Машина на Тюринг

*Изготвил:* Георги Станчев, II курс, спец. ИС, фак. № 71959

[СЪДЪРЖАНИЕ 2](#_Toc60916201)

[Глава 1. Увод 3](#_Toc60916202)

[1.1. Описание и идея на проекта 3](#_Toc60916203)

[1.2. Цел и задачи на разработка 3](#_Toc60916204)

[1.3. Структура на документацията 4](#_Toc60916205)

[Глава 2. Преглед на предметната област 4](#_Toc60916206)

[2.1. Основни дефиниции, концепции и алгоритми, които ще бъдат използвани 4](#_Toc60916208)

[2.2. Дефиниране на проблеми и сложност на поставената задача 4](#_Toc60916209)

[2.3. Подходи, методи (евентуално модели и стандарти) за решаване на поставените проблемите 5](#_Toc60916210)

[2.4. Потребителски (функционални) изисквания (права, роли, статуси, диаграми, ...) и качествени (нефункционални) изисквания (скалируемост, поддръжка, ...) 5](#_Toc60916211)

[Глава 3. Проектиране 6](#_Toc60916212)

[3.1. Обща архитектура – ООП дизайн 6](#_Toc60916214)

[Глава 4. Реализация, тестване 7](#_Toc60916215)

[4.1. Реализация на класове (включва важни моменти от реализацията на класовете и малки фрагменти от кода) 7](#_Toc60916217)

[4.2. Управление на паметта и алгоритми. Оптимизации 8](#_Toc60916218)

[4.3. Планиране, описание и създаване на тестови сценарии (създаване на примери) 8](#_Toc60916219)

[Глава 5. Заключение 16](#_Toc60916220)

[5.1. Обобщение на изпълнението на началните цели 16](#_Toc60916222)

[5.2. Насоки за бъдещо развитие и усъвършенстване 16](#_Toc60916223)

[Глава 6. Използвана литература 16](#_Toc60916224)

# Увод

* 1. Описание и идея на проекта

Да се реализира библиотека за ефективна работа с машини на Тюринг. Машините на Тюринг имат потенциално безкрайна двупосочна лента, в която могат да бъдат записвани произволни символи.

Преход в Машината на Тюринг се задава със следния синтаксис:

* <състояние> ::= **{** <низ> **}**
* <команда> ::= **L** | **R** | **S**
* <преход> ::= <символ><състояние> **->** <символ><състояние><команда>

Командите управляват движението на главата, съответно наляво (L), надясно (R), или без промяна (S). Специалното състояние **{halt}** спира изпълнението на машината.

Пример: 6{increment} -> 7{decrement}L.

Интерпретацията на примера е следната: ако на лентата (под текущото положение на главата) е записан символа 6 и машината е в състояние {increment}, да се замени 6 със 7, да се премине в състояние {decrement}, и да се придвижи лентата наляво.

Машината на Тюринг да може да се конструира от множество преходи, подадени като вход на програмата, като автоматично се извлекат състоянията на машината, командите и преходите. Да се поддържа възможност да се извърши изпълнение на машината над някаква лента, получена като вход. Като резултат да се изведе състоянието на лентата и позицията на главата при достигане до състояние **{halt}.**

Внимание: тъй като лентата е безкрайна, да се изведе само тази част от нея, която е претърпяла промени по време на изпълнението на машината.

* 1. Цел и задачи на разработка

Да се реализира програма, която прочита от входен файл информация за машина на Тюринг и състояние на лентата, след което изпълнява машината на Тюринг върху така зададената лента. Ако машината завърши, да се изведе в изходен файл състоянието на изходната лента.

Да се поддържат следните операции над машини на Тюринг (*Таблица 1*):

*Таблица 1*

|  |  |
| --- | --- |
| compose | Композиция на две машини на Тюринг. |
| branch | Разклонение на две машини на Тюринг относно трета (пуска се третата машина над даден вход и в зависимост дали крайният резултат е 1 (истина) или 0 (лъжа), се пуска първата или втората машина над същия вход). |
| combine | машина на Тюринг, реализираща while-цикъл над дадена машина на Тюринг, където условието се задава с друга машина на Тюринг. |

* 1. Структура на документацията

Документа е структуриран в 6 (шест) основни глави, описващи целта и идеята на проекта, функционални изисквания, проектиране, разработка и тестване, заключение.

# Преглед на предметната област

1. 1. Основни дефиниции, концепции и алгоритми, които ще бъдат използвани

Програмата е реализирана на Visual Studio 2019, C++. Основната идея, по която е разработена, е симулация на концептуалната работа на известната машина на Тюринг. Както самото абстрактно изчислително устройство, разполагаме с глава, състояния, преходи и ленти. Главата представлява малко устройство, което винаги започва да се движи от началото на лентата и в зависимост от изпълняваната инструкция (преход) и прочетения символ, то записва нов символ и се премества наляво или надясно по лентата, като може и да остане на място. Следователно, по-просто казано, главата може да се движи двупосочно и да чете, изтрива и печата нови символи. Освен това в главата са записани и преходите, които могат да бъдат изпълнявани по време на обработката на потенциално безкрайна лента и началното състояние на машината. Трябва и да се отбележи, че машината прекратява своята работа самостоятелно при достигане на крайното състояние {halt}, а при всеки друг вариант тя се прекратява поради липсата на подходящ преход или встъпване в безкраен цикъл.

* 1. Дефиниране на проблеми и сложност на поставената задача

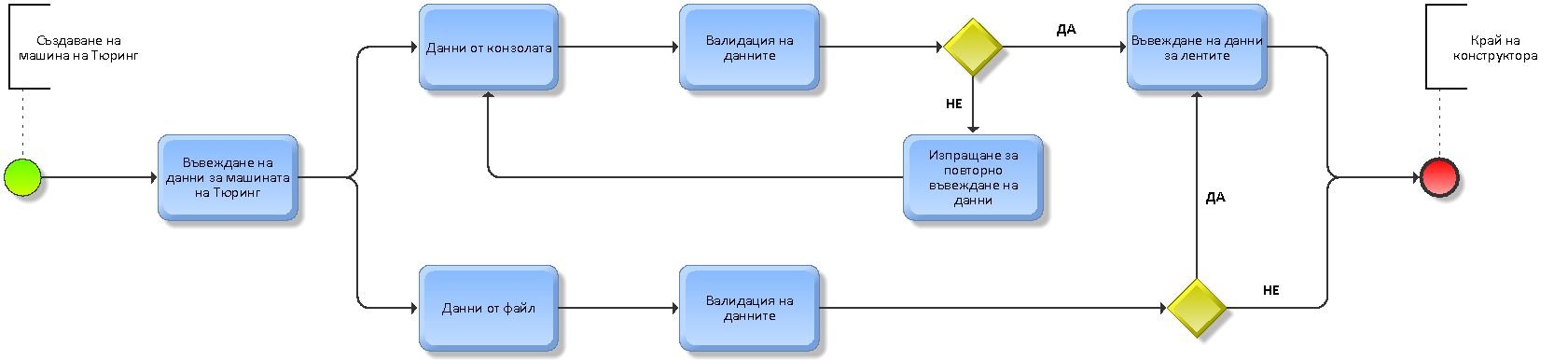
В самото начало постигането на целите на задачата изглеждаше сложно, но в последствие на писане на код и стъпка по стъпка, всичко започна да се подрежда логически. Трудността се състоеше в това, че беше необходимо да се обмисли внимателно по какъв начин главата ще се движи по лентата и как ще се определя позицията й, тъй като лентата е потенциално безкрайна. Момент на затруднение беше и въпросът дали може да се избегне един от вариантите на изход на машината при обработка на една лента, а именно встъпване на машината в безкраен цикъл.

* 1. Подходи, методи (евентуално модели и стандарти) за решаване на поставените проблемите

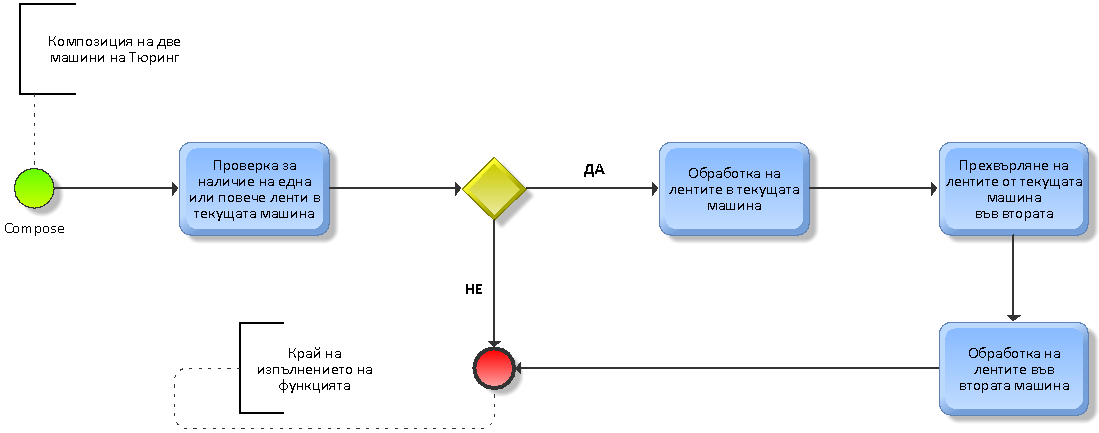
След задълбочен анализ върху поставената задача, подходи и методи за реализираното и постигането на крайните цели и обстойно запознаване с подробно разгледаната тема за машина на Тюринг в книгата „Математическа лингвистика“ на Радослав Павлов, идеите за намиране на решение на поставените проблеми бяха налице. Тъй като лентата е потенциално безкрайна, в началото и в края винаги ще стои по една празна клетка и дори при запълването й от символ, отново ще се допълва. По този начин началната позиция на главата ще започва от 1, като 0 ще се счита за празната клетка в началото на лентата. Същото се отнася и за края на лентата – това ще бъде предпоследната позиция, като крайната позиция ще бъде празна клетка. А доколкото се отнася до въпросът за по-рано споменатият специфичен изход на машината, намерено е прилично решение при встъпване на устройството в безкраен цикъл.

* 1. Потребителски (функционални) изисквания (права, роли, статуси, диаграми, ...) и качествени (нефункционални) изисквания (скалируемост, поддръжка, ...)

Потребителските (функционални) изисквания към приложението (*Фигура 1*) се дефинират и детайлизират с цел постигане на високо ниво на разбиране за потребностите на потребителя от функционалността на приложението и дават ясна представа за отделните компоненти, които да се разработят по подходящ метод и алгоритъм.

*Фигура 1*

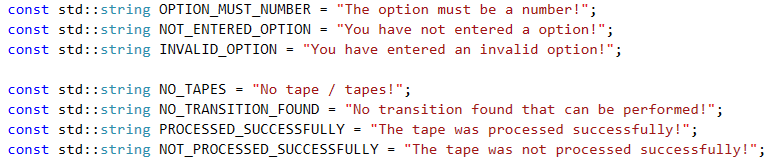
На следващата схема (*Фигура 2*) е показана детайлно функционалността на операция „compose”, която представлява композиция на две машини на Тюринг.

*Фигура 2*

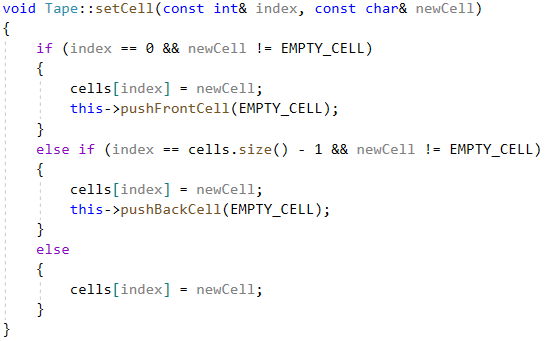
# Проектиране

1. 1. Обща архитектура – ООП дизайн

Архитектурата на програмата е внимателно моделирана, вземайки се предвид и най-малките детайли. Създадени са нужните класове за поддържането на функционалността, а изгледа е достатъчно улеснен с цел подпомагане на потребителя как да използва програмата правилно и лесно. Предвидени са и всички случаи, в които потребителя може да сгреши или неволно, ако обърка нещо, за да може ако това се случи, програмата да му изведе подходящо съобщение и от съображение на удобство, не се почва отначало въвеждането, а само трябва да въведе компонента, в който е допуснал грешката. (*Фрагмент 1*)

*Фрагмент 1*

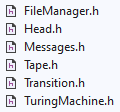
Що се отнася до работата на машината, имаме три основни елемента – глава, ленти и преходи. Всички необходими операции се случват в самата машина автоматизирано и без излишни действия. Позицията на главата след всеки изпълнен преход се изчислява, разбира се, ако командата е да се премести наляво или надясно, а щом стигне до началото на лентата, тоест празна клетка в този край, тя никога няма да премине границата от 0. Друго е позиционирането на главата, ако се разглежда случая, в който върви към края на лентата – тогава позицията може да се увеличава докато машината не спре да работи по някакъв начин. Следователно винаги щом главата се намира над някоя от двете празни клетки, в началото или края, то ще бъде допълнена нова, където е необходимо, а позицията ще бъде правилно пресметната. (*Фрагмент 2*)

*Фрагмент 2*

# Реализация, тестване

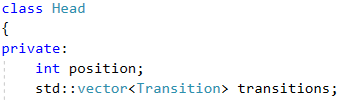
1. 1. Реализация на класове (включва важни моменти от реализацията на класовете и малки фрагменти от кода)

Създадените класове, както се подразбира от техните имена, са имплементации на компонентите, които задължително една машина на Тюринг трябва да има, а именно глава, преходи и лента. Разбира се, самият клас „Машина на Тюринг“ се грижи за построяването на машината и за нейното правилно функциониране. Останалите класове – съобщения и файлов мениджър, благодарение на тях не трябва да има повод за притеснения от възникване на грешка по време на работа с файлове, а дори и да възникне, ще бъде предоставено подходящо съобщение. Включени са и необходимите съобщения, които трябва да се изкарват при изпълнение или дори още по време на построяването, тъй като може да се подаде неточна информация, независимо дали от файл или от конзолата. (*Фрагмент 3*)



*Фрагмент 3*

Това е малък фрагмент от класа Глава, представящ член данните му. Използвам последователният контейнер вектор, за да може в него да бъдат запазени преходите (инструкциите) на машината, които се създават още в началото на построяването й, като трябва да се отбележи и тяхната належаща необходимост и число, което се счита за позицията на символа, над който се намира главата и е от съществено значение за правилното функциониране на машината. (*Фрагмент 4*)



*Фрагмент 4*

* 1. Управление на паметта и алгоритми. Оптимизации

Кодът е писан максимално съобразен със заемането на колкото се може по-малко памет и извършването на само необходимият брой операции, необходими за реализирането на функционалните изисквания, с цел да се доведе до минимум времето на изпълнение на приложението.

* 1. Планиране, описание и създаване на тестови сценарии (създаване на примери)

За постигане на успешната работа с машина на Тюринг е необходимо много добре да се планират стъпките и дейностите в етапите на тестване и внедряване. Крайният сорс код ще се окомплектова заедно с помощните материали и публикува в GitHub, като същевременно е спазвано и качването на междинните разработки.

За преглед - [*https://github.com/FantomGSS/TuringMachine*](https://github.com/FantomGSS/TuringMachine)

С цел проверка на постигнатата реализация на приложението с необходимите критерии по задание, е необходимо да се изпълняват минимум следните функционални тестови сценарии (*Таблица 2*) и следователно още по времето на създаване на всяка една функционалност са направени нужните тестове, за да се потвърди тяхното правилно изпълнение. Не винаги се получаваше от първия път, но по този начин излизаха и още нови идеи, по които по-лесно и оптимално да се направи задачата.

*Таблица 2*

|  |  |
| --- | --- |
| **Функция** | **Резултат (да/не)** |
| compose | да |
| branch | да |
| combine | не |

По-долу (*Таблица 3*) са представени допълнителни изисквания, които се считат като бонуси за проекта. Реализиран е само първият от тях и поради това се наложи преработка на по-голямата част от кода (рефакторинг / refactoring), за да може и такъв тип машини да се възползват от останалите функциите. Показани са и няколко примера от работата на приложението, покривайки основните функционалности, които се изискваха от задачата. Входа от потребителя е означен с долар ($) в примерите.

*Таблица 3*

|  |  |
| --- | --- |
| **Бонуси** | |
| 1. Да се реализира многолентова машина на Тюринг и преобразуването ѝ до еднолентова. | |
| **Осъществена реализация на многолентова машина на Тюринг.** | |
| **Функция** | **Резултат (да/не)** |
| toSingleTape | да |
| 2. Да се реализира недетерминирана машина на Тюринг и преобразуването ѝ до детерминирана. | |
| **Неосъществена реализация на недетерминирана машина на Тюринг.** | |
| **Функция** | **Резултат (да/не)** |
| determine | не |

Пример 1:

How would you like to provide information about the Turing Machine? The options are:

1. From file, providing the full path to the file.

2. From console, entering the state of the machine and the transitions.

Option: $ 1

Enter the full path to the file: $ FirstHead.txt

===========================================================

Successfully opened FirstHead.txt!

===========================================================

How would you like to provide the tape(s) information? The options are:

1. From file, providing the full path to the file.

2. From console, entering one or more tapes.

Option: $ 1

Enter the full path to the file: $ FirstTapes.txt

===========================================================

Successfully opened FirstTapes.txt!

===========================================================

How would you like to review the results? Save the information to a file or display it on the screen?

1. In a file, providing the full path to the file.

2. In the console

Option: $ 2

\_111000101010\_ The tape was processed successfully! 13

\_000011110000\_ The tape was processed successfully! 13

Пример 2:

How would you like to provide information about the Turing Machine? The options are:

1. From file, providing the full path to the file.

2. From console, entering the state of the machine and the transitions.

Option: $ 1

Enter the full path to the file: $ FirstHead.txt

===========================================================

Successfully opened FirstHead.txt!

===========================================================

How would you like to provide the tape(s) information? The options are:

1. From file, providing the full path to the file.

2. From console, entering one or more tapes.

Option: $ 1

Enter the full path to the file: $ FirstTapes.txt

===========================================================

Successfully opened FirstTapes.txt!

===========================================================

How would you like to provide information about the Turing Machine? The options are:

1. From file, providing the full path to the file.

2. From console, entering the state of the machine and the transitions.

Option: $ 2

Enter the initial state of the machine: $ A

Each transition must be entered on a new line, strictly following its format.

To stop entering transitions, you must enter stop!

$ 1{A} -> 0{A}R

$ 0{A} -> 2{B}L

$ 0{B} -> 2{B}L

$ 6{A} -> 2{B}L

$ \_{B} -> 4{C}R

$ 2{C} -> 5{C}L

$ 4{C} -> E{halt}S

$ stop

How would you like to provide the tape(s) information? The options are:

1. From file, providing the full path to the file.

2. From console, entering one or more tapes.

Option: $ 2

You must enter at least one tape!

To stop entering tapes, you must enter stop!

$ 116

$ 111

$ stop

Regarding the results of the first machine:

How would you like to review the results? Save the information to a file or display it on the screen?

1. In a file, providing the full path to the file.

2. In the console

Option: $ 2

\_111000101010\_ The tape was processed successfully! Head position: 13

\_000011110000\_ The tape was processed successfully! Head position: 13

Regarding the final results of both machines:

How would you like to review the results? Save the information to a file or display it on the screen?

1. In a file, providing the full path to the file.

2. In the console

Option: $ 2

\_E522\_ The tape was processed successfully! Head position: 1

\_000\_ The tape was not processed successfully! Head position: 4

\_E522200101010\_ The tape was processed successfully! Head position: 1

\_E500011110000\_ The tape was processed successfully! Head position: 1

Пример 3:

How would you like to provide information about the Turing Machine? The options are:

1. From file, providing the full path to the file.

2. From console, entering the state of the machine and the transitions.

Option: $ 1

Enter the full path to the file: $ FirstHead.txt

===========================================================

Successfully opened FirstHead.txt!

===========================================================

How would you like to provide the tape(s) information? The options are:

1. From file, providing the full path to the file.

2. From console, entering one or more tapes.

Option: $ 1

Enter the full path to the file: $ FirstTapes.txt

===========================================================

Successfully opened FirstTapes.txt!

===========================================================

How would you like to provide information about the Turing Machine? The options are:

1. From file, providing the full path to the file.

2. From console, entering the state of the machine and the transitions.

Option: $ 2

Enter the initial state of the machine: $ A

Each transition must be entered on a new line, strictly following its format.

To stop entering transitions, you must enter stop!

$ 1{A} -> 0{A}R

$ 0{A} -> 2{B}L

$ 0{B} -> 2{B}L

$ 6{A} -> 2{B}L

$ \_{B} -> 4{C}R

$ 2{C} -> 5{C}L

$ 4{C} -> E{halt}S

$ stop

How would you like to provide the tape(s) information? The options are:

1. From file, providing the full path to the file.

2. From console, entering one or more tapes.

Option: $ 2

You must enter at least one tape!

To stop entering tapes, you must enter stop!

$ 116

$ 111

$ stop

How would you like to provide information about the Turing Machine? The options are:

1. From file, providing the full path to the file.

2. From console, entering the state of the machine and the transitions.

Option: $ 1

Enter the full path to the file: $ ThirdHead.txt

===========================================================

Successfully opened ThirdHead.txt!

===========================================================

How would you like to provide the tape(s) information? The options are:

1. From file, providing the full path to the file.

2. From console, entering one or more tapes.

Option: $ 2

You must enter at least one tape!

To stop entering tapes, you must enter stop!

$ stop

Regarding the results of the first machine:

How would you like to review the results? Save the information to a file or display it on the screen?

1. In a file, providing the full path to the file.

2. In the console

Option: $ 2

\_111000101010\_ The tape was processed successfully! Head position: 13

\_000011110000\_ The tape was processed successfully! Head position: 13

The result of the machine operation is positive. Regarding the result of the machine, which have to continue depending on the result of the first machine:

How would you like to review the results? Save the information to a file or display it on the screen?

1. In a file, providing the full path to the file.

2. In the console

Option: $ 2

\_E522\_ The tape was processed successfully! Head position: 1

\_000\_ The tape was not processed successfully! Head position: 4

\_E500111010101\_ The tape was processed successfully! Head position: 1

\_E522220001111\_ The tape was processed successfully! Head position: 1

# Заключение

1. 1. Обобщение на изпълнението на началните цели

Като цяло, проекта беше интересен. На пръв поглед, задачата изглеждаше трудна, но впоследствие след като направих анализ и си изградих представа за машината, всичко почна да изглежда с една идея по-лесно. Трябваше да се запозная първо обстойно с работата на това устройство, за да мога да създам впоследствие правилната симулация. Лично аз съм доволен от своята работа, тъй като имах доста идеи и успях да реализирам най-добрите, постигайки стабилна работа и коректно изпълняване на поставените цели.

* 1. Насоки за бъдещо развитие и усъвършенстване

Като възможно усъвършенстване на приложението в бъдеще, може да се реализира недетерминирана машина на Тюринг и преобразуването ѝ до детерминирана.

# Използвана литература

*Използван продукт за схеми по фиг.1 и 2 – ARIS Express* [*https://www.ariscommunity.com/aris-express*](https://www.ariscommunity.com/aris-express)