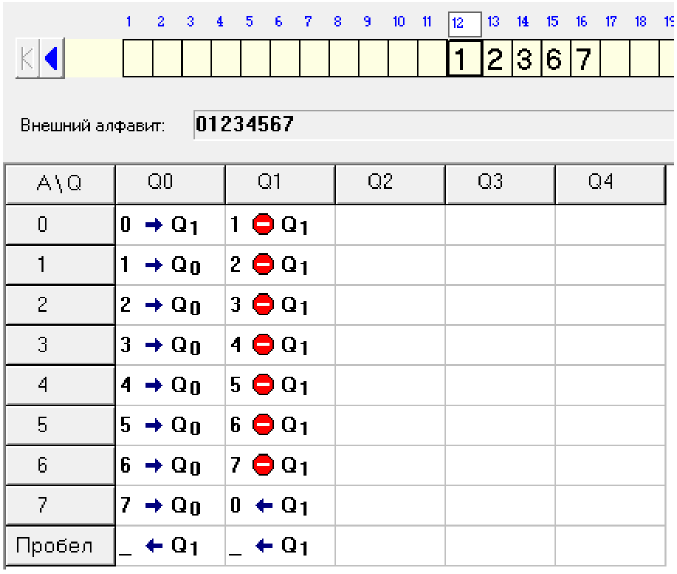


Рисунок 4.31 – Обчислення в вісімковій системі

Автомат у стані q_0 оглядає будь-яку цифру вхідного слова відповідно до команд першого стовпчика.

Команди другого стовпчика додають одиницю до потрібної цифри (тієї, що введена на інформаційну стрічку) та виконують зупинення програми.

Результат роботи програми наведено на рисунку 4.32.

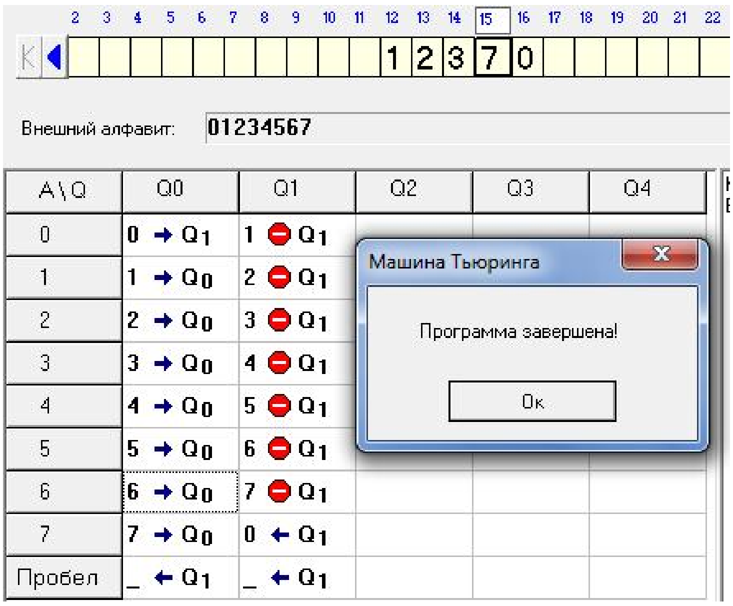


Рисунок 4.32 – Результат обчислення в вісімковій системі

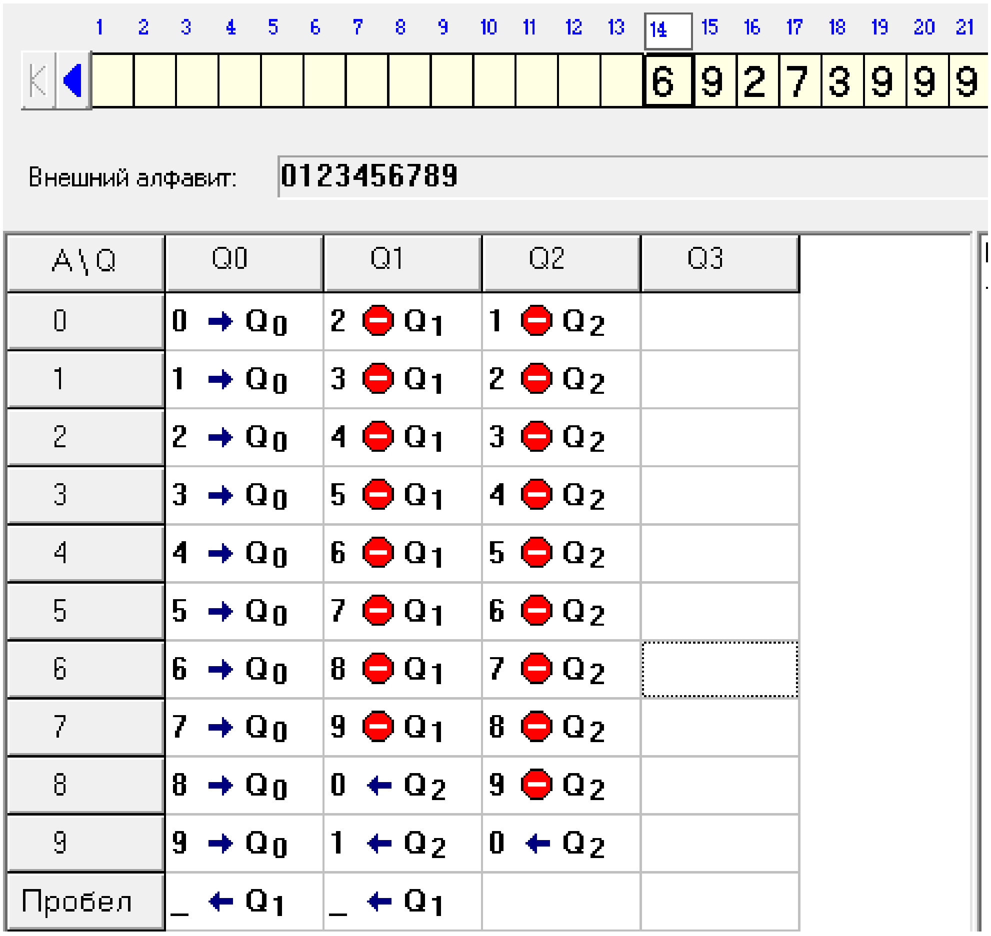
Розроблена програма збільшує розрядність введеного числа відповідно до правил математики.

Приклад 4

Написати програму, яка б моделювала роботу МТ, що збільшує задане десяткове число на 2. Разом із тим головка має бути розташована на першій цифрі числа.

4.1 Задати зовнішній алфавіт у вигляді послідовності цифр: *0 1 2 3 4 5 6 7 8 9 .*

4.2 Заповнити таблицю МТ командами, як це наведено на рисунку 4.33.



A\Q	Q0	Q1	Q2	Q3
0	0 → Q_0	2 ⛔ Q_1	1 ⛔ Q_2	
1	1 → Q_0	3 ⛔ Q_1	2 ⛔ Q_2	
2	2 → Q_0	4 ⛔ Q_1	3 ⛔ Q_2	
3	3 → Q_0	5 ⛔ Q_1	4 ⛔ Q_2	
4	4 → Q_0	6 ⛔ Q_1	5 ⛔ Q_2	
5	5 → Q_0	7 ⛔ Q_1	6 ⛔ Q_2	
6	6 → Q_0	8 ⛔ Q_1	7 ⛔ Q_2	
7	7 → Q_0	9 ⛔ Q_1	8 ⛔ Q_2	
8	8 → Q_0	0 ← Q_2	9 ⛔ Q_2	
9	9 → Q_0	1 ← Q_2	0 ← Q_2	
Пробел	_ ← Q_1	_ ← Q_1		

Рисунок 4.33 – Робота з десятковим числом

Команди першого стовпчика призначено для перегляду окремо заданої на інформаційній стрічці цифри та переміщення головки вправо до наступної цифри. За досягнення кінця послідовності та переходу до першої пустої клітинки виконується переміщення вліво відповідно до послідовності за допомогою команди $\_ \leftarrow q_1$.

Команди другого стовпчика виконують додавання цифри 2 до кожної з проглянутих цифр, а додавання цифри 2 до цифри 8 приведе до заповнення клітинки 0, а додавання цифри 2 до цифри 9 – до заповнення клітинки 1.

Результат роботи програми наведено на рисунку 4.34.

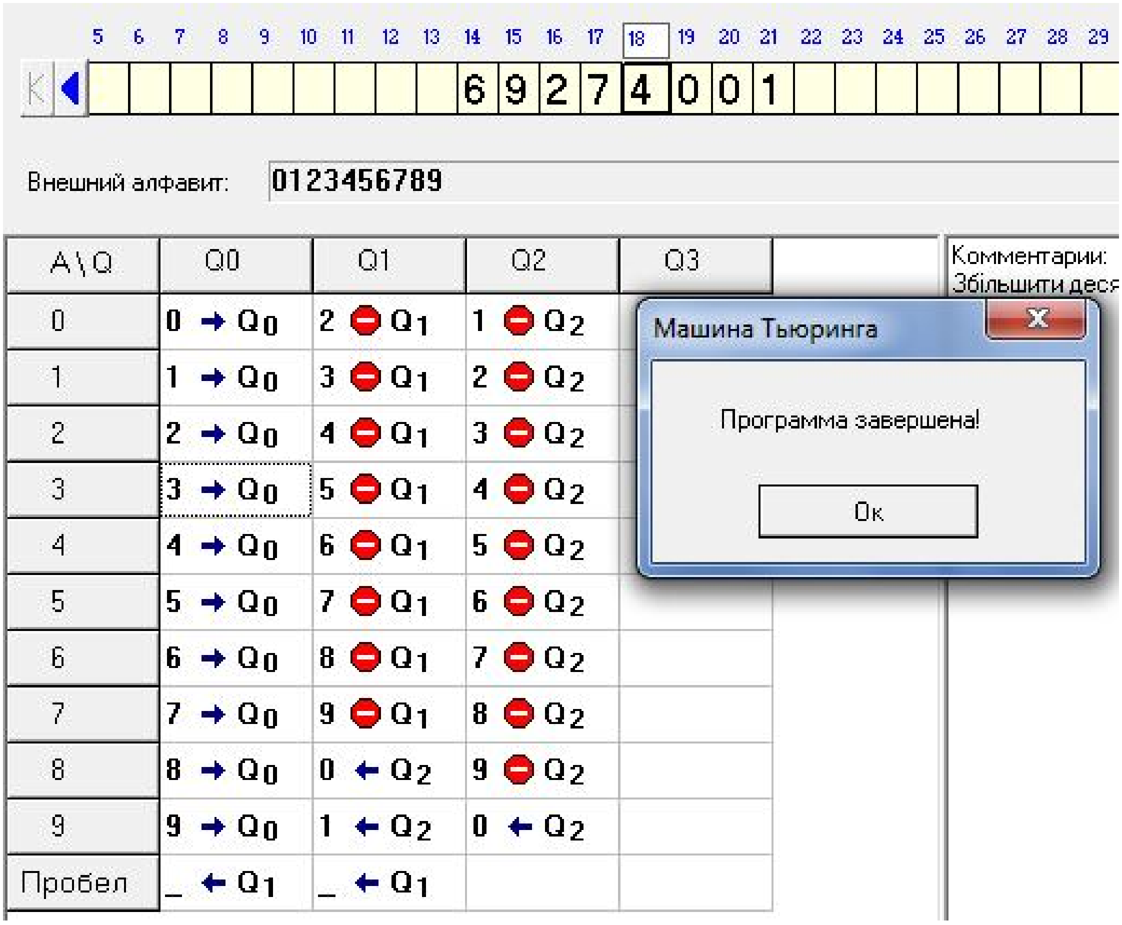


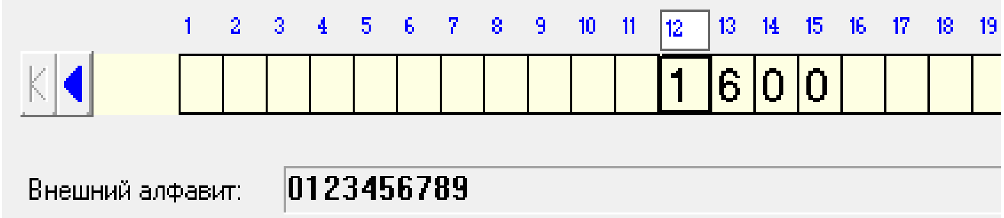
Рисунок 4.34 – Результат роботи програми

Приклад 5

Розробити МТ, яка б змогла записане в десятковій системі числення число множити на 2. При цьому головка розташована над лівою цифрою, тобто автомат у стані q_0 оглядає крайню ліву цифру числа (але головка може знаходитися й над крайньою правою клітинкою)

5.1 Задати зовнішній алфавіт у вигляді послідовності цифр: *0 1 2 3 4 5 6 7 8 9.*

5.2 Заповнити таблицю МТ командами, як це наведено на рисунку 4.35.



A\Q	Q0	Q1	Q2	Q3	Q4
0	0 → Q_0	0 ← Q_1	1 ← Q_1		
1	1 → Q_0	2 ← Q_1	3 ← Q_1		
2	2 → Q_0	4 ← Q_1	5 ← Q_1		
3	3 → Q_0	6 ← Q_1	7 ← Q_1		
4	4 → Q_0	8 ← Q_1	9 ← Q_1		
5	5 → Q_0	0 ← Q_2	1 ← Q_2		
6	6 → Q_0	2 ← Q_2	3 ← Q_2		
7	7 → Q_0	4 ← Q_2	5 ← Q_2		
8	8 → Q_0	6 ← Q_2	7 ← Q_2		
9	9 → Q_0	8 ← Q_2	9 ← Q_2		
Пробел	_ ← Q_1	_ ⛔ Q_1	1 ← Q_3	_ ⛔ Q_3	

Рисунок 4.35 – Програма множення на 2 числа, записаного в десятковій системі числення

У цій програмі:

– стан q_0 слугує для виконання команд пошуку правої (молодшої) цифри числа;

– стан q_1 слугує для виконання команд множення чергової цифри числа на 2 без додавання одного переносу;

– стан q_2 слугує для виконання команд множення чергової цифри числа на 2 з додаванням одного переносу.

Результат роботи програми наведено на рисунку 4.36.

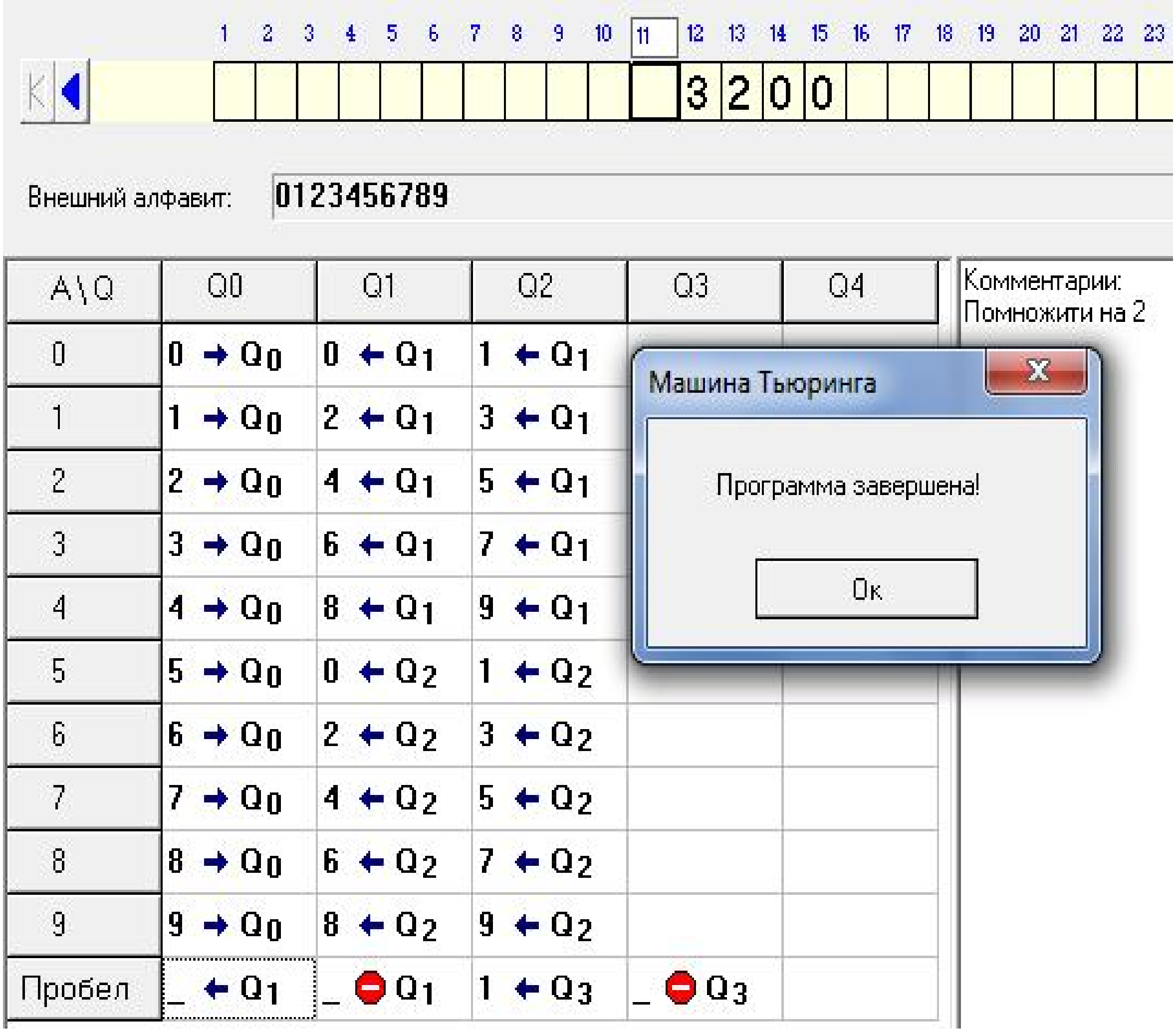


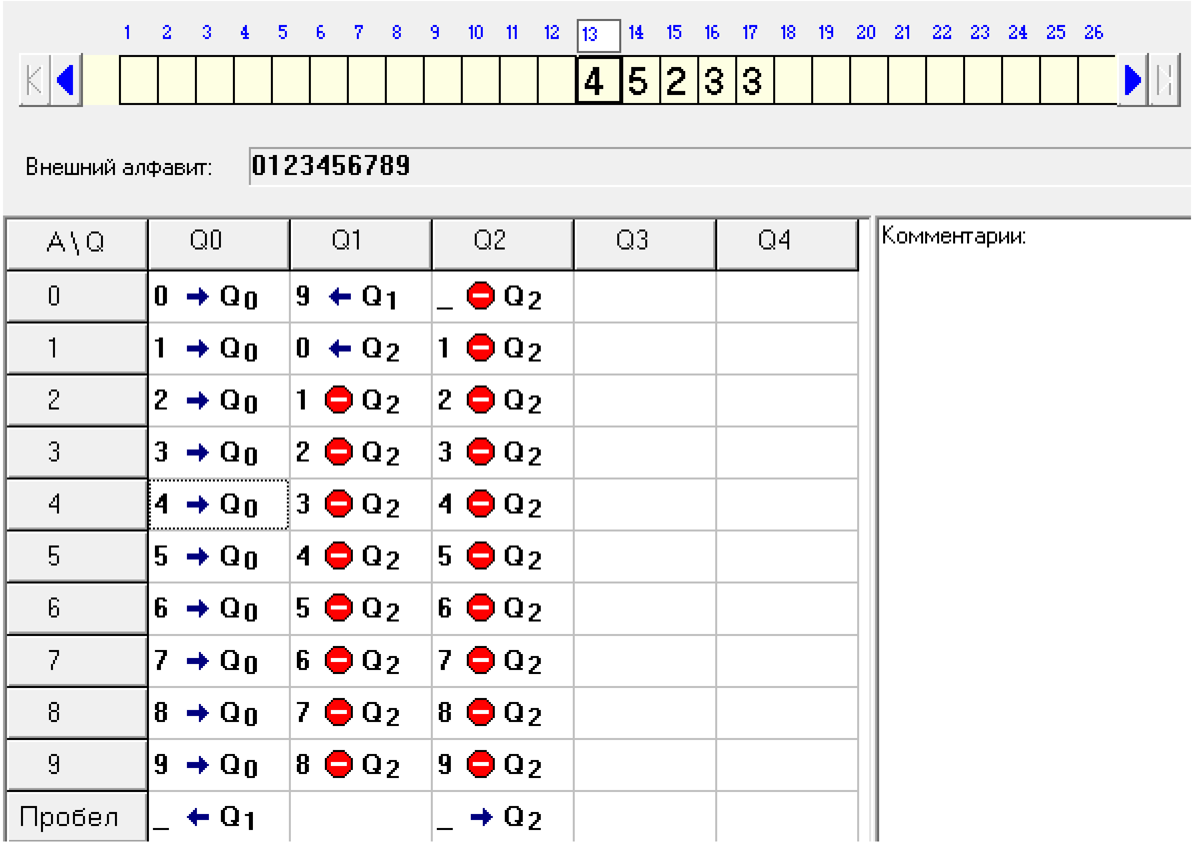
Рисунок 4.36 – Результат виконання програми множення на 2 числа, записаного в десятковій системі числення

Приклад 6

Розробити МТ, яка б зменшувала на одиницю задане натуральне число *n*, що є більшим за *1*. При цьому в вихідному слові старша цифра не повинна дорівнювати 0. Головка знаходиться над лівою цифрою.

6.1 Задати зовнішній алфавіт у вигляді такої послідовності цифр: *0 1 2 3 4 5 6 7 8 9*.

6.2 Заповнити таблицю МТ командами, як це наведено на рисунку 4.37.



A\Q	Q0	Q1	Q2	Q3	Q4
0	0 → Q0	9 ← Q1	_ ⛔ Q2		
1	1 → Q0	0 ← Q2	1 ⛔ Q2		
2	2 → Q0	1 ⛔ Q2	2 ⛔ Q2		
3	3 → Q0	2 ⛔ Q2	3 ⛔ Q2		
4	4 → Q0	3 ⛔ Q2	4 ⛔ Q2		
5	5 → Q0	4 ⛔ Q2	5 ⛔ Q2		
6	6 → Q0	5 ⛔ Q2	6 ⛔ Q2		
7	7 → Q0	6 ⛔ Q2	7 ⛔ Q2		
8	8 → Q0	7 ⛔ Q2	8 ⛔ Q2		
9	9 → Q0	8 ⛔ Q2	9 ⛔ Q2		
Пробел	_ ← Q1		_ → Q2		

Рисунок 4.37 – Робота з натуральними числами

Стан q_1 надає можливість зменшувати молодшу цифру на 1.

Якщо ця цифра є більшою за 1, то після її зменшення відразу буде виконано зупинення програми.

Якщо молодша цифра дорівнює 0, то замість неї буде записано цифру 9, виконано переміщення вліво і знову виконано дію віднімання.

A\Q	Q0	Q1
0	0 → Q0	9 ← Q1

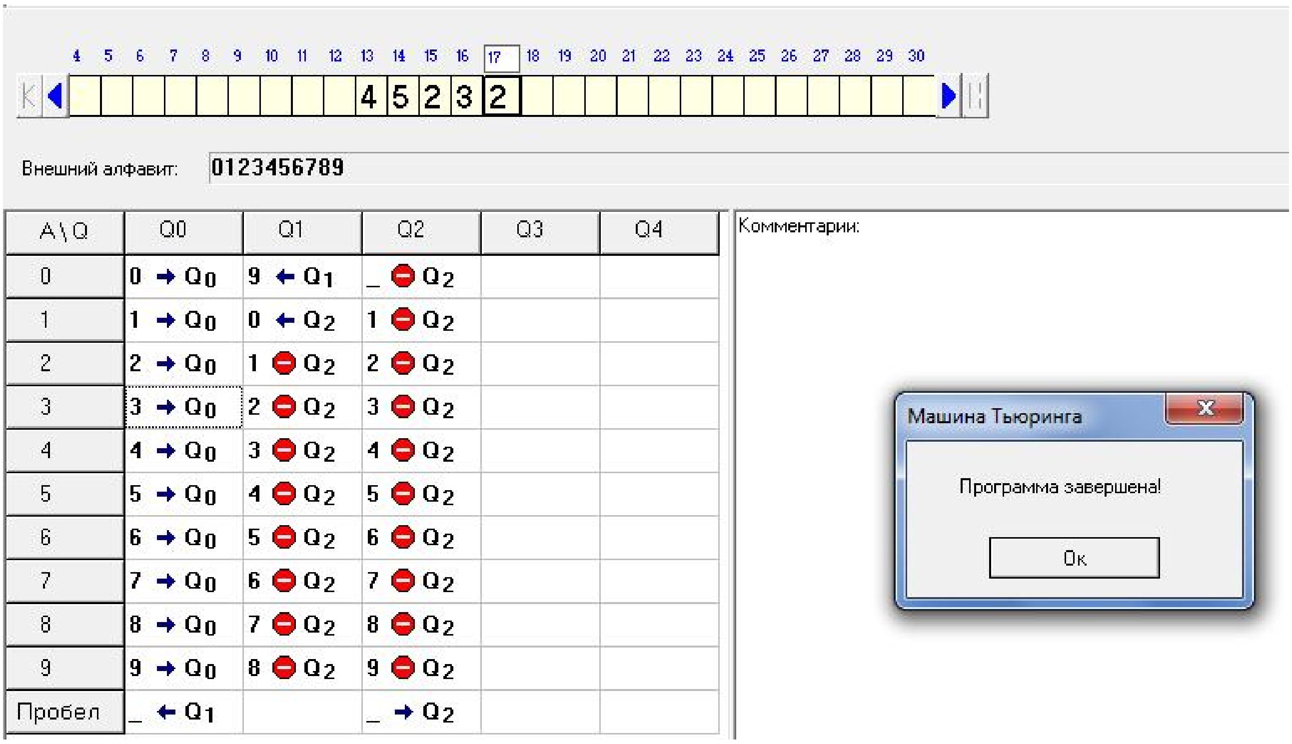
.

Якщо цифра, що зменшується на одиницю, сама дорівнює 1, то замість неї (тобто одиниці) буде записано цифру 0 і виконано перехід в стан q_2.

A\Q	Q0	Q1
0	0 → Q0	9 ← Q1
1	1 → Q0	0 ← Q2

.

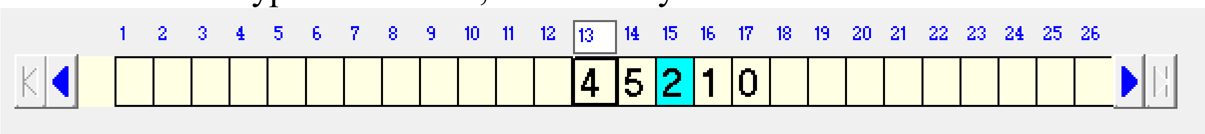
Результат роботи програми наведено на рисунку 4.38.



A\Q	Q0	Q1	Q2	Q3	Q4
0	0 → Q0	9 ← Q1	_ ⛔ Q2		
1	1 → Q0	0 ← Q2	1 ⛔ Q2		
2	2 → Q0	1 ⛔ Q2	2 ⛔ Q2		
3	3 → Q0	2 ⛔ Q2	3 ⛔ Q2		
4	4 → Q0	3 ⛔ Q2	4 ⛔ Q2		
5	5 → Q0	4 ⛔ Q2	5 ⛔ Q2		
6	6 → Q0	5 ⛔ Q2	6 ⛔ Q2		
7	7 → Q0	6 ⛔ Q2	7 ⛔ Q2		
8	8 → Q0	7 ⛔ Q2	8 ⛔ Q2		
9	9 → Q0	8 ⛔ Q2	9 ⛔ Q2		
Пробел	_ ← Q1		_ → Q2		

Рисунок 4.38 – Результат зменшення натурального числа на одиницю

6.3 Увести натуральне число, яке закінчується на 10.



Якщо число закінчується на 10, то згідно з програмою:

– розглядаються всі цифри числа з переміщенням вправо до першої пустої клітинки, далі виконується переміщення вліво з переходом у стан q_1:

Пробел	_ ← Q1

;

– розглядається остання цифра числа, яка в цьому випадку дорівнює цифрі 0. Згідно з цією програмою замість цифри 0 записується цифра 9, виконується перехід вліво з подальшою обробкою наступної цифри і програма залишається в стані q_1:

A\Q	Q0	Q1
0	0 → Q0	9 ← Q1

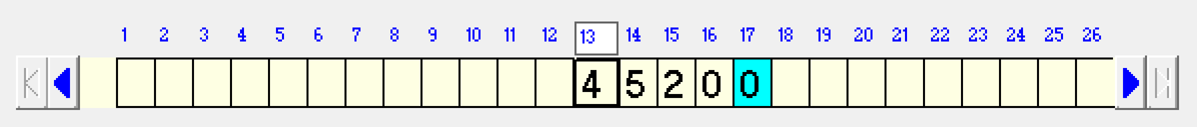
;

– розглядається передостання цифра числа, яка в цьому випадку дорівнює цифрі 1. Згідно з цією програмою замість цифри 1 записується цифра 0, виконується перехід вліво з подальшою обробкою наступної цифри (цифри 2) і програма переходить у стан q_2;

– перемістившись до цифри 2 у стані q_2, програма залишає в клітинці цифру 2 та виконує зупинення програми.

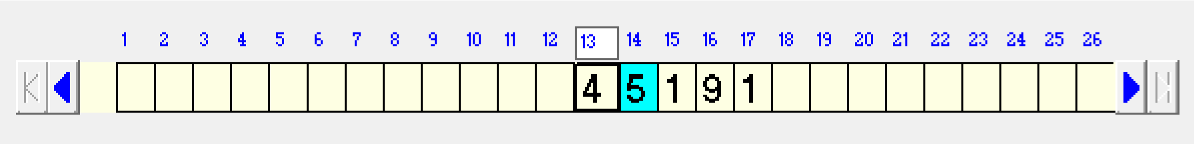
2	2 → Q_0	1 ⛔ Q_2	2 ⛔ Q_2

6.4 Увести натуральне число, яке закінчується двома нулями.



Проаналізувати результат та пояснити роботу програми в цьому випадку.

6.5 Увести натуральне число, яке закінчується одиницею.



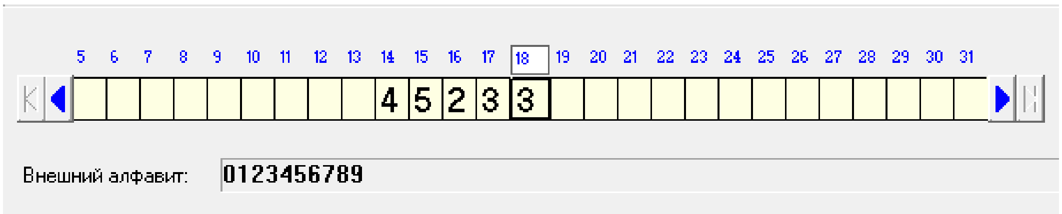
Проаналізувати результат.

Приклад 7

Розробити МТ, яка б зменшувала на одиницю задане натуральне число *n*, що є більшим за *1*, якщо головка знаходиться над правою цифрою числа, та зменшує на одиницю старшу цифру числа, якщо головка знаходиться над лівою цифрою.

7.1 Задати зовнішній алфавіт у вигляді наступної послідовності цифр: *0 1 2 3 4 5 6 7 8 9*.

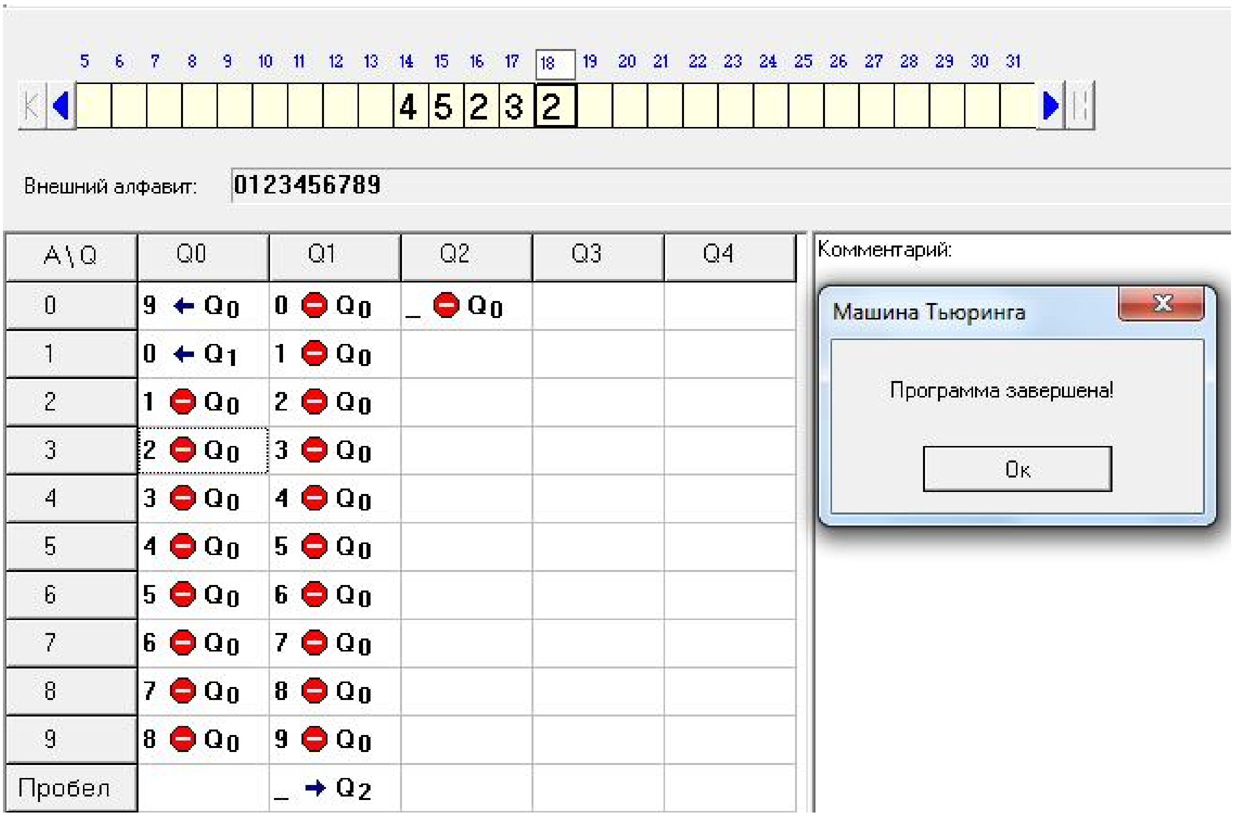
7.2 Заповнити таблицю МТ командами, як це наведено на рисунку 4.39.



A\Q	Q0	Q1	Q2	Q3	Q4
0	9 ← Q_0	0 ⛔ Q_0	_ ⛔ Q_0		
1	0 ← Q_1	1 ⛔ Q_0			
2	1 ⛔ Q_0	2 ⛔ Q_0			
3	2 ⛔ Q_0	3 ⛔ Q_0			
4	3 ⛔ Q_0	4 ⛔ Q_0			
5	4 ⛔ Q_0	5 ⛔ Q_0			
6	5 ⛔ Q_0	6 ⛔ Q_0			
7	6 ⛔ Q_0	7 ⛔ Q_0			
8	7 ⛔ Q_0	8 ⛔ Q_0			
9	8 ⛔ Q_0	9 ⛔ Q_0			
Пробел		_ → Q_2			

Рисунок 4.39 – Виконання операції віднімання з введеного числа 1 (зменшення числа на одиницю)

Результат роботи програми наведено на рисунку 4.40.



A\Q	Q0	Q1	Q2	Q3	Q4
0	9 ← Q0	0 ⛔ Q0	_ ⛔ Q0		
1	0 ← Q1	1 ⛔ Q0			
2	1 ⛔ Q0	2 ⛔ Q0			
3	2 ⛔ Q0	3 ⛔ Q0			
4	3 ⛔ Q0	4 ⛔ Q0			
5	4 ⛔ Q0	5 ⛔ Q0			
6	5 ⛔ Q0	6 ⛔ Q0			
7	6 ⛔ Q0	7 ⛔ Q0			
8	7 ⛔ Q0	8 ⛔ Q0			
9	8 ⛔ Q0	9 ⛔ Q0			
Пробел		_ → Q2			

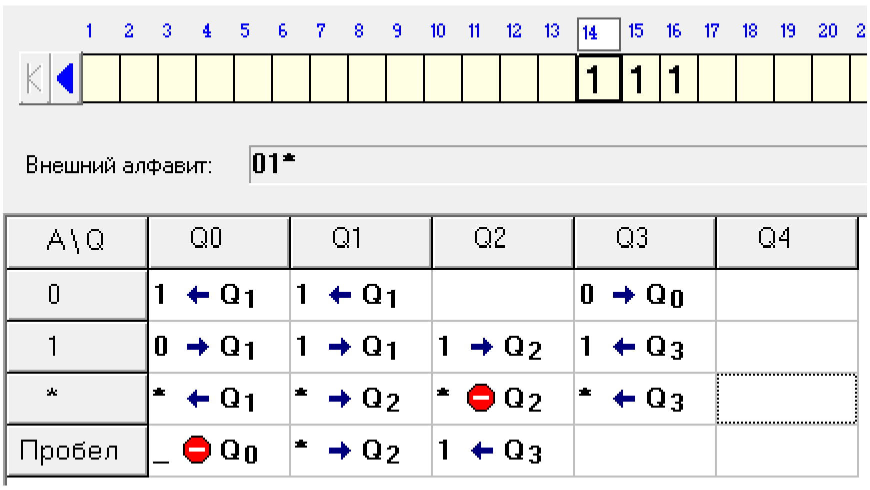
Рисунок 4.40 – Результат роботи програми

Приклад 8

Розробити МТ, яка б виконувала копіювання заданого аргументу.

8.1 Обрати вхідний алфавіт $A = \{0,\ 1,\ \varepsilon\}$, де ε – пустий символ (пробіл).

8.2 Записати програму МТ для заданого вхідного ланцюжка на стрічці, який дорівнює двійковому числу 111. Представити МТ у вигляді таблиці відповідності, як це наведено на рисунку 4.41.



A\Q	Q0	Q1	Q2	Q3	Q4
0	1 ← Q1	1 ← Q1		0 → Q0	
1	0 → Q1	1 → Q1	1 → Q2	1 ← Q3	
*	* ← Q1	* → Q2	* ⛔ Q2	* ← Q3	
Пробел	_ ⛔ Q0	* → Q2	1 ← Q3		

Рисунок 4.41 – Програма, призначена для копіювання аргументу

Зліва від кожної команди наведено представлення вхідного ланцюжка на стрічці до виконання певної команди. Символ, який знаходиться над головкою, позначено підкресленням [11].

1) $q_0 1 \to q_1 0R$	$\varepsilon\underline{1}11\varepsilon$ (у стані q_0 головка знаходиться під першою цифрою 1 та згідно з командою змінює стан на q_1, заміняє цифру 1 на цифру 0 та переміщується вправо);
2) $q_1 1 \to q_1 1R$	$\varepsilon 0\underline{1}1\varepsilon$ (у стані q_1 не виконується заміна цифри 1 в другій клітинці, головка переміщується вправо);
3) $q_1 1 \to q_1 1R$	$\varepsilon 01\underline{1}\varepsilon$ (у стані q_1 не виконується заміна цифри 1 у другій клітинці, головка переміщується вправо);
4) $q_1\varepsilon \to q_2{*}R$	$\varepsilon 011\underline{\varepsilon}$ (головка розміщується на символі ε, переходить у стан q_2, переміщується вправо, де додається символ «*». Головка залишилася на символі ε (див. п. 5));
5) $q_2\varepsilon \to q_3 1L$	$\varepsilon 011{*}\underline{\varepsilon;}$
6) $q_3{*} \to q_3{*}L$	$\varepsilon 011\underline{*}1\varepsilon$;
7) $q_3 1 \to q_3 1L$	$\varepsilon 01\underline{1}{*}1\varepsilon$;
8) $q_3 1 \to q_3 1L$	$\varepsilon 0\underline{1}1{*}1\varepsilon$;
9) $q_3 0 \to q_0 0R$	$\varepsilon\underline{0}11{*}1\varepsilon$;
10) $q_0 1 \to q_1 0R$	$\varepsilon 0\underline{1}1{*}1\varepsilon$;
11) $q_1 1 \to q_1 1R$	$\varepsilon 00\underline{1}{*}1\varepsilon$;
12) $q_1{*} \to q_2{*}R$	$\varepsilon 001\underline{*}1\varepsilon$;
13) $q_2 1 \to q_2 1R$	$\varepsilon 001{*}\underline{1}\varepsilon$;
14) $q_2\varepsilon \to q_3 1L$	$\varepsilon 001{*}1\underline{\varepsilon;}$
15) $q_3 1 \to q_3 1L$	$\varepsilon 001{*}\underline{1}1\varepsilon$;
16) $q_3{*} \to q_3{*}L$	$\varepsilon 001\underline{*}11\varepsilon$;
17) $q_3 1 \to q_3 1L$	$\varepsilon 00\underline{1}{*}11\varepsilon$;
18) $q_3 0 \to q_0 0R$	$\varepsilon 0\underline{0}1{*}11\varepsilon$;
19) $q_0 1 \to q_1 0R$	$\varepsilon 00\underline{0}{*}11\varepsilon$;
20) $q_1{*} \to q_2{*}R$	$\varepsilon 000\underline{*}11\varepsilon$;
21) $q_2 1 \to q_2 1R$	$\varepsilon 000{*}\underline{1}1\varepsilon$;
22) $q_2 1 \to q_2 1R$	$\varepsilon 000{*}1\underline{1}\varepsilon$;
23) $q_2\varepsilon \to q_3 1L$	$\varepsilon 000{*}11\underline{\varepsilon;}$
24) $q_3 1 \to q_3 1L$	$\varepsilon 000{*}1\underline{1}1\varepsilon$;
25) $q_3 1 \to q_3 1L$	$\varepsilon 000{*}\underline{1}11\varepsilon$;
26) $q_3{*} \to q_3{*}L$	$\varepsilon 000\underline{*}111\varepsilon$;
27) $q_3 0 \to q_0 0R$	$\varepsilon 00\underline{0}{*}111\varepsilon$;
28) $q_0{*} \to q_0{*}L$	$\varepsilon 000\underline{*}111\varepsilon$;
29) $q_0 0 \to q_0 1L$	$\varepsilon 00\underline{0}{*}111\varepsilon$;
30) $q_0 0 \to q_0 1L$	$\varepsilon 0\underline{0}1{*}111\varepsilon$;
31) $q_0 0 \to q_0 1L$	$\varepsilon\underline{0}11{*}111\varepsilon$;
32) $q_0\varepsilon \to q_к 1R$	$\underline{\varepsilon}111{*}111\varepsilon$.

Із прикладу видно, що МТ із стандартної початкової конфігурації, маючи на стрічці аргумент 111 та виконавши сукупність команд 1–32, перейшла в стандартний заключний стан, маючи на стрічці результат *ε111*111ε*.

Результат роботи програми в ALGO 2000 наведено на рисунку 4.42.

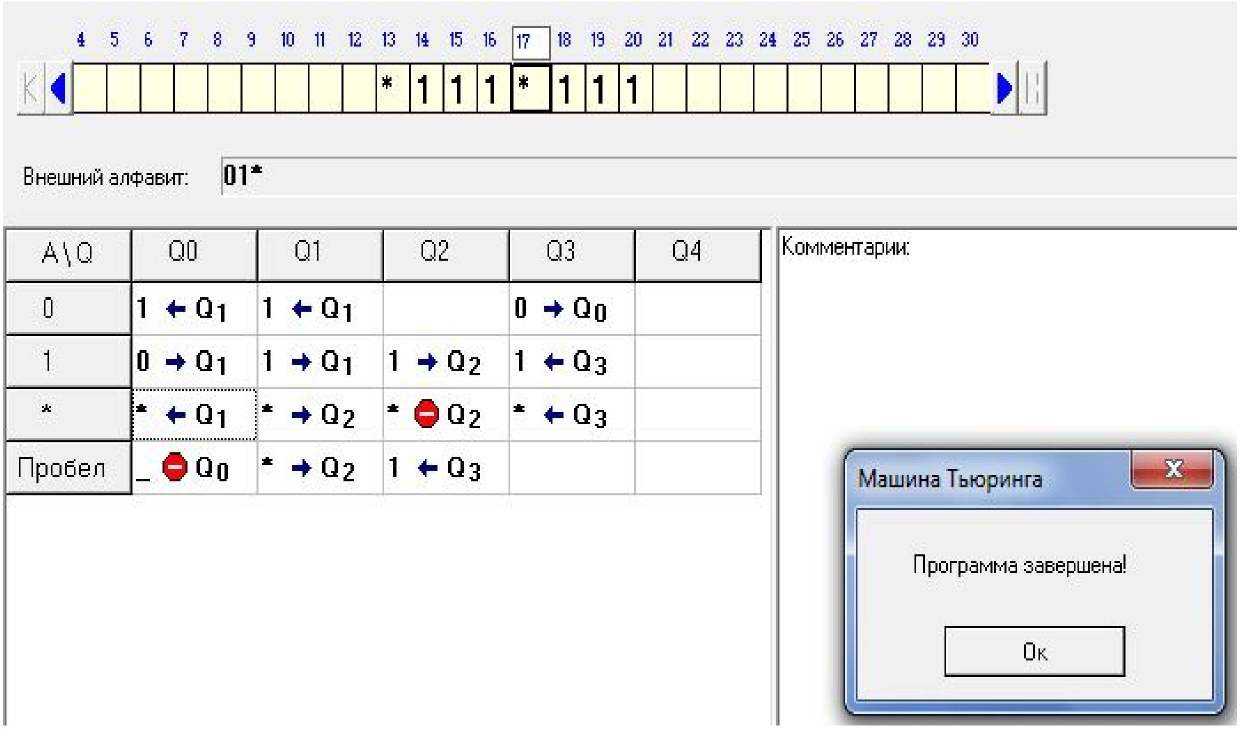


Рисунок 4.42 – Результат роботи програми, яка призначена для копіювання аргументу

8.3 Увести інше двійкове число у вигляді одиниць для копіювання. Проаналізувати отримані результати.

8.4 Доопрацювати наведену програму для введення нулів.

Приклад 9

Розробити МТ, яка визначає, чи ділиться на 5 без залишку десяткове число, розташоване на інформаційній стрічці. Якщо число ділиться без залишку на 5, то справа від числа потрібно написати слово «ТАК», якщо не ділиться – слово «НІ». Головка знаходиться над лівою цифрою числа.

9.1 Обрати вхідний алфавіт: *A ={0, 1, 2, 3, 4, 5, 6, 7, 8, 9, T, A, K, H, I}*.

9.2 Представити МТ у вигляді таблиці відповідності, як це наведено на рисунку 4.43.

Внешний алфавит: 0123456789AIKHT

A\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
0	0 → Q_1	0 → Q_1	0 → Q_3					
1	1 → Q_1	1 → Q_1	1 → Q_6					
2	2 → Q_1	2 → Q_1	2 → Q_6					
3	3 → Q_1	3 → Q_1	3 → Q_6					
4	4 → Q_1	4 → Q_1	4 → Q_6					
5	5 → Q_1	5 → Q_1	5 → Q_3					
6	6 → Q_1	6 → Q_1	6 → Q_6					
7	7 → Q_1	7 → Q_1	7 → Q_6					
8	8 → Q_1	8 → Q_1	8 → Q_6					
9	9 → Q_1	9 → Q_1	9 → Q_6					
A								
I								
K								
H								
T								
Пробел		_ ← Q_2		T → Q_4	A → Q_5	K ⛔ Q_0	H → Q_7	I ⛔ Q_7

Рисунок 4.43 – Функціональна схема для визначення кратності 5

У цій програмі:

– стан q_1 слугує для пошуку правого кінця числа;

– стан q_2 слугує для аналізу молодшої цифри числа. Якщо вона дорівнює цифрі 0 або цифрі 5, тобто число ділиться на 5, то виконується перехід у стан q_3, інакше – у стан q_5;

– стан q_3 слугує для запису літери «Т» справа від числа на стрічці;

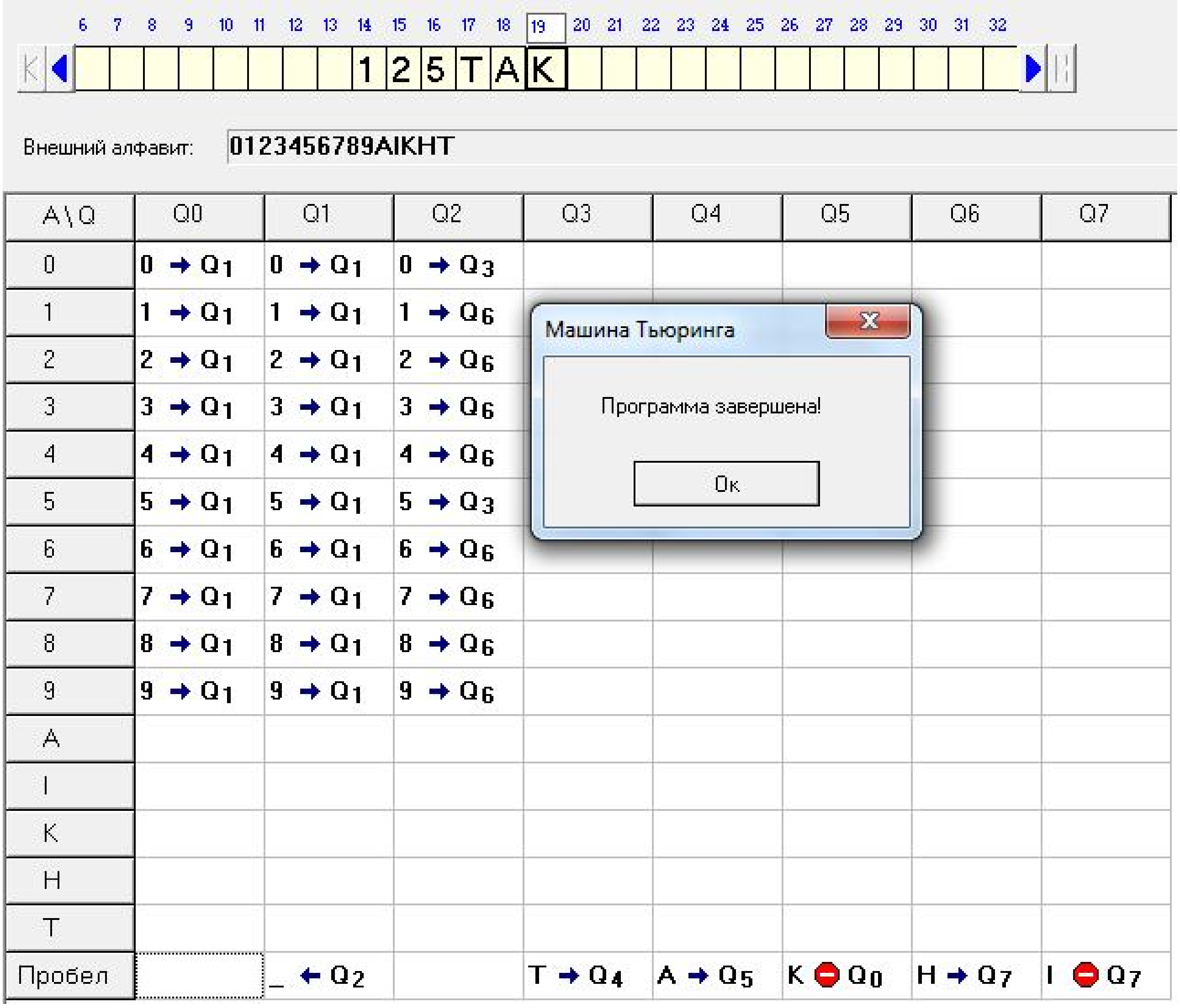
– стан q_4 слугує для запису літери «А» справа від літери «Т» на стрічці;

– стан q_5 слугує для запису літери «К» справа від літери «А» на стрічці та зупинення машини;

– стан q_6 слугує для запису літери «Н» справа від літери «К» на стрічці;

– стан q_7 слугує для запису літери «І» справа від літери «Н» на стрічці та зупинення машини.

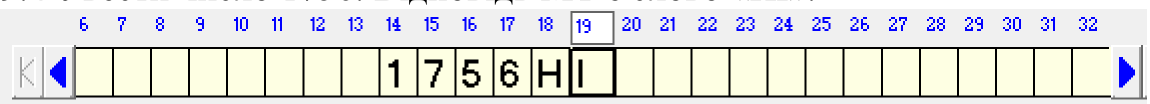
9.3 Увести число, кратне 5, наприклад, 125. Результат роботи програми наведено на рисунку 4.44.



A\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
0	0 → Q_1	0 → Q_1	0 → Q_3					
1	1 → Q_1	1 → Q_1	1 → Q_6					
2	2 → Q_1	2 → Q_1	2 → Q_6					
3	3 → Q_1	3 → Q_1	3 → Q_6					
4	4 → Q_1	4 → Q_1	4 → Q_6					
5	5 → Q_1	5 → Q_1	5 → Q_3					
6	6 → Q_1	6 → Q_1	6 → Q_6					
7	7 → Q_1	7 → Q_1	7 → Q_6					
8	8 → Q_1	8 → Q_1	8 → Q_6					
9	9 → Q_1	9 → Q_1	9 → Q_6					
A								
I								
K								
H								
T								
Пробел		_ ← Q_2		T → Q_4	A → Q_5	K ⛔ Q_0	H → Q_7	I ⛔ Q_7

Рисунок 4.44 – Результат роботи програми

9.4 Увести число 1756. Відповідь МТ є слово «HI».



9.5 Удосконалити МТ так, щоб відповіді МТ писала англійською мовою: «YES» або «NO».

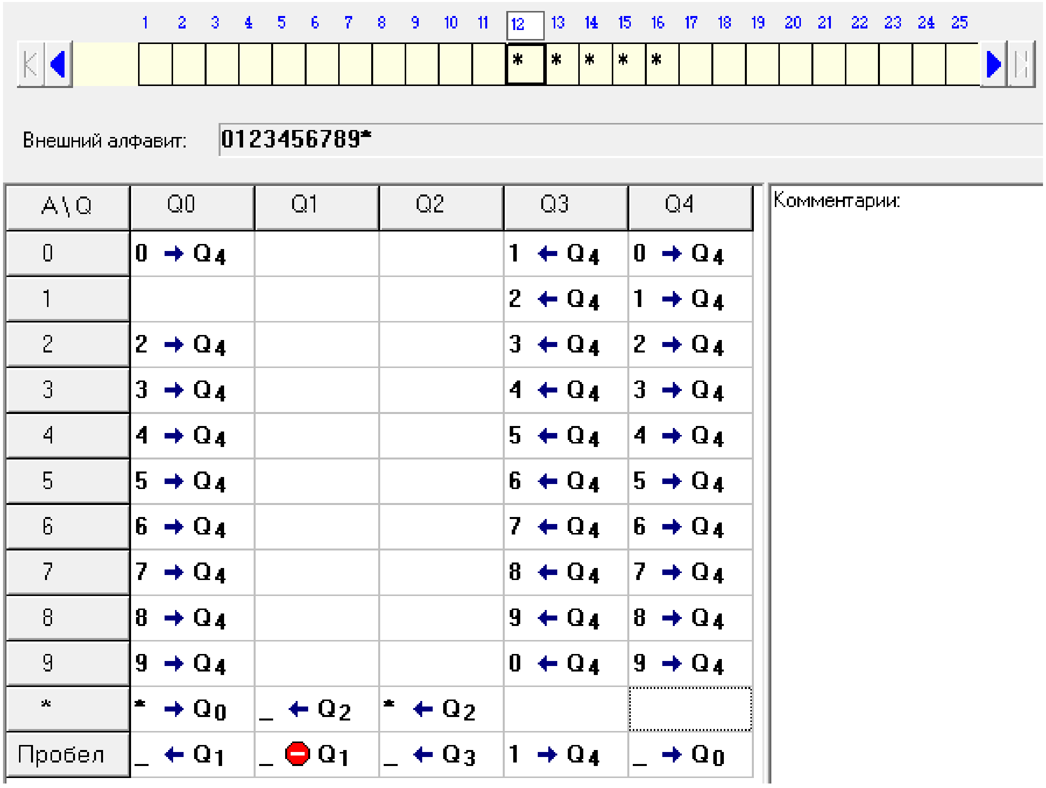
Приклад 10

Розробити МТ, яка б підраховувала кількість введених на інформаційну стрічку символів «*». Якщо ця кількість менша або дорівнює 9, то програма виводить цю кількість числом десяткової системи числення.

Головка може бути розташована або на першій або на останній клітинці стрічки.

10.1 Обрати вхідний алфавіт, який складається з цифр та символу: *0 1 2 3 4 5 6 7 8 9 *.*

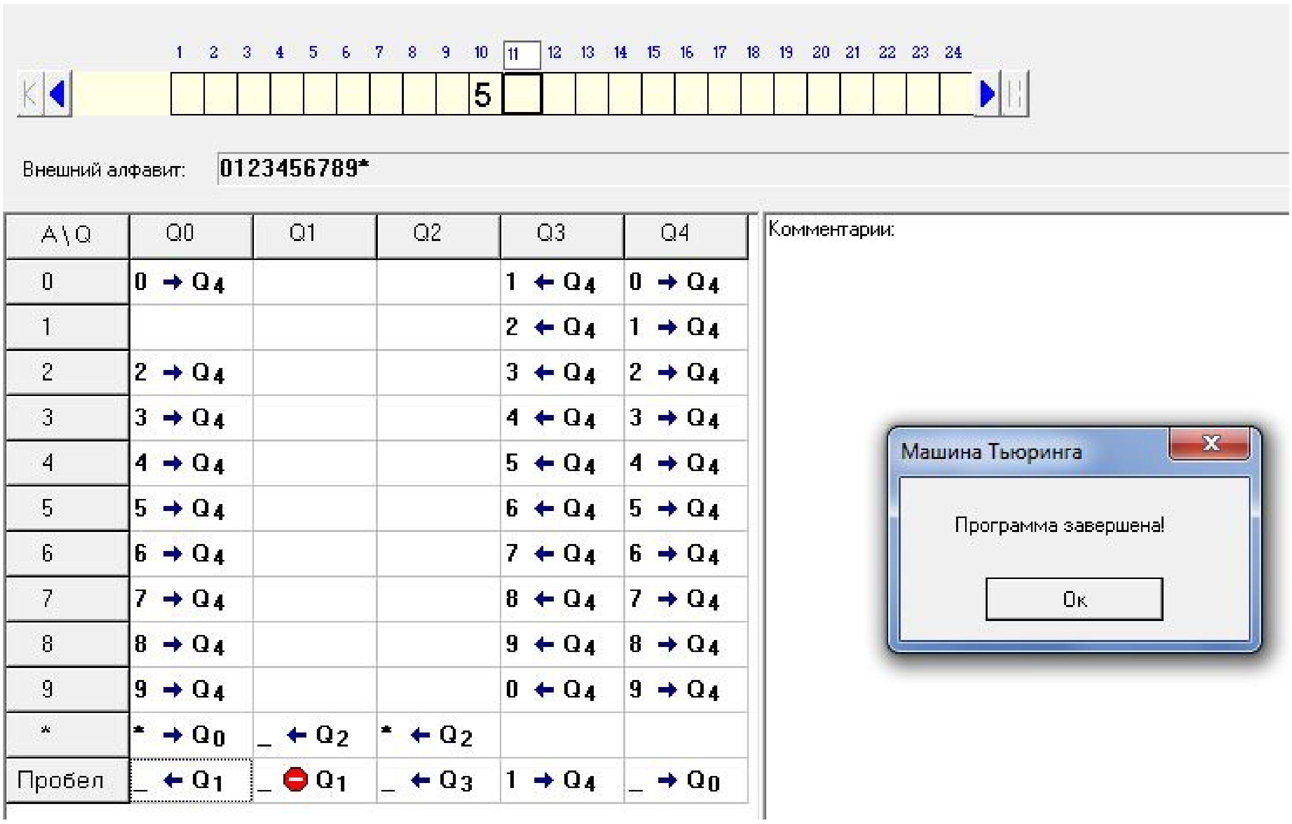
10.2 Представити МТ у вигляді таблиці відповідності (рис. 4.45).



A\Q	Q0	Q1	Q2	Q3	Q4
0	0 → Q4			1 ← Q4	0 → Q4
1				2 ← Q4	1 → Q4
2	2 → Q4			3 ← Q4	2 → Q4
3	3 → Q4			4 ← Q4	3 → Q4
4	4 → Q4			5 ← Q4	4 → Q4
5	5 → Q4			6 ← Q4	5 → Q4
6	6 → Q4			7 ← Q4	6 → Q4
7	7 → Q4			8 ← Q4	7 → Q4
8	8 → Q4			9 ← Q4	8 → Q4
9	9 → Q4			0 ← Q4	9 → Q4
*	* → Q0	_ ← Q2	* ← Q2		
Пробел	_ ← Q1	_ ⛔ Q1	_ ← Q3	1 → Q4	_ → Q0

Рисунок 4.45 – Таблиця відповідності для підрахунку кількості символів «*»

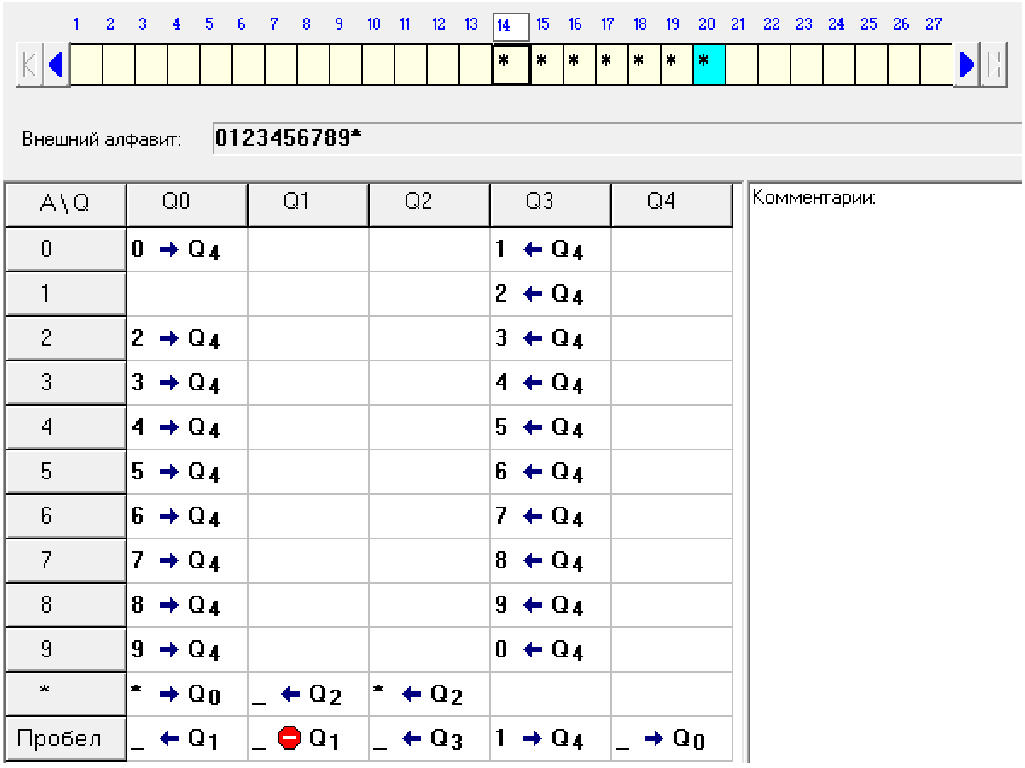
Результат роботи програми наведено на рисунку 4.46.



A\Q	Q0	Q1	Q2	Q3	Q4
0	0 → Q4			1 ← Q4	0 → Q4
1				2 ← Q4	1 → Q4
2	2 → Q4			3 ← Q4	2 → Q4
3	3 → Q4			4 ← Q4	3 → Q4
4	4 → Q4			5 ← Q4	4 → Q4
5	5 → Q4			6 ← Q4	5 → Q4
6	6 → Q4			7 ← Q4	6 → Q4
7	7 → Q4			8 ← Q4	7 → Q4
8	8 → Q4			9 ← Q4	8 → Q4
9	9 → Q4			0 ← Q4	9 → Q4
*	* → Q0	_ ← Q2	* ← Q2		
Пробел	_ ← Q1	_ ⛔ Q1	_ ← Q3	1 → Q4	_ → Q0

Рисунок 4.46 – Результат роботи програми

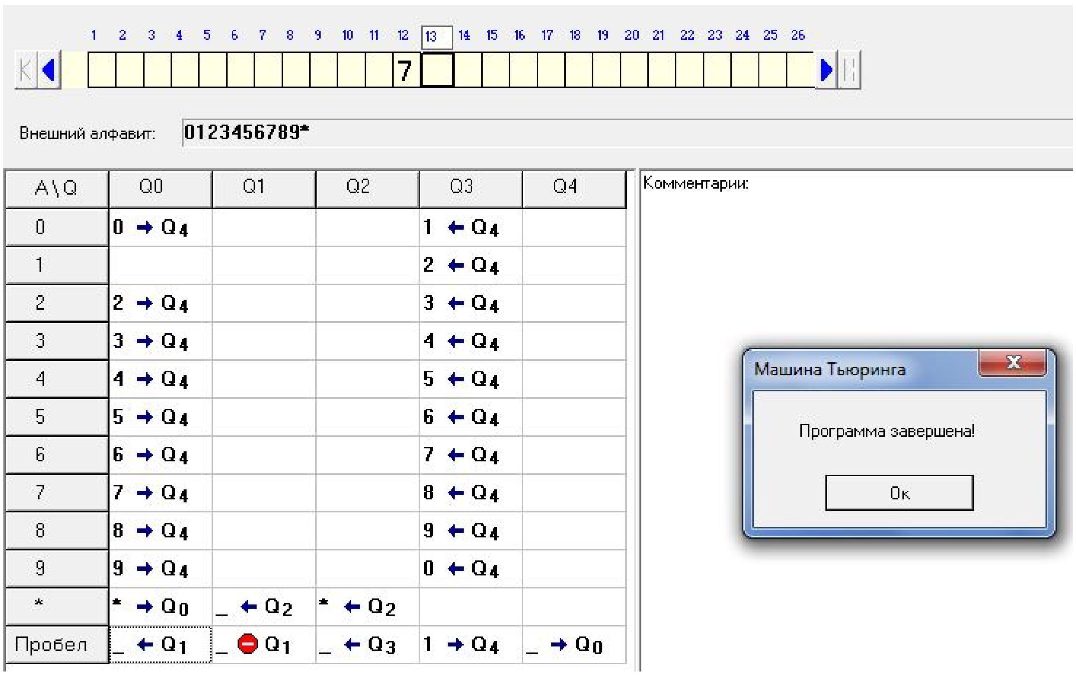
Програму для підрахунку кількості символів «*» можна спростити, що наведено на рисунку 4.47.



A\Q	Q0	Q1	Q2	Q3	Q4
0	0 → Q4			1 ← Q4	
1				2 ← Q4	
2	2 → Q4			3 ← Q4	
3	3 → Q4			4 ← Q4	
4	4 → Q4			5 ← Q4	
5	5 → Q4			6 ← Q4	
6	6 → Q4			7 ← Q4	
7	7 → Q4			8 ← Q4	
8	8 → Q4			9 ← Q4	
9	9 → Q4			0 ← Q4	
*	* → Q0	_ ← Q2	* ← Q2		
Пробел	_ ← Q1	_ Q1	_ ← Q3	1 → Q4	_ → Q0

Рисунок 4.47 – Модифікована програма підрахунку символів «*»

Результат роботи модифікованої програми наведено на рисунку 4.48.



A\Q	Q0	Q1	Q2	Q3	Q4
0	0 → Q4			1 ← Q4	
1				2 ← Q4	
2	2 → Q4			3 ← Q4	
3	3 → Q4			4 ← Q4	
4	4 → Q4			5 ← Q4	
5	5 → Q4			6 ← Q4	
6	6 → Q4			7 ← Q4	
7	7 → Q4			8 ← Q4	
8	8 → Q4			9 ← Q4	
9	9 → Q4			0 ← Q4	
*	* → Q0	_ ← Q2	* ← Q2		
Пробел	_ ← Q1	_ Q1	_ ← Q3	1 → Q4	_ → Q0

Рисунок 4.48 – Результат роботи модифікованої програми підрахунку символів «*»

10.3 Доопрацювати програму так, щоб з її допомогою можна було обраховувати кількість введених символів «*», більших за число 9.

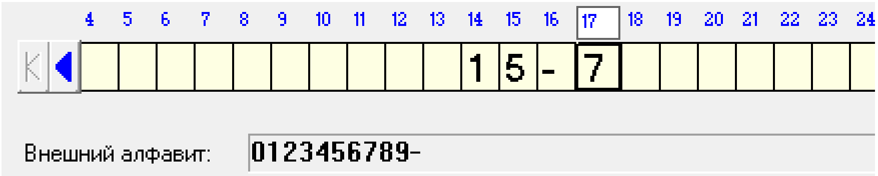
Приклад 11

Розробити МТ, яка виконує дію віднімання двох чисел, записаних у десятковій системі числення.

Автомат у стані q_0 оглядає крайню праву цифру числа.

11.1 Обрати вхідний алфавіт, який складається з цифр та символу: *0 1 2 3 4 5 6 7 8 9 –*.

11.2 Представити МТ у вигляді таблиці відповідності (рис. 4.49).



A\Q	Q0	Q1	Q2	Q3	Q4
0	9 ← Q_0	0 ← Q_1	9 ← Q_2	0 → Q_3	
1	0 ← Q_1	1 ← Q_1	0 → Q_3	1 → Q_3	
2	1 ← Q_1	2 ← Q_1	1 → Q_3	2 → Q_3	
3	2 ← Q_1	3 ← Q_1	2 → Q_3	3 → Q_3	
4	3 ← Q_1	4 ← Q_1	3 → Q_3	4 → Q_3	1 → Q_2
5	4 ← Q_1	5 ← Q_1	4 → Q_3	5 → Q_3	
6	5 ← Q_1	6 ← Q_1	5 → Q_3	6 → Q_3	
7	6 ← Q_1	7 ← Q_1	6 → Q_3	7 → Q_3	
8	7 ← Q_1	8 ← Q_1	7 → Q_3	8 → Q_3	
9	8 ← Q_1	9 ← Q_1	8 → Q_3	9 → Q_3	_ ← Q_4
-	- → Q_4	- ← Q_2		- → Q_3	_ → Q_4
Пробел				_ ← Q_0	_ ⛔ Q_4

Рисунок 4.49 – Функціональна схема віднімання двох чисел

У цій програмі:

– стан q_0 слугує для віднімання одиниці;

– стан q_1 слугує для здійснення переходу до першого числа;

– стан q_2 слугує для віднімання одиниці від першого числа;

– стан q_3 слугує для зсуву каретки вправо до пробіла;

– стан q_4 слугує для знищення зайвих символів.

Переміщення каретки до знака « – » означає, що в стані q_0 друге число вичерпане.

Результат роботи програми наведено на рисунку 4.50.

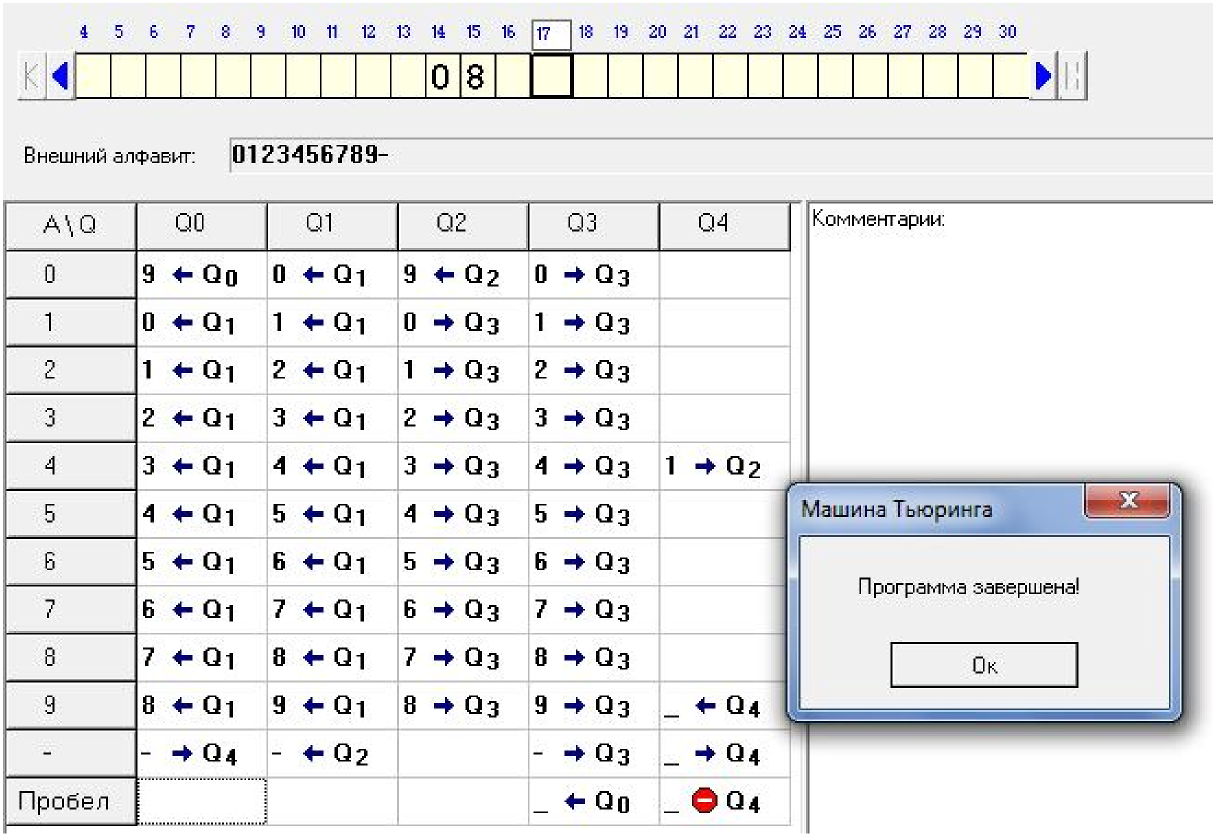


Рисунок 4.50 – Результат віднімання двох чисел

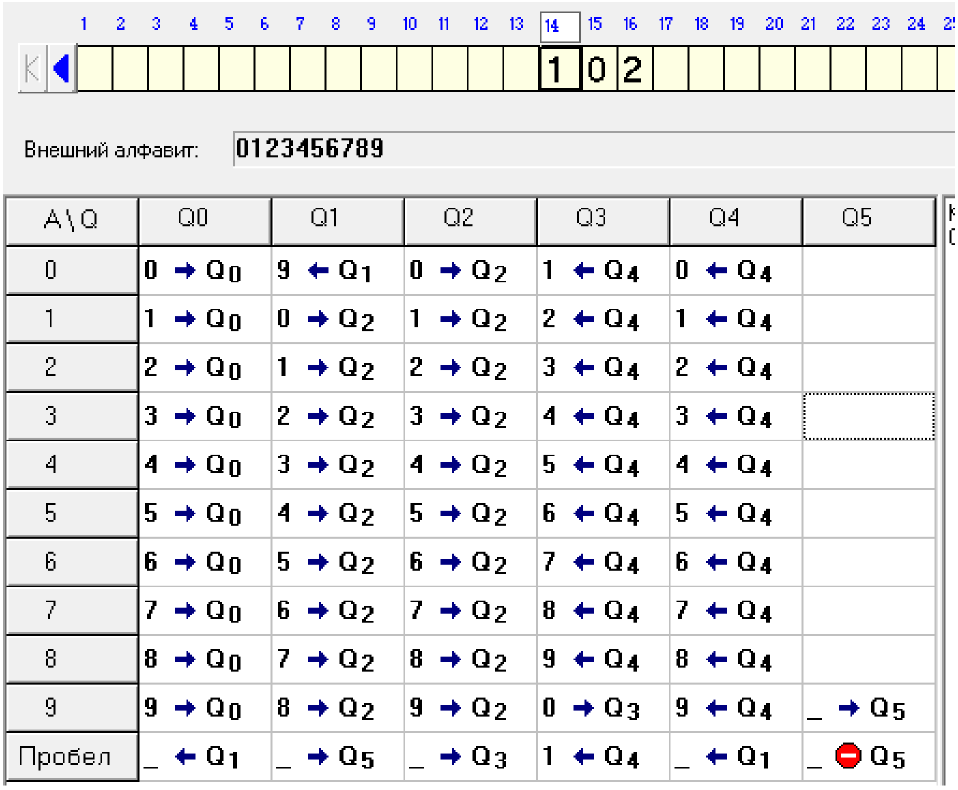
Приклад 12

Скласти функціональну схему МТ, яка могла б число, подане в десятковій системі числення, записувати в зворотному порядку.

Каретка може знаходиться як на крайній правій цифрі числа, так і на крайній лівій цифрі числа.

12.1 Обрати вхідний алфавіт, який складається з цифр: *0 1 2 3 4 5 6 7 8 9.*

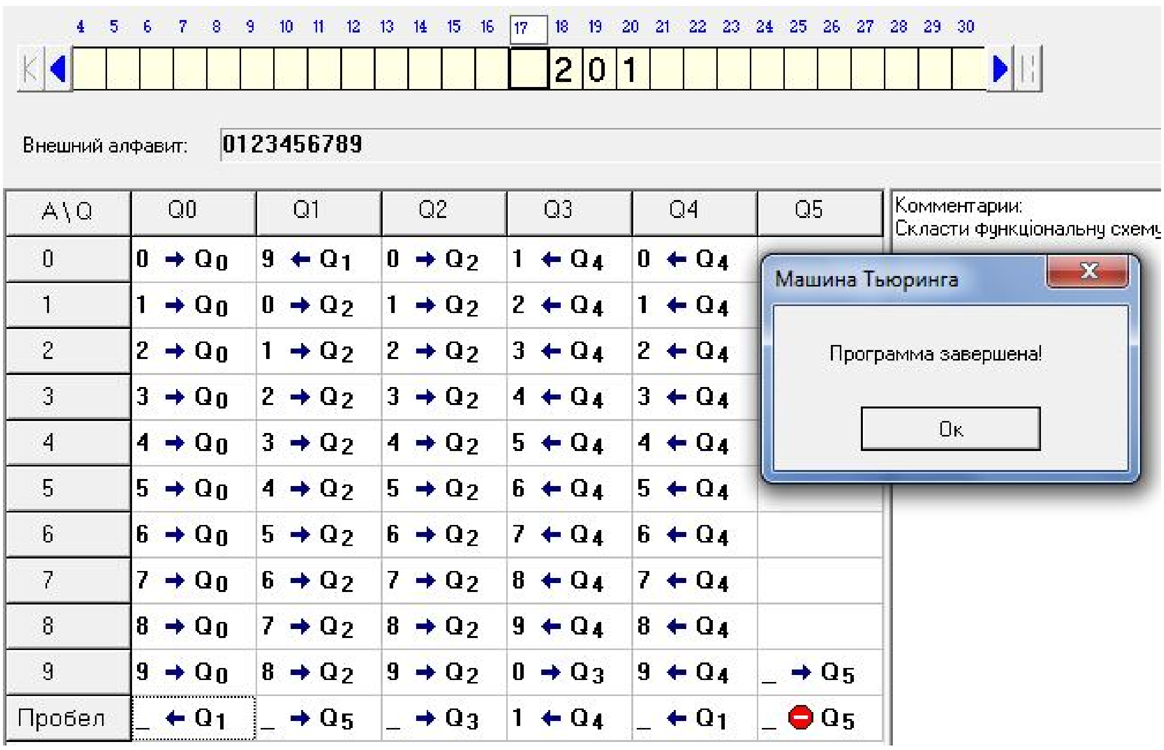
12.2 Представити МТ у вигляді таблиці відповідності (рис. 4.51).



A\Q	Q0	Q1	Q2	Q3	Q4	Q5
0	0 → Q0	9 ← Q1	0 → Q2	1 ← Q4	0 ← Q4	
1	1 → Q0	0 → Q2	1 → Q2	2 ← Q4	1 ← Q4	
2	2 → Q0	1 → Q2	2 → Q2	3 ← Q4	2 ← Q4	
3	3 → Q0	2 → Q2	3 → Q2	4 ← Q4	3 ← Q4	
4	4 → Q0	3 → Q2	4 → Q2	5 ← Q4	4 ← Q4	
5	5 → Q0	4 → Q2	5 → Q2	6 ← Q4	5 ← Q4	
6	6 → Q0	5 → Q2	6 → Q2	7 ← Q4	6 ← Q4	
7	7 → Q0	6 → Q2	7 → Q2	8 ← Q4	7 ← Q4	
8	8 → Q0	7 → Q2	8 → Q2	9 ← Q4	8 ← Q4	
9	9 → Q0	8 → Q2	9 → Q2	0 → Q3	9 ← Q4	_ → Q5
Пробел	_ ← Q1	_ → Q5	_ → Q3	1 ← Q4	_ ← Q1	_ Q5

Рисунок 4.51 – Функціональна схема МТ

12.3 Увести число 102 та запустити програму для виконання. Результат роботи МТ наведено на рисунку 4.52.



A\Q	Q0	Q1	Q2	Q3	Q4	Q5
0	0 → Q0	9 ← Q1	0 → Q2	1 ← Q4	0 ← Q4	
1	1 → Q0	0 → Q2	1 → Q2	2 ← Q4	1 ← Q4	
2	2 → Q0	1 → Q2	2 → Q2	3 ← Q4	2 ← Q4	
3	3 → Q0	2 → Q2	3 → Q2	4 ← Q4	3 ← Q4	
4	4 → Q0	3 → Q2	4 → Q2	5 ← Q4	4 ← Q4	
5	5 → Q0	4 → Q2	5 → Q2	6 ← Q4	5 ← Q4	
6	6 → Q0	5 → Q2	6 → Q2	7 ← Q4	6 ← Q4	
7	7 → Q0	6 → Q2	7 → Q2	8 ← Q4	7 ← Q4	
8	8 → Q0	7 → Q2	8 → Q2	9 ← Q4	8 ← Q4	
9	9 → Q0	8 → Q2	9 → Q2	0 → Q3	9 ← Q4	_ → Q5
Пробел	_ ← Q1	_ → Q5	_ → Q3	1 ← Q4	_ ← Q1	_ Q5

Рисунок 4.52 – Результат роботи МТ для запису числа в зворотному порядку

Контрольні питання

1. Як зміниться робота програми (рис. 4.31), якщо головка буде знаходитися не на першій цифрі числа, а на останній?
2. Яку відповідь буде отримано після вводу числа 12370, якщо використати програму, наведену на рисунку 4.31?
3. Яку відповідь буде отримано після вводу числа 12477, якщо використати програму, наведену на рисунку 4.31?
4. Який результат буде отримано за використання програми, зображеної на рисунку 4.41, якщо замість одиниць ввести нулі?
5. Які дії будуть виконуватися, якщо в програмі, наведеній на рисунку 4.49, використовувати тільки перші два стовпчика програми?

4.3 Практичне заняття 3
Обробка масивів та символьних даних
у програмному інтерпретаторі ALGO 2000

Мета заняття – обробка літерних символів та масивів.

Завдання на виконання практичного заняття

1. Розробити МТ в алфавіті $A = \{0, 1\}$, яка б починала роботу з останньої одиниці масиву, що складається з декількох одиниць, зсувала його на одну клітинку вліво, залишаючи без змін решту вмісту стрічки. Головка зупиняється на першій одиниці перенесеного масиву.
2. Розробити МТ, яка б у вхідний ланцюжок, заданий в алфавіті нулем та одиницею, переставляла одиниці й нулі так, щоб усі одиниці були розташовані на початку, а нулі – на кінці ланцюжка.
3. Розробити МТ, яка б починала роботу з першої одиниці масиву, що складається з одиниць, зсувала його на дві клітинки вправо, не змінюючи при цьому решту вмісту стрічки, та зупинялась би на останній одиниці перенесеного масиву.
4. Розробити МТ, яка б починала роботу з лівої непустої клітинки, знаходила нуль, після якого розташований масив з трьох одиниць, з правого боку якого також розташовані нулі, та зупинялась на першій одиниці знайденого масиву, якщо такий є. Вміст стрічки не змінюється.
5. Розробити МТ, яка б починала роботу з довільної клітинки та переміщуючись вправо, записувала підряд $2n$ нулів за заданого $n>=1$ і зупинялась на останньому з них.
6. Розробити МТ, яка б переміщуючись вправо від будь-якої пустої клітинки, знаходила перший за такого переміщення масив, що містить не менше семи одиниць, стирала в ньому перші сім одиниць та зупинялась на

клітинці, розташованій найбільш правіше від тих, у яких були стерті одиниці. Інший вміст стрічки не змінюється.

7. Розробити МТ, яка б з $n>=2$ записаних на стрічці одиниць залишала на стрічці $n–2$ одиниці, записані підряд.

8. Скласти функціональну схему МТ, яка б могла міняти місцями крайні літери з трьох записаних у довільному порядку літер: *A, B, C*. Каретка в стані q_0 оглядає літеру, розташовану справа [12].

Загальні положення

Алгоритми становлять послідовність кроків, реалізованих строго описаною математичною моделлю машини з нескінченною пам’яттю.

Станом або конфігурацією МТ називається стан стрічки в сукупності з номером k поточної клітинки стрічки, з якої зчитується інформація.

Стан МТ може записуватися одним словом, записаним в алфавіті, яке є об’єднанням зовнішнього та внутрішнього алфавітів.

Можна визначити МТ із будь-яким раніше заданим зовнішнім та внутрішнім алфавітами. Для того щоб машина почала працювати, її необхідно привести в деякий початковий стан та запустити в роботу.

Нехай визначено деяке машинне слово W_1. Тоді наступне машинне слово, яке отримується в результаті одного кроку роботи машини, можна позначити через W_2, наступне – через W_3 і так далі. У результаті роботи МТ отримується така послідовність слів: $W_1, W_2 \dots W_n$.

Множина всіх команд, які записані для конкретної машини Тюрінга і які використовуються машиною в процесі роботи, називається програмою.

Під час дослідження алгоритмів в теорії алгоритмів зазвичай розглядається не будова машини, а процеси заміни слів-станів як заміна одного машинного слова на інше.

Таким чином, можна розглядати алгоритми як процеси змінення машинних слів шляхом заміни одних символів, що входять до слова, на інші [14–15].

Приклад 1

Розробити МТ, яка б зменшувала в два рази масив, що складається з *2N* елементів. Головка розташована на лівій клітинці.

1.1 Обрати вхідний алфавіт, який складається з символів: *|.

1.2 Представити МТ у вигляді таблиці відповідності.

1.3 Увести на інформаційну стрічку масив із парної кількості символів « | », наприклад 6 (рис. 4.53).

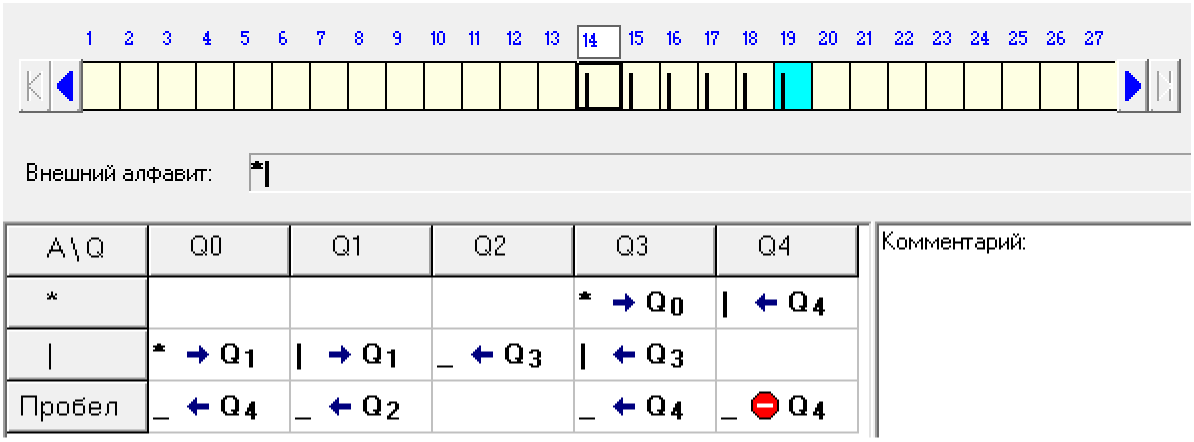


Рисунок 4.53 – МТ для обробки масиву символів « | »

1.4 Запустити програму для виконання.

Результат роботи програми наведено на рисунку 4.54.

1.5 Увести іншу послідовність символів « | ». Проаналізувати роботу програми в цьому випадку.

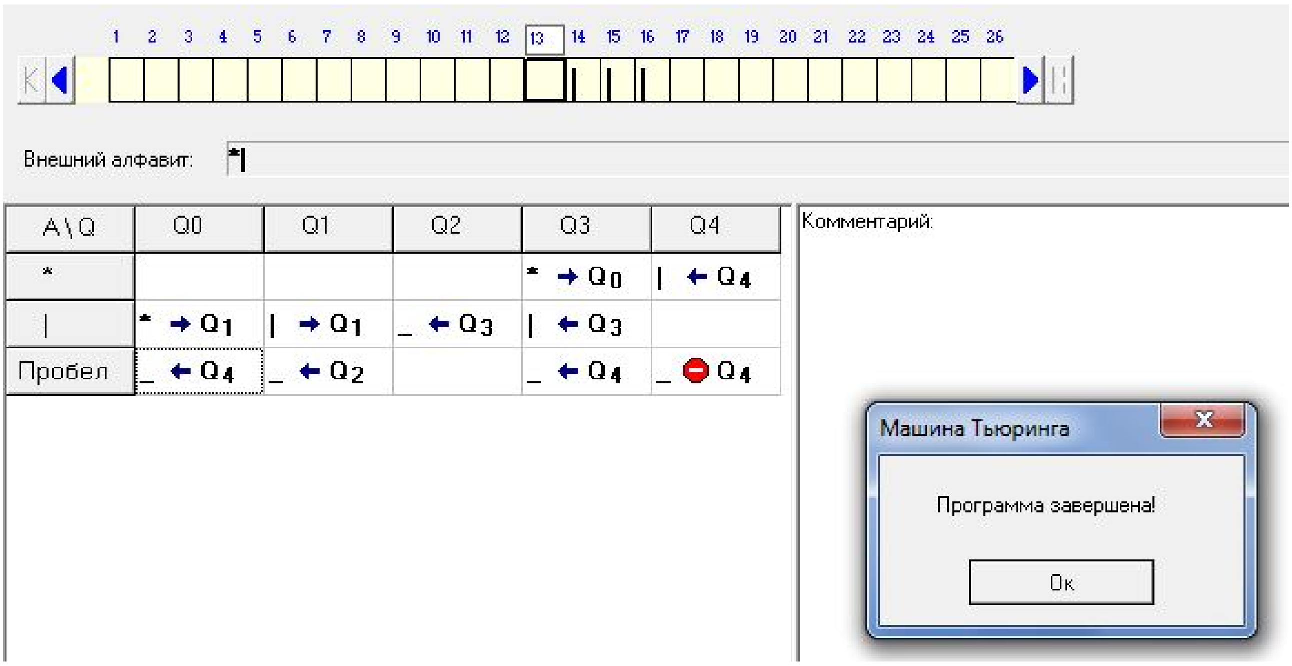


Рисунок 4.54 – Результат зменшення в два рази початкового масиву

Приклад 2

Розробити МТ, яка б з масиву, що складається з літер *A* та *B*, видаляла всі літери *B*, виконуючи також групування літер *A*.

Головка може бути розташованою як на лівій, так і на правій клітинках інформаційної стрічки.

2.1 Обрати вхідний алфавіт, що складається з літер *A* і *B* та символу *.

2.2 Представити МТ у вигляді таблиці відповідності (рис. 4.55).

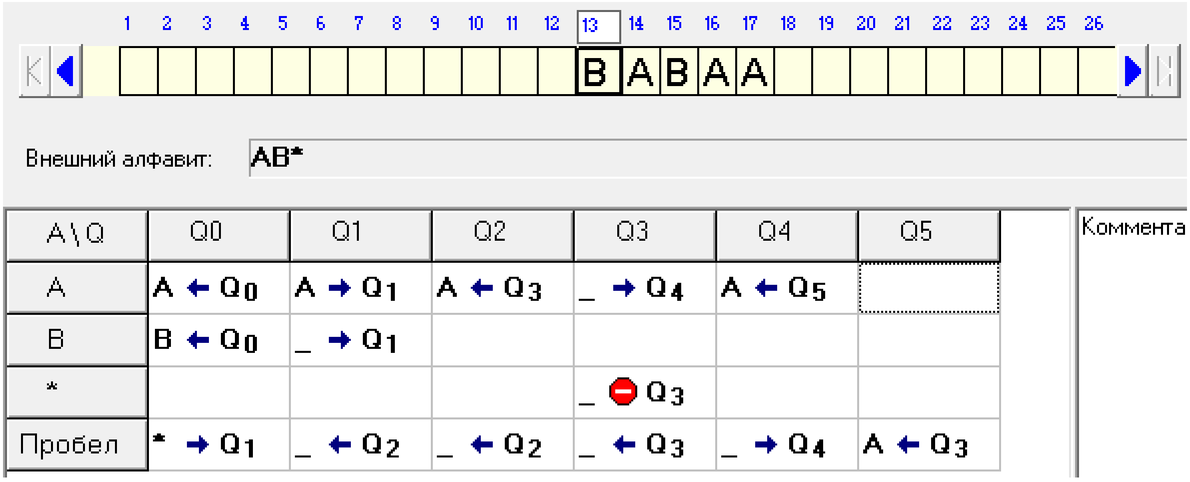


Рисунок 4.55 – Уведення початкового масиву

Результат стискання довільного масиву наведено на рисунку 4.56.

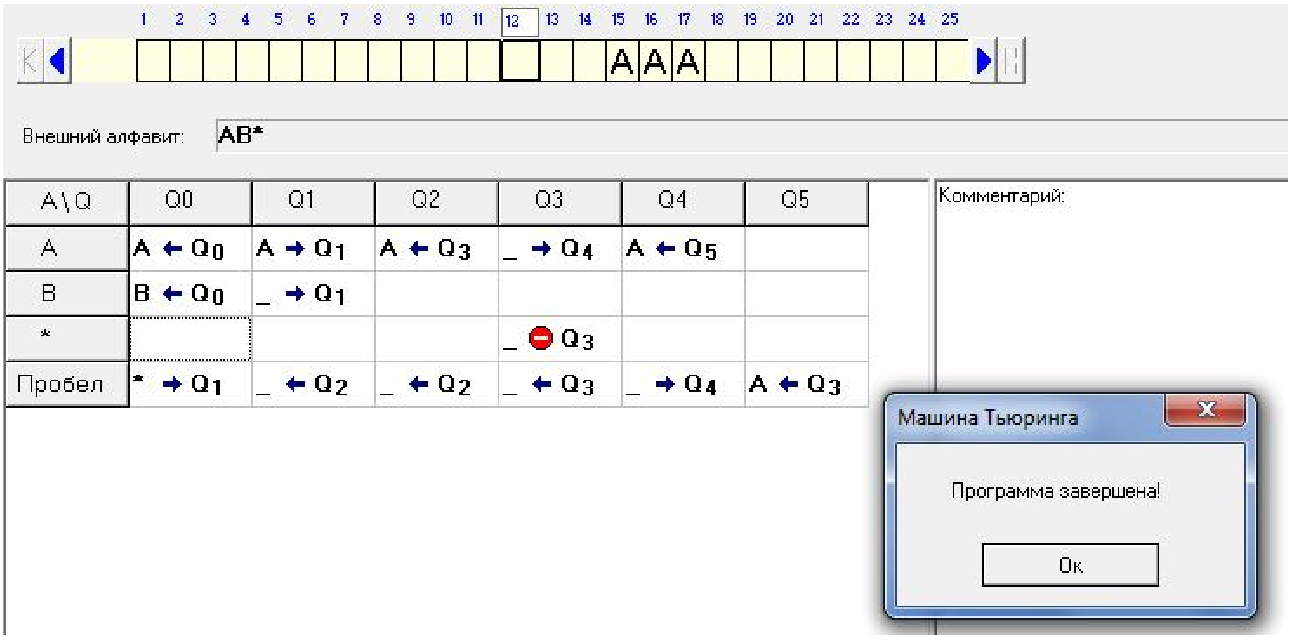


Рисунок 4.56 – Результат стискання довільного масиву

2.3 Увести інший масив, проаналізувати отримані результати.

Приклад 3

Розробити МТ, яка б опрацьовувала слово з літер *A* і *B* так, щоб зліва були розташовані літери *A*, а справа – літери *B*. Головка знаходиться на лівій літері.

3.1 Обрати вхідний алфавіт, який складається з літер та символів: *A B* * |.

3.2 Представити МТ у вигляді таблиці відповідності (рис. 4.57).

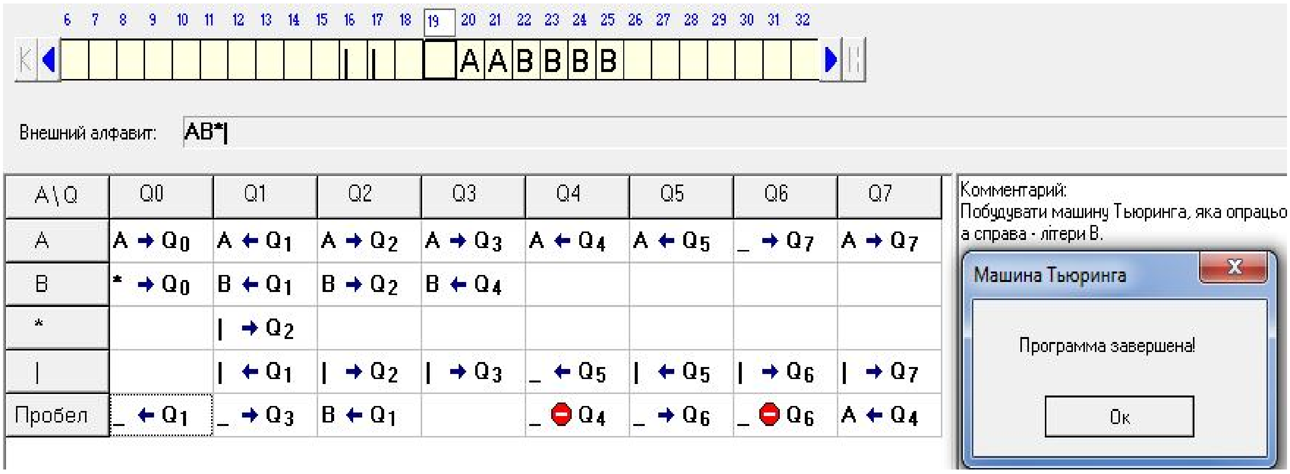
Внешний алфавит: AB*|

A\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
A	A → Q0	A ← Q1	A → Q2	A → Q3	A ← Q4	A ← Q5	_ → Q7	A → Q7
B	* → Q0	B ← Q1	B → Q2	B ← Q4				
*		| → Q2						
|		| ← Q1	| → Q2	| → Q3	_ ← Q5	| ← Q5	| → Q6	| → Q7
Пробел	_ ← Q1	_ → Q3	B ← Q1		_ ⛔ Q4	_ → Q6	_ ⛔ Q6	A ← Q4

Рисунок 4.57 – Програма обробки слова з літер *A* і *B*

3.3 Увести слово з послідовності літер *A* і *B*, наприклад *BBAABB*.

Результат обробки слова *BBAABB* наведено на рисунку 4.58.



A\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7
A	A → Q0	A ← Q1	A → Q2	A → Q3	A ← Q4	A ← Q5	_ → Q7	A → Q7
B	* → Q0	B ← Q1	B → Q2	B ← Q4				
*		| → Q2						
|		| ← Q1	| → Q2	| → Q3	_ ← Q5	| ← Q5	| → Q6	| → Q7
Пробел	_ ← Q1	_ → Q3	B ← Q1		_ ⛔ Q4	_ → Q6	_ ⛔ Q6	A ← Q4

Рисунок 4.58 – Результат обробки введеного слова

Проаналізувати роботу програми та отримані результати.

3.4 Удосконалити програму роботи МТ так, щоб на інформаційній стрічці не виводилися символи « | » у кінці роботи програми як це показано на рисунку 4.58.

3.5 Увести інше слово, наприклад *ABBA*.

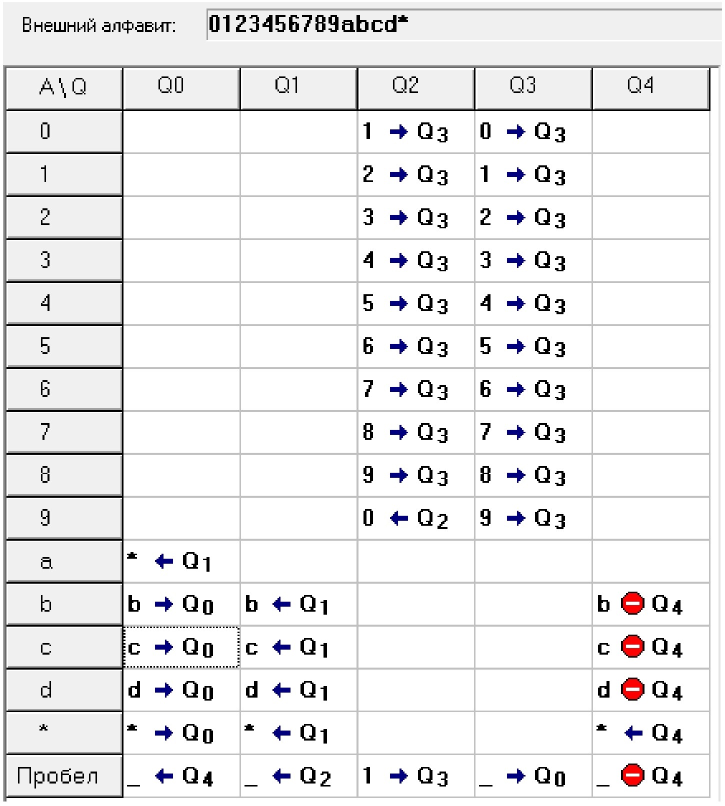
Проаналізувати роботу програми та отримані результати.

Приклад 4

Розробити МТ, яка б підраховувала в уведеному на інформаційну стрічку слові з літер *a, b, c, d* кількість літер *a*, замінювала в слові всі літери *a* на символ «*» та виводила кількість літер *a* зліва від цього слова через одну клітинку. Каретка розташовується на крайній лівій літері.

4.1 Обрати вхідний алфавіт, який складається з цифр, літер та символу: *0 1 2 3 4 5 6 7 8 9 a b c d **.

4.2 Представити МТ у вигляді таблиці відповідності (рис. 4.59).

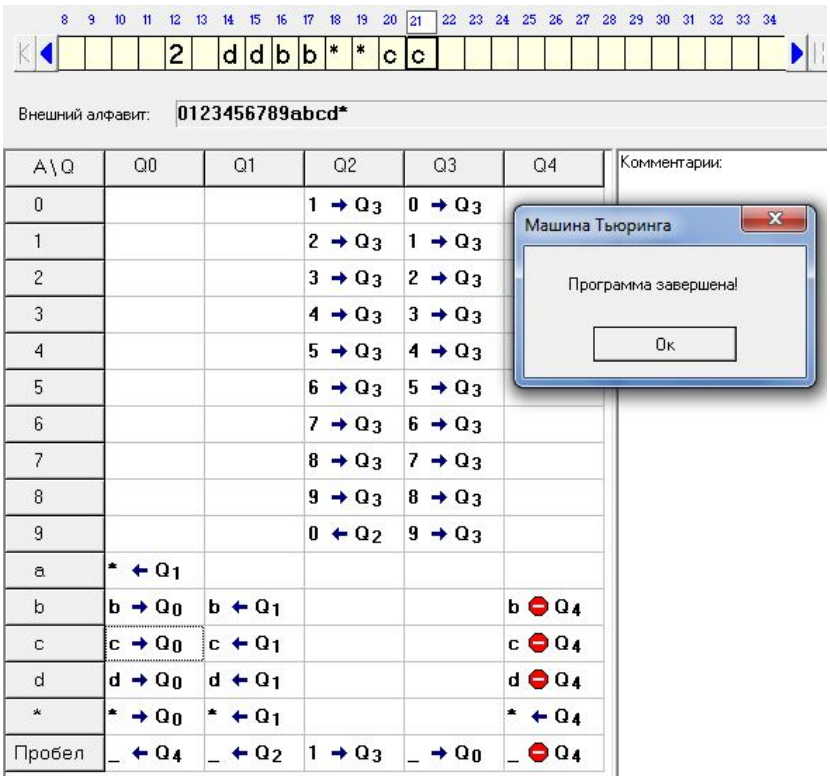


Внешний алфавит: 0123456789abcd*

A\Q	Q0	Q1	Q2	Q3	Q4
0			1 → Q3	0 → Q3	
1			2 → Q3	1 → Q3	
2			3 → Q3	2 → Q3	
3			4 → Q3	3 → Q3	
4			5 → Q3	4 → Q3	
5			6 → Q3	5 → Q3	
6			7 → Q3	6 → Q3	
7			8 → Q3	7 → Q3	
8			9 → Q3	8 → Q3	
9			0 ← Q2	9 → Q3	
a	* ← Q1				
b	b → Q0	b ← Q1			b ⛔ Q4
c	c → Q0	c ← Q1			c ⛔ Q4
d	d → Q0	d ← Q1			d ⛔ Q4
*	* → Q0	* ← Q1			* ← Q4
Пробел	_ ← Q4	_ ← Q2	1 → Q3	_ → Q0	_ ⛔ Q4

Рисунок 4.59 – Функціональна схема обробки введеного слова

4.3 Увести слово *ddbbaacc*. Результат обробки введеного слова наведено на рисунку 4.60.



Внешний алфавит: 0123456789abcd*

A\Q	Q0	Q1	Q2	Q3	Q4
0			1 → Q3	0 → Q3	
1			2 → Q3	1 → Q3	
2			3 → Q3	2 → Q3	
3			4 → Q3	3 → Q3	
4			5 → Q3	4 → Q3	
5			6 → Q3	5 → Q3	
6			7 → Q3	6 → Q3	
7			8 → Q3	7 → Q3	
8			9 → Q3	8 → Q3	
9			0 ← Q2	9 → Q3	
a	* ← Q1				
b	b → Q0	b ← Q1			b ⛔ Q4
c	c → Q0	c ← Q1			c ⛔ Q4
d	d → Q0	d ← Q1			d ⛔ Q4
*	* → Q0	* ← Q1			* ← Q4
Пробел	_ ← Q4	_ ← Q2	1 → Q3	_ → Q0	_ ⛔ Q4

Рисунок 4.60 – Результат обробки введеного слова

4.4 Увести інше слово та підтвердити працеспроможність програми.

Приклад 5

Розробити МТ, яка б могла опрацьовувати масив із *N* символів «*». Якщо введено *N > 5* символів, то буде виведено *N–2* символів. При введенні *N < 5* символів на інформаційній стрічці буде виведено *2N* символів, а при введенні *N = 5* символів результат буде дорівнювати одному символу.

5.1 Обрати вхідний алфавіт, який складається з символу *.

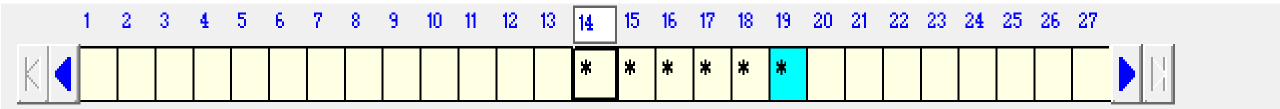
5.2 Представити МТ у вигляді таблиці відповідності (рис. 4.61).

A\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
*	* → Q1	* → Q2	* → Q3	* → Q4	* → Q14	* → Q9	* → Q10	_ → Q11	* → Q12		
Пробел		_ ← Q8	_ ← Q7	_ ← Q6	_ ← Q5					* → Q10	* → Q11

A\Q	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20
*				* → Q17	* ← Q16	_ ← Q16	* → Q17	_ ← Q19	_ ← Q20	* ⛔ Q20
Пробел	* → Q12	* → Q13	_ ⛔ Q13	_ ← Q15		_ ⛔ Q16	_ ← Q18			

Рисунок 4.61 – Програма для МТ

5.3 Увести на інформаційну стрічку 6 симолів «*», як це показано на рисунку 4.62.



Внешний алфавит: *

A\Q	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12
*	* → Q9	* → Q10	_ → Q11	* → Q12				
Пробел					* → Q10	* → Q11	* → Q12	* → Q13

Рисунок 4.62 – Введення початкових даних за першою умовою

У першому випадку згідно з програмою, якщо введено *N > 5* символів, то буде виведено *N–2* символу. Отриманий результат в чотири символи наведено на рисунку 4.63.

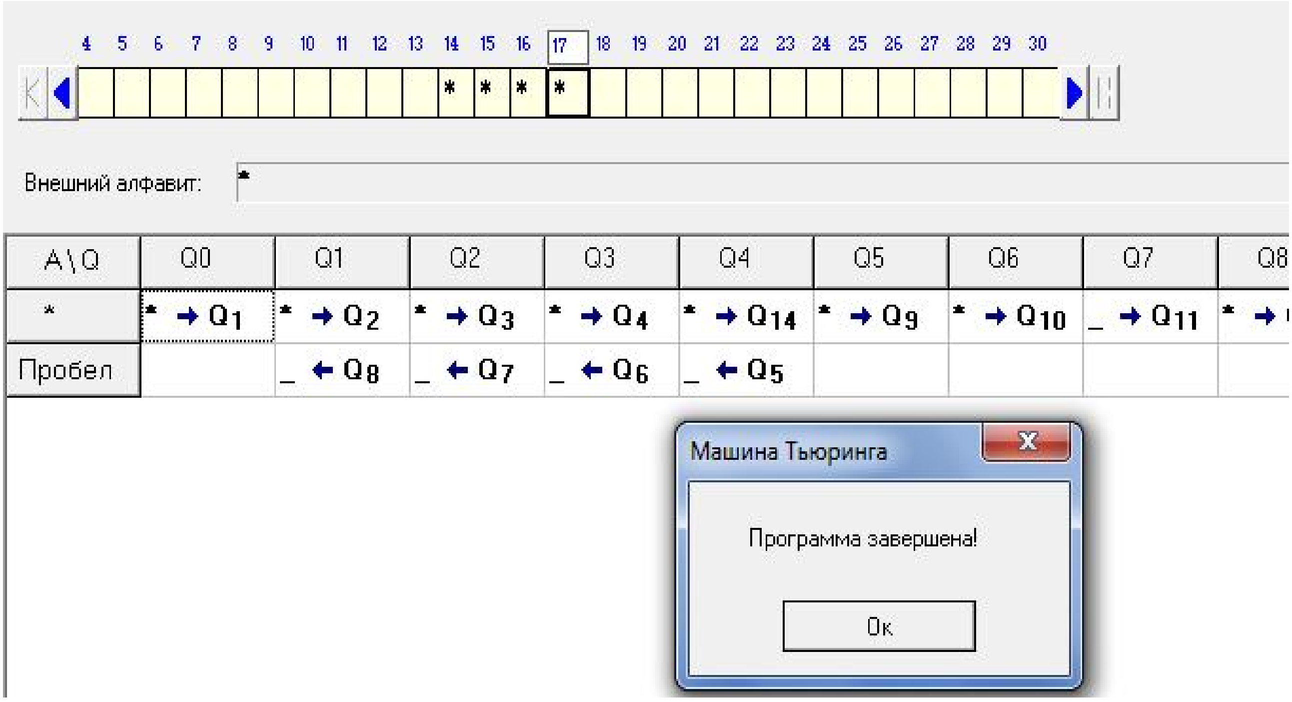


Рисунок 4.63 – Отримання результату за першою умовою

5.4 Увести на інформаційну стрічку три символи «*», як це показано на рисунку 4.64.

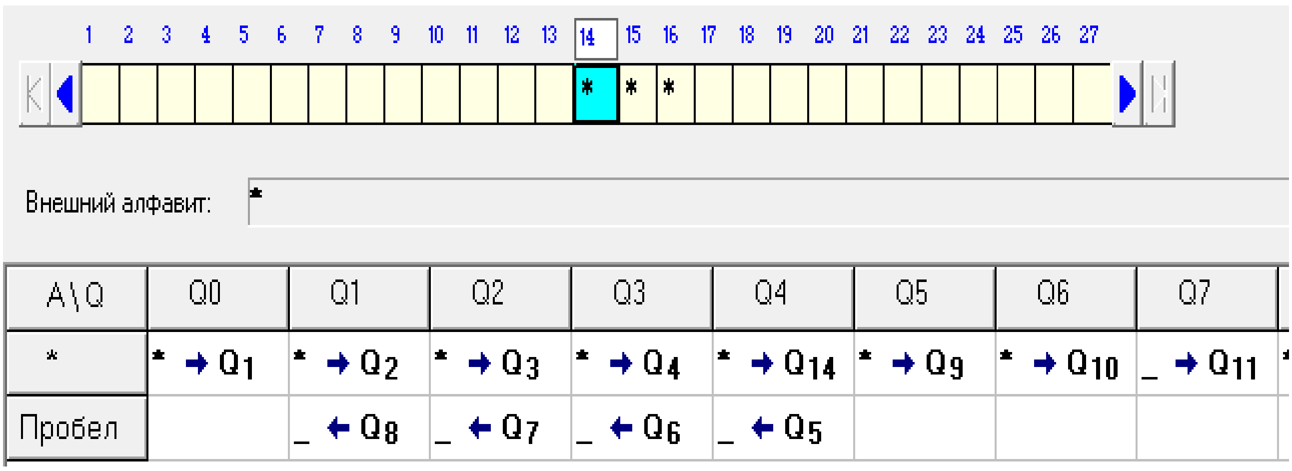


Рисунок 4.64 – Введення початкових даних для перевірки правильності виконання програми за другою умовою

У даному випадку згідно з програмою, якщо введено *N* <*5* символів, то кількість символів подвоюється на інформаційній стрічці.

Отриманий результат одержання *2N* символів наведено на рисунку 4.65.

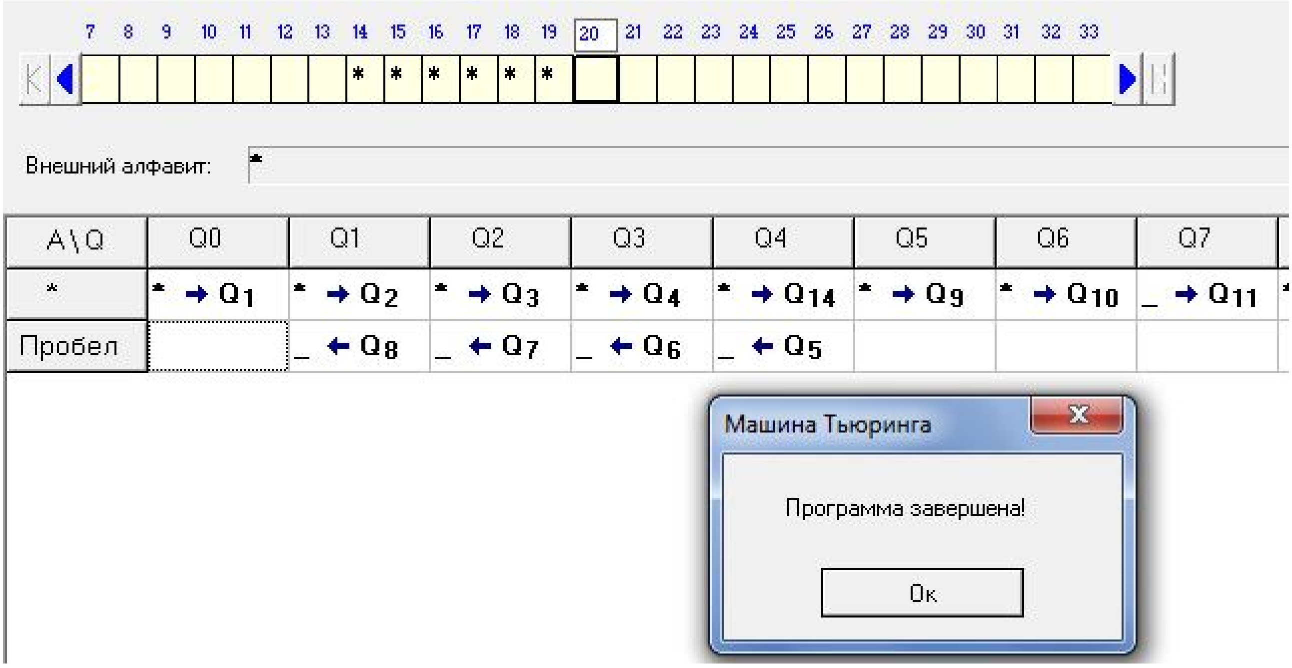


Рисунок 4.65 – Отримання подвоєного результату обробки масиву

5.5 Увести на інформаційну стрічку п’ять символів «*», як показано на рисунку 4.66.

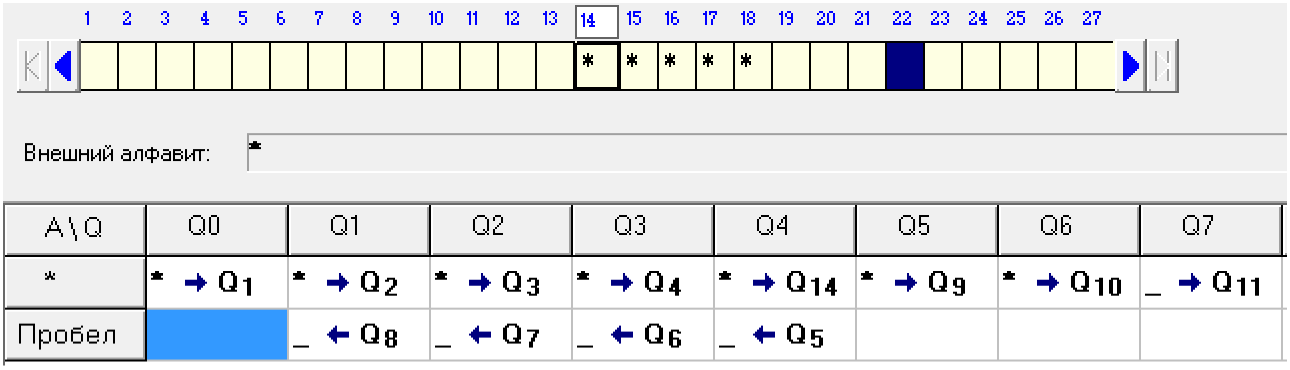


Рисунок 4.66 – Введення початкових даних для перевірки правильності виконання програми за третьою умовою

У результаті введення $N = 5$ символів результат буде дорівнювати одному символу, що й підтверджує отриманий результат, який наведено на рисунку 4.67.

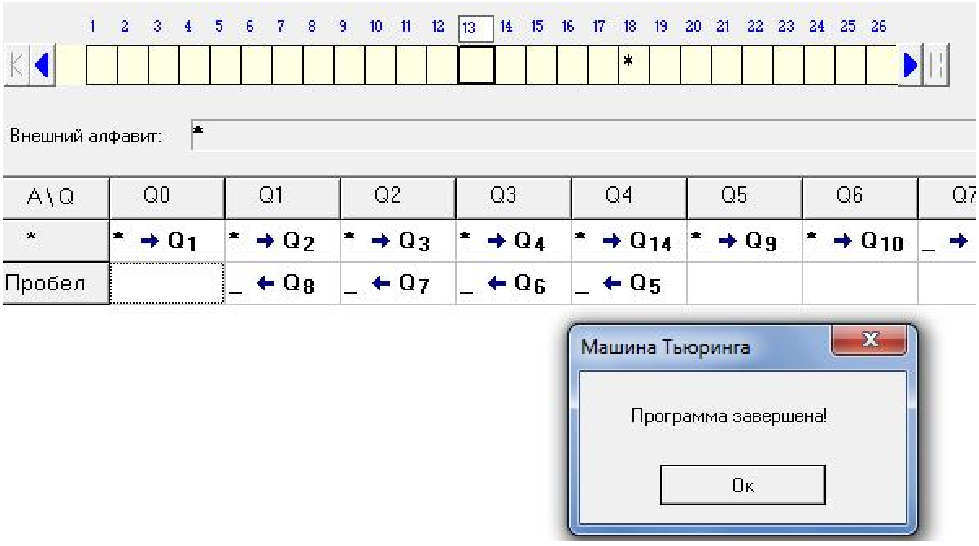


Рисунок 4.67 – Отриманий результат за введення *N* = *5* символів

5.6 Увести на інформаційну стрічку довільне число символів «*» та перевірити роботу програми.

Приклад 6

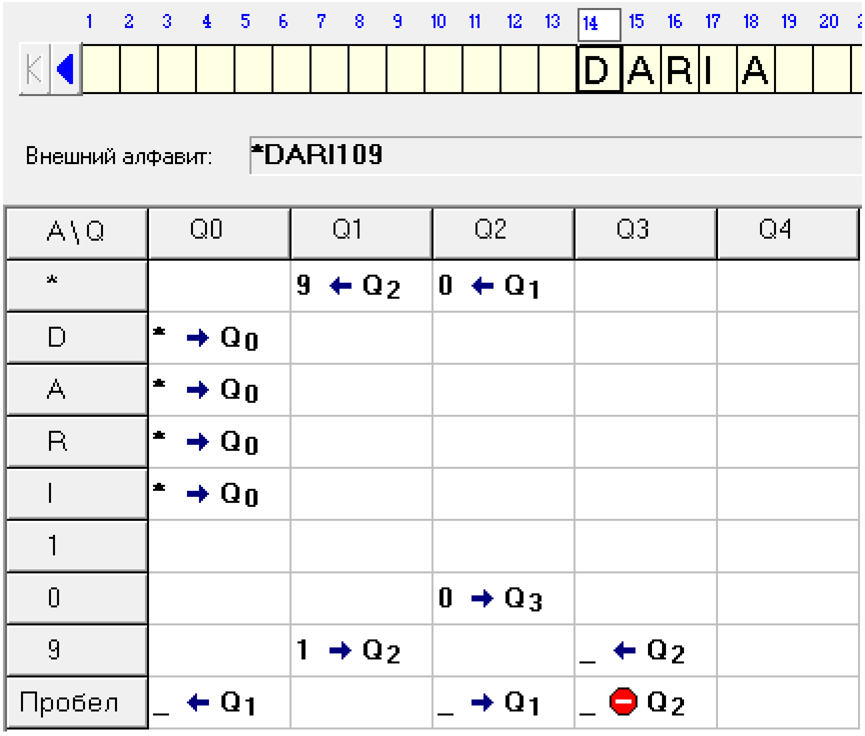
Розробити МТ, яка б, по-перше, замінювала усі літери введених імен на зірочки, по-друге, замість отриманих зірочок виводила певну комбінацію цифр: день та місяць народження.

6.1Обрати вхідний алфавіт, який складається з символів *, D, A, R, I, 1, 0, 9.

6.2 Представити МТ у вигляді таблиці відповідності (рис. 4.68).

6.3 Увести на інформаційну стрічку ім’я DARIA (рис. 4.68).

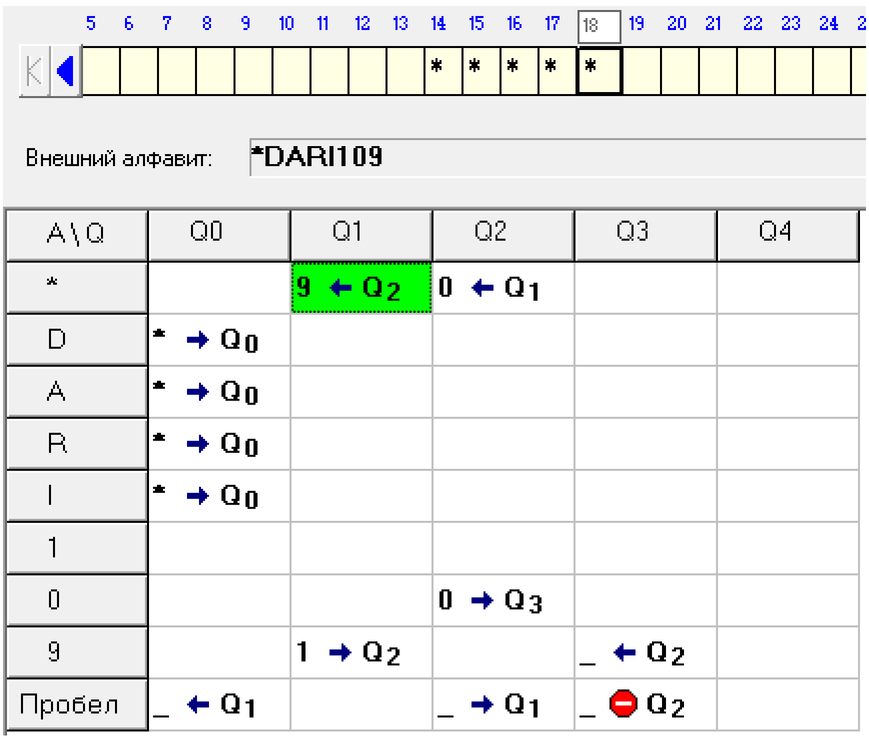
Каретка знаходиться на першій літері ім’я.



A\Q	Q0	Q1	Q2	Q3	Q4
*		9 ← Q2	0 ← Q1		
D	* → Q0				
A	* → Q0				
R	* → Q0				
I	* → Q0				
1					
0			0 → Q3		
9		1 → Q2		_ ← Q2	
Пробел	_ ← Q1		_ → Q1	_ Q2	

Рисунок 4.68 – Програма МТ для ім’я DARIA

6.3 Запустити програму на виконання. Проміжні результати роботи програми по заміні літер ім’я DARIA на символи «*» наведено на рисунку 4.69.



A\Q	Q0	Q1	Q2	Q3	Q4
*		9 ← Q2	0 ← Q1		
D	* → Q0				
A	* → Q0				
R	* → Q0				
I	* → Q0				
1					
0			0 → Q3		
9		1 → Q2		_ ← Q2	
Пробел	_ ← Q1		_ → Q1	_ Q2	

Рисунок 4.69 – Процес заміни літер на символи «*» для імені DARIA

Результат роботи програми по заміні проміжних символів «*» на цифри, що відповідають даті народження DARIA, наведено на рисунку 4.70.

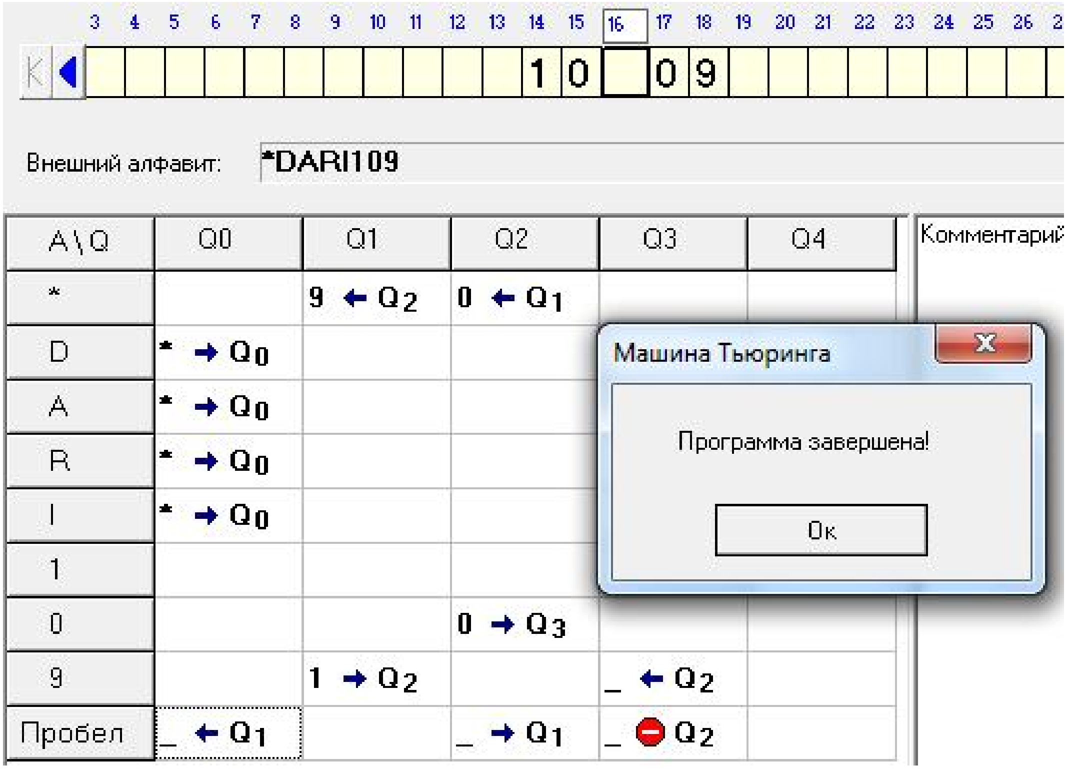


Рисунок 4.70 – Процес виводу дати народження для ім’я DARIA

6.4 Розробити МТ, яка аналогічним чином перетворює ім’я OLENA в цифри дня народження (рис. 4.71).

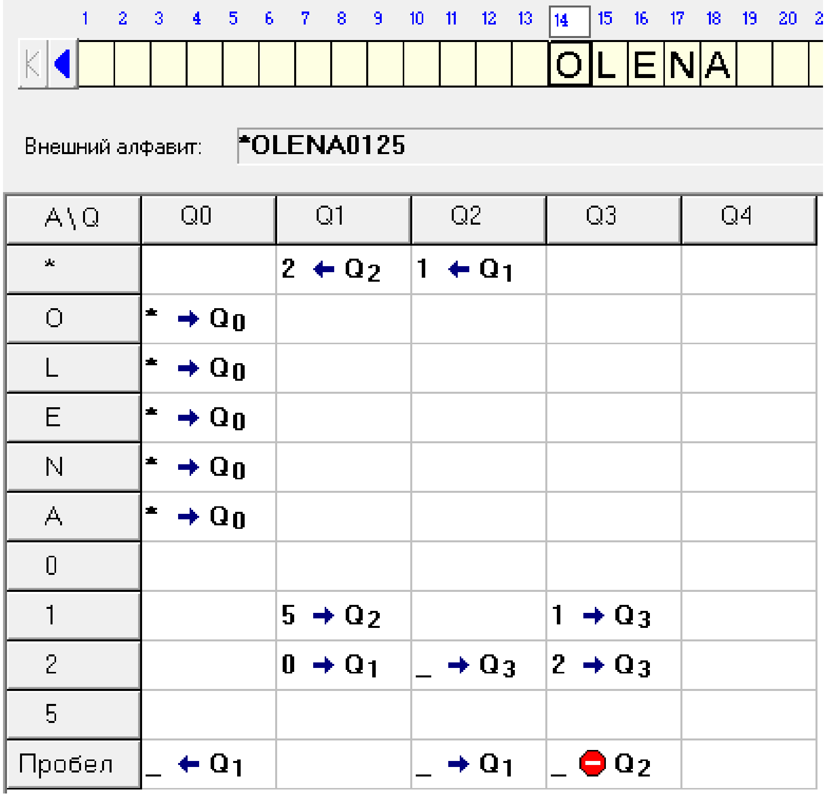


Рисунок 4.71 – Програма МТ для ім'я OLENA

Результати роботи МТ для ім'я OLENA наведено на рисунку 4.72

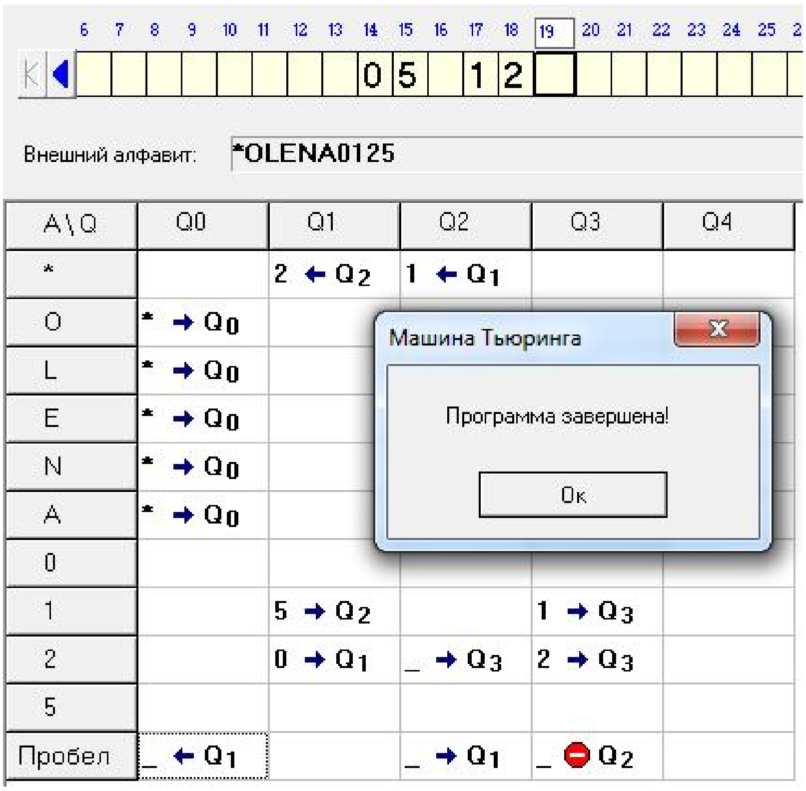


Рисунок 4.72 – Процес виводу дати народження для ім'я OLENA

6.5 Розробити МТ, яка перетворює ім’я ОЛЕНА в цифри дня народження та вітає з цим святом (рис. 4.73).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
													О	Л	Е	Н	А									

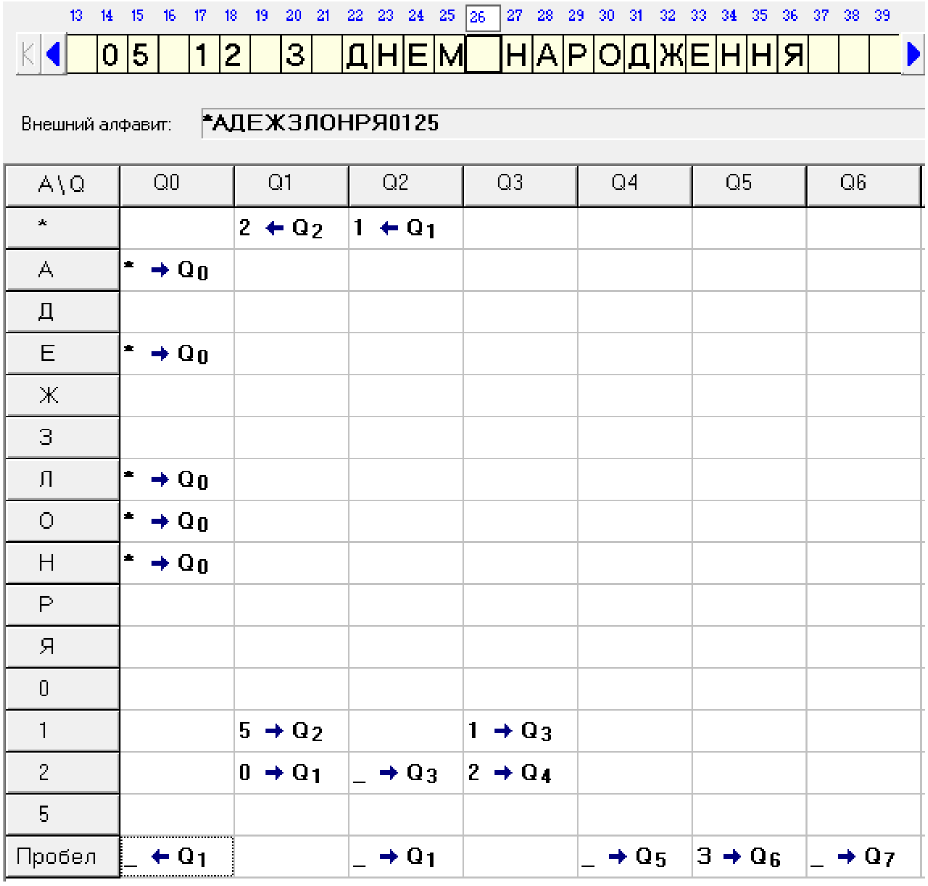
Внешний алфавит: *АДЕЖЗЛОНРЯ0125

А\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6
*		2 ← Q_2	1 ← Q_1				
А	* → Q_0						
Д							
Е	* → Q_0						
Ж							
З							
Л	* → Q_0						
О	* → Q_0						
Н	* → Q_0						
Р							
Я							
0							
1		5 → Q_2		1 → Q_3			
2		0 → Q_1	_ → Q_3	2 → Q_4			
5							
Пробел	_ ← Q_1		_ → Q_1		_ → Q_5	З → Q_6	_ → Q_7

Н → Q_9	Е → Q_{10}	М → Q_{11}	_ → Q_{12}	Н → Q_{13}	А → Q_{14}	Р → Q_{15}	
О → Q_{16}	Д → Q_{17}	Ж → Q_{18}	Е → Q_{19}	Н → Q_{20}	Н → Q_{21}	Я → Q_{22}	_ ⛔ Q_{22}

Рисунок 4.73 – Таблиця відповідності для вітальної МТ

Результати роботи МТ для привітання ОЛЕНИ з днем народження наведено на рисунку 4.74.



A\Q	Q0	Q1	Q2	Q3	Q4	Q5	Q6
*		2 ← Q_2	1 ← Q_1				
А	* → Q_0						
Д							
Е	* → Q_0						
Ж							
З							
Л	* → Q_0						
О	* → Q_0						
Н	* → Q_0						
Р							
Я							
0							
1		5 → Q_2		1 → Q_3			
2		0 → Q_1	_ → Q_3	2 → Q_4			
5							
Пробел	_ ← Q_1		_ → Q_1		_ → Q_5	3 → Q_6	_ → Q_7

Рисунок 4.74 – Результати роботи вітальної МТ

6.6 Розробити МТ, яка перетворює літери Вашого власного ім’я на Вашу дату народження, та підтвердити працеспроможність програми.

Контрольні питання

1. Дати визначення конфігурації МТ.
2. Які два кінцевих алфавіти пов’язані з МТ?
3. У повторенні якого циклу дій полягає МТ?
4. Чи існують такі дві команди, які б мали однакові перші два символи?
5. Що є результатом роботи МТ після її зупинення?
6. З якою метою в програмі, зображеній на рисунку 4.53, у зовнішньому алфавіті використовується символ «*»?
7. Пояснити яким чином літери ім’я DARIA замінюються на символи «*» (рис. 4.69).
8. Якими командами проміжні символи «*» замінюються на цифри дати народження.
9. Описати та пояснити призначення загальних команд для МТ, які перетворюють літери обох імен в цифри дат народження (рис. 4.68 та рис. 4.71).
10. В процесі роботи програми, які символи виводяться на інформаційну стрічку, після заміни символів «*» в іменах (рис. 4.70, 4.72)?

СПИСОК ДЖЕРЕЛ

1. Машины Тьюринга и вычислимые функции : пер. с нем. – М. : Мир, 1972. – 211 с.

2. Хопкрофт Джон. Введение в теорию автоматов, языков и вычислений (Introduction to Automata Theory, Languages, and Computation) / Джон Хопкрофт, Раджив Мотвани, Джеффри Ульман. – М. : Вильямс, 2002. – 528 с.

3. Зюзьков В. М. Математическая логика и теория алгоритмов : учебное пособие для вузов. – 2-е изд. / В. М. Зюзьков, А. А. Шелупанов – М. : Горячая линия – Телеком, 2007. – 176 с.

4. Трахтенброт Б. А. Алгоритмы и машинное решение задач / Б. А. Трахтенброт – М. : Государственное издательство технико-теоретической литературы, 1957. – 96 с.

5. Алферова З. А. Теория алгоритмов / З. А. Алферова – М. : Статистика, 1973. – 256 с.

6. Мальцев А. И. Алгоритмы и вычислимые функции / А. И. Мальцев – М. : Наука, 1986. – 178 с.

7. Марков А. А. Теория алгорифмов / А. А. Марков, Н. М. Нагорный – М. : Наука, 1984. – 213 с.

8. Программный иммитатор [Электронный ресурс]. – Режим доступа: http://fayloobmennik.cloud/1492236, свободный (дата обращения: 25.10.14) – Название с экрана.

9. Лавров И. А. Задачи по теории множеств, математической логике и теории алгоритмов / И. А. Лавров, Л. Л. Максимова. – 4-е изд. – М. : ФИЗМАТЛИТ, 2001. – 256 с.

10. Лебедев В. Б. Методические указания к практическим занятиям «Теория алгоритмов» / В. Б. Лебедев, Н. Е. Кирсанова – Пенза : Пензенская государственная технологическая академия, 2001. – 67 с.

11. Козлов К. П. Алгоритмы : учеб. пособие / К. П. Козлов – Ленинград : ЛГПИ, 1989. – 39 с.

12. Миков А. И. Информатика. Введение в компьютерные науки / А. И. Миков – Пермь : ПГУ, 1998. – 45 с.

13. Что такое машина Тьюринга [Электронный ресурс]. – Режим доступа: http://xbb.uz/IT/Chto-takoe-mashina-Tjuringa, свободный (дата обращения : 17.11.14) – Название с экрана.

14. Пильщиков В. Н. Машина Тьюринга и алгоритмы Маркова / В. Н. Пильщиков, В. Г. Абрамов, А. А. Вылиток – М. : МАКС Пресс, 2016. – 72 с.

15. Пенроуз Роджер. Новый ум короля: о компьютерах, мышлении и законах физики / Роджер Пенроуз – Киев : ЛКИ, 2015. – 402 с.

Навчальне видання

ПЕТРОВА Олена Олександрівна

ДОСЛІДЖЕННЯ РОБОТИ МАШИНИ ТЮРІНГА

НАВЧАЛЬНО-МЕТОДИЧНИЙ ПОСІБНИК

2-ге видання, перероблене і доповнене

Відповідальний за випуск *М. В. Новожилова*

Редактор *О. В. Михаленко*

Комп’ютерне верстання *О. О. Петрова*

Дизайн обкладинки *Т. А. Лазуренко*

Підп. до друку 25.05.2020. Формат 60 х 84/16
Друк на ризографі. Ум. друк. арк. 4,4.
Тираж 50 прим. Зам. №

Видавець і виготовлювач:
Харківський національний університет
міського господарства імені О. М. Бекетова,
вул. Маршала Бажанова, 17, Харків, 61002.
Електронний адрес: rectorat@kname.edu.ua
Свідоцтво суб’єкта видавничої справи:
ДК № 5328 від 11.04.2017.