**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ  
імені Ігоря Сікорського»**

**Факультет прикладної математики**

**Кафедра прикладної математики**

«До захисту допущено»

Завідувач кафедри

\_\_\_\_\_\_\_\_\_\_\_\_\_ О. Р. Чертов

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 р.

**Дипломна робота**

**на здобуття ступеня бакалавра**

з напряму підготовки 6.040301 «Прикладна математика»

на тему: «Автоматизація генерації емоційно забарвлених реплік»

Виконав: студент IV курсу, групи КМ-61

|  |  |  |
| --- | --- | --- |
|  | Мозговий Микита Андрійович | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Керівник | асистент  Громова Вікторія Вікторівна | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Консультант зі спеціальних питань | асистент  Борисенко П. Б. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Консультант із нормоконтролю | старший викладач Мальчиков В. В. | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| Рецензент |  | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

Засвідчую, що в цій дипломній роботі немає запозичень із праць інших авторів без відповідних посилань.

Студент \_\_\_\_\_\_\_\_\_\_\_\_

Київ — 2020**Національний технічний університет україни**

**«Київський політехнічний інститут  
імені Ігоря Сікорського»**

Факультет прикладної математики

Кафедра прикладної математики

Рівень вищої освіти — перший (бакалаврський)

Напрям підготовки 6.040301 «Прикладна математика»

ЗАТВЕРДЖУЮ

Завідувач кафедри

\_\_\_\_\_\_\_\_\_\_\_\_\_ О. Р. Чертов

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 р.

**ЗАВДАННЯ**

**на дипломну роботу студенту**

Мозговому Микиті Андрійовичу

1. Тема роботи: «Автоматизація генерації емоційно забарвлених реплік»,

керівник роботи Громова Вікторія Вікторівна, асистент,

затверджені наказом по університету від .

2. Термін подання студентом роботи:

3. Вихідні дані до роботи: розроблена система повинна мати можливість масштабування на інші задачі генерації.

4. Зміст роботи: виконати аналіз існуючих методів розв’язання задачі, вибрати метод генерації тексту, спроектувати автоматизовану систему генерації емоційно забарвлених реплік, здійснити програмну реалізацію розробленої системи, провести тестування розробленої системи.

5. Перелік ілюстративного матеріалу: архітектурні графи ланцюгів Маркова, блок-схеми розроблених алгоритмів, схема взаємодії модулів системи, знімки екранних.

6. Консультанти розділів роботи:

|  |  |  |  |
| --- | --- | --- | --- |
| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
| завдання видав | завдання прийняв |
| Розділ 4. Програмне забезпечення | Борисенко П. Б., асистент |  |  |

7. Дата видачі завдання:

АНОТАЦIЯ

Дипломну роботу присвячено розробці математичних та програмних

засобів для розв’язання задачі генерації емоційнозабарвлених реплік.

У роботі проведено аналіз існуючих рішень указаної задачі, виконано їх порівняння з погляду використання ресурсів та адаптації до різних умов генерації.

Для розв’язання задачі в роботі використано метод ланцюгів Маркова. Користувач описує схему ланцюга Маркова, де станом може бути як текст, так і інший ланцюг Маркова. Розроблено автоматизовану систему, що реалізує обраний метод. У ході її тестування було виявлено, що … .

Роботу виконано на … аркушах, вона містить … додатки та перелік

посилань на використані джерела з … найменувань. У роботі наведено …

рисунків та … таблиць.

Ключові слова: емоція, генерація тексту, ланцюги Маркова

ABSTRACT

This thesis deals with developing of the mathematical and software tools

for solving the problem of generating emotional text.

In the thesis, the comparative analysis of existing solutions in terms of the resource consumption and adaptability to generating different kinds of texts.

Markov chain approach is used to solve the task. User creates scheme for Markov chain, in which each node represents either statement, or another Markov chain.

The automated system implementing the chosen method is developed. During testing the system, it was found that the system … .

The thesis is presented in … pages. It contains … appendixes and bibliography of … references. … figures and … tables are given in the thesis.

Keywords: emotion, text generating, Markov chain

Зміст

[Вступ 7](#_Toc40555994)

[1 Постановка задачі 8](#_Toc40555995)

[2 Аналіз існуючих методів 9](#_Toc40555996)

[2.1 Математичні методи 9](#_Toc40555997)

[2.1.1 Лінгвістичні моделі 9](#_Toc40555998)

[2.1.2 Моделювання з блоків 10](#_Toc40555999)

[2.1.3 Використання міток або тегів 10](#_Toc40556000)

[2.1.4 Генерація з використанням ієрархій 11](#_Toc40556001)

[2.1.5 Марківські моделі 12](#_Toc40556002)

[2.2 Висновки 13](#_Toc40556003)

[3. Математичне забезпечення 14](#_Toc40556004)

[3.1 Принцип моделювання ланцюгами Маркова 14](#_Toc40556005)

[3.2 Формування моделі 17](#_Toc40556006)

[3.3 Висновки 18](#_Toc40556007)

[4. Програмне забезпечення 19](#_Toc40556008)

[4.1 Архітектура програми 19](#_Toc40556009)

[4.2 Опис розроблених алгоритмiв 20](#_Toc40556010)

[4.3 Формат вхiдних даних 24](#_Toc40556011)

[4.3.1 Формат схеми ланцюга 24](#_Toc40556012)

[4.3.2 Форма файлу лінкування 24](#_Toc40556013)

[4.4 Формат результуючих даних 24](#_Toc40556014)

[Перелiк посилань 25](#_Toc40556015)

# Вступ

У наш час попиту набувають системи обробки великої кількості даних. Разом із тим, все більше уваги приділяється суміжній сфері генерації даних за певним критерієм.

Ця сфера допомагає нам не лише генерувати реалістичні текстові приклади, а й створювати нову корисну інформацію. Інформаційні системи, що здатні інтерполювати велику кількість інформації здатні змінити наш підхід до перегляду фільмів, пошукових запитів і генерації нової інформації. Зараз більша кількість інформації генерується людиною, що звужує можливість аналізу і збільшує можливість повторення. Ця проблема стосується як малювання, яке із приходом зручних засобів навчання, створення та розповсюдження контенту, змінює фокус у напрям сюжетів у малюнках, так і науки, яка зараз вирішує проблему псевдо позитивних результатів, яка пов’язана не тільки з великою кількістю досліджень, але і з не публікацією усіх досліджень та стандартним критерієм p<.05. Можливість генерації та аналізу згенерованих даних дозволяє нам знайти можливі шляхи за якими пройшла генерація та по новому подивитися на старі критерії.

Вже зараз ми маємо можливість прослухати згенеровану музику, або власноруч згенерувати її онлайн. Все частіше з’являються нові алгоритми у цій сфері. Проте, вони мають обмеження, або потребують повного перерахунку моделі чи великої кількості ресурсів

У даній роботі, я проаналізував існуючі рішення для генерації текстових даних. Оскільки ця тема найчастіше зустрічається у ігровому просторі, більшість із прикладів будуть пов’язані із іграми. У даній сфері існує попит на генерацію текстів, які відрізняються за розміром, стилем та інтерактивністю. Я вважаю, що вирішення цієї задачі створює обмеження у витратах ресурсів та можливості створювати нову модель при кожному запуску.

# 1 Постановка задачі

Метою даної дипломної роботи є створення математичного та програмного забезпечення для генерації емоційнозабарвленного тексту в умовах ігрового простору.

Множина емоцій не фіксована, але у наведених прикладах розглядаються наступні емоції:

а)щастя;

б)сум;

в)злість;

При розробленні відповідного забезпечення я планую розв’язати наступні завдання:

а) проведення порівняльного аналізу існуючих методів генерації тексту;

б)вибір реалізація одного з методів генерації;

в) розробка програмного забезпечення на базі вибраного математичного методу;

Реалізована система має задовольняти такі вимоги:

а) мати можливість адаптуватися до різних умов генерації, без перебудови моделі

б) бути достатньо оптимальною с точки зору витрати ресурсів, для інтеграції у таку навантажену систему, як ігри

в) потреба у людських ресурсах

# 2 Аналіз існуючих методів

## 2.1 Математичні методи

### 2.1.1 Лінгвістичні моделі

Описана у [1] модель являє собою оцінку ймовірності певного розподілу з множини прикладів , кожен з яких складається з послідовності символів різної довжини .

Оцінка ймовірності вираховується за формулою:

При такій реалізації модель вираховує ймовірність вихідних символів при умові вхідних. Наступним кроком є отримання завдання як параметра, тоді формулу автори умовно описують як .

Деталі цього метода можна знайти у [5]. Суть цього процесу складається у отриманні *контексту*, *питання* та відповіді представлених у послідовності міток. Потім з цього формуються відповідні матриці, де рядок – відповідна репрезентація мітки у символах. Після навчання отримуємо матриці *контексту* та *питання*.

Така модель, при зміні відповідних ймовірностей символів, дозволяє реалізувати систему, яка залежить можем приймати завдання як вхідний параметр. Основною метою цих систем є набуття стійкою відносно цього параметру.

Ця модель була протестована на таких завданнях, як : моделювання мови, переклад, розуміння прочитаного, та відповідь на питання по тексту. З наступними результатами:

Моделювання мови, датасет «Children’s Book Test»: 93.3% на загальних іменниках та 89.1% на конкретних( наділених власною назвою) сутностях

Моделювання мови, датасет «LAMBADA»: 52.66% та 63.24% після урахування проблем з моделюванням останнього слова у реченні.

Моделювання мови, датасет «Winograd Schema Challenge»: 70.70%

Розуміння прочитаного, датасет «CoQA»: 55 за метрикою ROUGE F1, без збору відповідних даних

Переклад, Англійська-Французька: 5 BLEU

Переклад, Французька -Англійська: 11.5 BLEU

Відповідь на питання по тексту, датасет «SQUAD»: 4.1%

Основною вадою цього методу є розмір датасету «WebText», на якому проходило тренування - 40 GB. Обробка цього датасету генерує модель з розміром словника 50,257 слів. Робота з таким словником може потребувати великої кількості ресурсів.

### 2.1.2 Моделювання з блоків

Підхід моделі [2] будується на розбиті тексту на блоки и побудову ациклічного графу з цих блоків. Кожен наступний блок бере потрібну інформацію за попередніх блоків. Таким чином, один з блоків задає локацію, і всі наступні блоки використовують цю інформацію.

Такий підхід дозволяє використовувати статичну генерацію, тобто економний с точки зору ресурсів системи, але , як зазначили автору цієї моделі, потребує великої кількості планування та навичок проектування таких систем.

Також порівняння вручну підібраних блоків(Hs), згенерованих(Gs) та абсолютно випадковими(Rs) виявило, що різниця між Gs та Rs невелика, хоча різниця між Hs там Gs приблизно такого ж масштабу.

### 2.1.3 Використання міток або тегів

Підхід [3] заснований на принципі створення деякого контенту, та розмітки його відповідними тегами. Алгоритм генерації полягає у запиті за тегами, знаходженні усіх сутностей з відповідними тегами, та вибором випадкової сутності з знайдених. Така система набула широкого використання у багатьох іграх та сферах, тому її дуже легко розробити, підтримувати та використати у різних сферах без значних змін, але така схема потребує значної кількості людського планування так часу і погано масштабується із збільшенням кількості тегів.

### 2.1.4 Генерація з використанням ієрархій

Модель [4] побудована спеціально для використання у ігровій сфері. Вона полягає у декомпозиції ігрового завдання на під-завдання та події у вигляді графа. Обробка цього графу відбувається за допомогою системи планування і моніторингу.   
Згідно цієї моделі завдання можна представити у вигляді проблеми планування

P – атомарні символи, O- доступні операції планування, – початковий стан, - список цілей

Система планування генерує таке завдання, користувач виконує одну с доступних операцій змінюючи стан системи а система моніторингу постійно перевіряє поточний стан на відповідність очікуваному і при заходженні різниці генерує робить запит системі планування на генерацію нового завдання.

Така модель, згідно проведених досліджень у [4] має невелику потребу у ресурсах, але потребує змін при кожному рішенні користувача, та швидко набуває потреб у ресурсах при збільшенні вкладеності плану. Хоча на мою думку така система не є достатнім рішенням проблеми генерації тексту, але вона може виступати системою моніторингу абстрактного стану систем генерації. Зараз алгоритми для генерації полігонів Вороного використовуються для генерації початкового стану деякої мапи місцевості, беручи у увагу, недолік варіанту [1] з великим словником є можливість комбінації цих двох методів, який дозволить використати вже натреновану першу модель, але без використання всього словника. Метод з [4] може виступати розміткою потрібного словника і здійснювати його постійний моніторинг протягом інтеракції. Нажаль, знайти результати використання такої схеми мені не вдалося, але вона залишає за собою недолік у 40 Gb початкового датасету та постійних змін у структурі.

### 2.1.5 Марківські моделі

Описаний у [6] підхід заснований на генерації емоції за допомогою Марківських моделей, базується на ймовірностях переходу між станами, асоційованими з певною емоцією під дією сигналу, але описує лише процес генерації емоції.

Даний підхід може бути застосований, для вибору потрібної емоції і передачі управління наступному ланцюгу Маркова, який за отриманою емоцією буде генерувати текст. Нажаль, у такому вигляді, система не дає можливості описати умовну схему, тому не має можливості адаптації, але не використовує великої кількості ресурсів.

Таблиця 2.1 – Порівняння математичних методів

|  |  |  |  |
| --- | --- | --- | --- |
| Метод | Можливість адаптації | Використання ресурсів системи | Людські ресурси |
| Лінгвістичні моделі | + | -- | + |
| Марківські моделі | - | + | + |
| Моделювання з блоків | + | + | -- |
| Використання міток або тегів | - | ++ | - |
| Генерація з використанням ієрархій | - | +- | + |

## 2.2 Висновки

На основі проведеного порівняльного аналізу (таблиця 2.1) можна

зробити висновок, що основними методами можуть виступати лінгвістичні моделі, Марківські моделі та моделювання з блоків. Кожен з них має свої вагомі недоліки, але модифікація лінгвістичної моделі, с точки зору використання ресурсів, зараз не здається можливою. У аналізі Марківських моделей був запропонований метод, який дозволить вирішити проблему адаптації.

Адаптацію методу Марківських моделей можна отримати, якщо дозволити вкладеність ланцюгів, та замість опису конкретного ланцюга, описувати його схему, за якою вже буде створюватися ланцюг Маркова. Цей підхід схожий до моделювання блоків і потребує значно більших навичок проектування таких систем ніж звичайні Марківські моделі.

# 3. Математичне забезпечення

## 3.1 Принцип моделювання ланцюгами Маркова

У класичній інтерпретації ланцюга Маркова складається з множини станів та переходів. Переходи характеризуються ймовірністю, а стани відповідають поточному результату. У таких моделях випадкова величина описується станом.

У моїй моделі випадкова величина описується переходом, а стан характеризується лише можливими з нього переходами.

Розглянемо архітектуру такої моделі(рис 3.1).

9

11

101

12

13

8

7

9

Ланцюг 2

3

4

2

6

1

Ланцюг 1

Рисунок 3.1 – Архітектура вкладеного ланцюга Маркова

5

При переході між ланцюгами ми потрапляємо в той стан другого ланцюга, з якого було зроблено переміщення. При закінченні роботи одного з ланцюгів – керування передається ланцюгу який його викликав у місце виклику.

Ця модель базується на принципі ланцюгів Маркова, але, у загальному випадку, не відповідає визначенню ланцюга Маркова:

Де ,

Для доведення цього твердження потрібно ввести декілька позначень.

Загальним ланцюгом назвемо пару, яка складається з множин пар , де S- множина станів, P- множина переходів, - початковий стан, . Позначимо таку множину .

Множину переходів можна представити у вигляді пари матриць Де, - ймовірність переходу із стану в стан , - випадкова величина отримана при переході із стану в стан

Під-ланцюгом назвемо пару з множини .

Ланкою назвемо пару , де ,

Для забезпечення кінцевої генерації, сума ймовірностей переходів із стану менше або рівна 1, та наявна умова відсутності циклічних під-ланцюгів та ланок в яких не входить вершина з сумою ймовірностей переходів з неї менше 1:

|  |  |
| --- | --- |
|  | (1) |

|  |  |
| --- | --- |
|  | (2) |

Ймовірність – назвемо ймовірністю виходу з ланцюга.

Побудуємо початковий ланцюг , який складається і підрахуємо ймовірності виходу: , . За умови наявності переходу між двома ланцюгами, виконаємо роботу одного з них. Нам неважлива кількість таких переходів, та їх розташування, доки виконуються умови (1) і (2), оскільки один з ланцюгів першим закінчить виконання своєї роботи і керування перейде до іншого. Будемо вважати, що перший ланцюг закінчив роботу першим, тоді підрахуємо ймовірність виходу для другого ланцюга:

|  |  |
| --- | --- |
|  | (3) |

Оскільки між двома ланцюгами існує хоча б один перехід, остання ймовірність не дорівнює 0. Тоді, при умові - вихід, отримуємо

## 3.2 Формування моделі

Для формування системи будемо використовувати вкладений ланцюг Маркова з двох рівнів. Перший рівень описує ланцюг для емоцій, а другий – ланцюг для генерації тексту.

Щастя/0.33

Злість/0.33

Сум/0.33

Щастя/0.5

Сум/0.33

Сум/0.33

Злість/0.2

Сум/0.2

Щастя/0.33

Рисунок 3.2 – Перший рівень у вигляді ланцюга Маркова

Стан «A» відповідає за щастя, «B»- сум, «C» - злість а стан «0»- початковий стан. Кожен з цих станів відповідає окремому ланцюгу Маркова. Такі ланцюги представляють собою дерево, деякі з вершин якого переходять у інший стан ланцюгaа з першого рівня.

So what? /0.5

I’m all right. Thanks! /0.5

Hi! /0.5

I’m not feeling god/0.33

Sorry. I was daydreaming. How are you? /0.5

Рисунок 3.3 –Другий рівень стану «А» у вигляді ланцюга Маркова

Схеми використаних ланцюгів наведено у Додатку В.

## 3.3 Висновки

Розроблено математичне забезпечення системи генерації емоційнозабарвлених текстів. Для цього було спроектовано дворівневий вкладений ланцюг Маркова, де перший рівень відповідає за обранні емоції, а другий – за текст для відповідних емоції. У ході генерації відбувається зміна станів першого рівня як через переходи у другому, так і після кінця генерації у відповідному рівні.

За зупинку генерації відповідає ймовірність зупинки, яка розраховується за формулою:

# 4. Програмне забезпечення

## 4.1 Архітектура програми

Програмне забезпечення для розв’язання задачі складається з трьох частин: Builder,Linker,Chain. Перші дві компоненти відповідають за побудову ланцюга із схеми, а Chain – за реалізацію переміщення по ланцюгу. Для роботи ПЗ отримує схему ланцюга. Схема описує переходи між станами та складається з полів From,To,Chance,Type,String. Поля From і To відповідають за напрям переходу. Chance відповідає за ймовірність переходу, Type – перехід між ланцюгами, або між станами, String – рядок для генерації, або шлях до схеми ланцюга. У рядку для генерації можуть знаходитись змінні, які відповідають синтаксису: @variable\_name@. Змінні введені для надання можливості описувати шаблони у схемах, замість самих схем.

На рисунку 4.1. зображено схему роботи та взаємодії цих компонент.

Отриманий ланцюг

Генерація

Запит на лінкування

Linker

Chain

Builder

Рисунок 4.1 – Схема взаємодії модулів розробленого програмного забезпечення

Вхідний файл із схемою

Згідно опису робота програми складається з двох етапів: побудова ланцюга, використання ланцюга для генерації.

Компонента Builder виконує рекурсивну побудову ланцюга. Стадія побудови кожного ланцюга складається з pre-build та build етапів.

Pre-build відповідає за валідацію вхідного файлу, та розрахунок потрібної пам’яті. Валідація складається з перевірки формату та розміру файлу. Якщо файл не знайдено, або він пустий – виконується зупинку процесу Build.

Build стадія складається з рекурентного зчитування файлу та розпізнавання вказаної схеми. Якщо у зчитуваній схемі вказано, що стану відповідає ланцюг – builder відкриває відповідний файл та починає побудову вказаного ланцюга. Якщо у схемі вказано вже існуючий ланцюг – builder виконає зв’язок з цим ланцюгом. У разі отримання переходу для побудови речення builder проаналізує це речення на наявність змінних і, за потреби, зробить запит на лінкування.

За стадію лінкування відповідає модуль Linker. Він отримує потрібний рядок і знаходить у ньому зміну, яку потім замінює потрібним значенням. Така модель представляє собою статичне лінкування, потрібне для підтримки можливості опису схем. У розробленому програмному забезпеченні підтримана лише функція декларації змінної.

Після закінчення стадії Build, керування передається компоненті Chain. Робота компоненти Chain складається із вибору наступного переходу та реалізації переходу між станами. Вибір наступного стану відбувається шляхом генерації випадкового значення із рівномірного розподілу та порівняння його із ймовірністю активації переходу + ймовірність перевірених переходів.

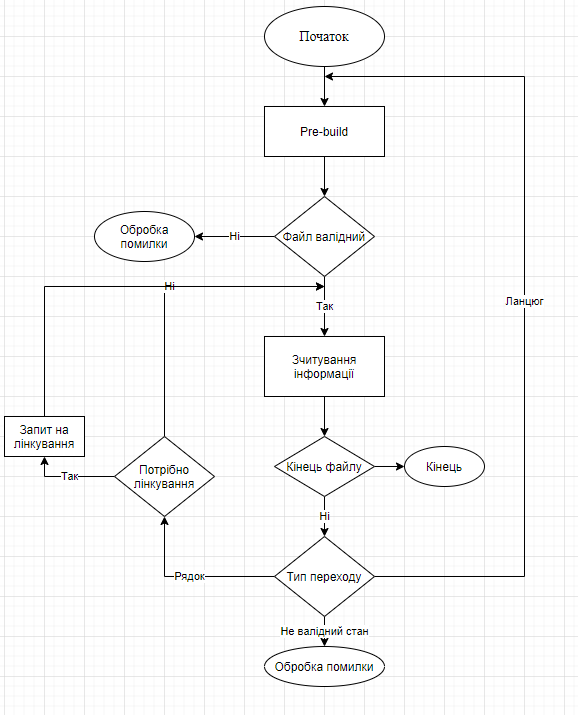
Після отримання потрібного переходу відбувається його активація, шляхом генерації потрібного рядка, або запуском вкладного ланцюга. Після активації відбувається пошук нового стану, доки не буде досягнуто стан виходу.

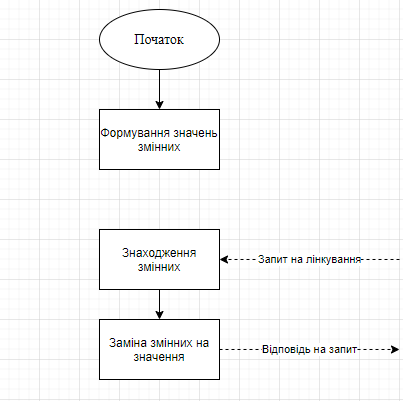
## 4.2 Опис розроблених алгоритмiв

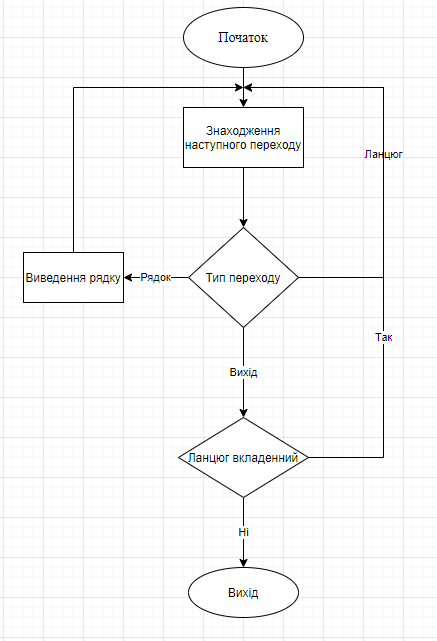
На рисунку 4.2 представлено схему алгоритму компоненти Build.

Блок-схема 4.3 описує алгоритм лінкування змінних.

Алгоритм зміни станів наведено на рисунку 4.4

Рисунок 4.2 – Блок-схема алгоритму компоненти Build

Рисунок 4.3 – Блок-схема алгоритму компоненти Link

Рисунок 4.4 – Блок-схема алгоритму компоненти Chain

## 4.3 Формат вхiдних даних

### 4.3.1 Формат схеми ланцюга

Вхідними даними для ініціалізації системи є схема вкладеного ланцюга Маркова із відповідними параметрами. Така схема описує переходи між станами. Для опису переходу використовують наступний параметри:

From – з якого стану відбувається перехід

To – у який стан відбувається перехід

Chance – ймовірність активації переходу

Type – тип переходу. “Chain” – відповідає ланцюгу, “String” – відповідає рядку

String – рядок для генерації, або шлях до файлу схеми ланцюга. Якщо тип переходу “String”, то рядок може містити зміні. Місце для вставки значення змінної відповідає синтаксису @variable\_name@

### 4.3.2 Форма файлу лінкування

Для ініціалізації модуль Linker потребує значень змінних, які описані у файлі Globals.txt. У цьому файлі перечисленні змінні у наступному вигляді:

* Операція над змінною. Наразі підтримується тільки операція define, яка декларує зміну
* Ім’я змінної
* Значення змінної

## 4.4 Формат результуючих даних

Результатом генерації емоційно запарвлених реплік є 10 згенерованих реплік (рисунок 4.5). Така кількість потрібна для показу можливостей схеми генерувати різні тексти.

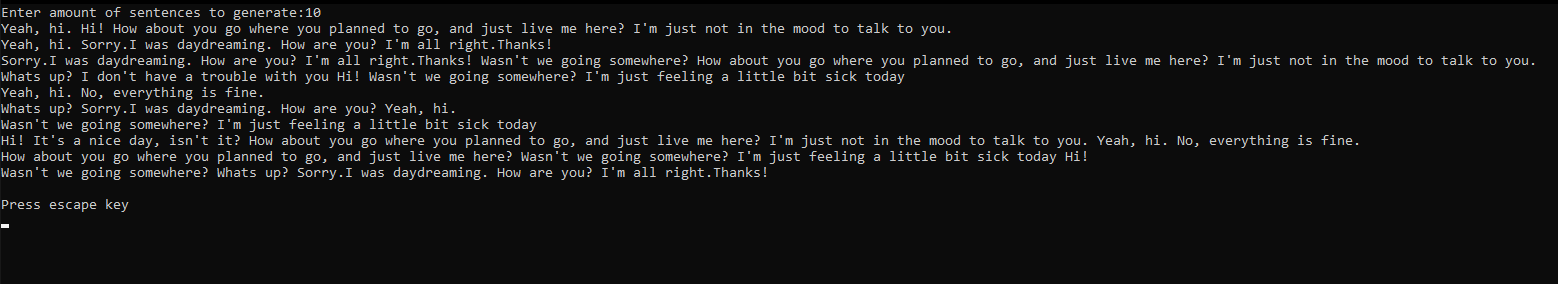
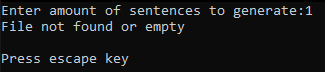
 Під час спроби завантажити невалідну схему ланцюга в програму, остання виведе повідомлення про помилку( рисунок 4.6).

Рисунок 4.5 – Приклад згенерованих речень

Рисунок 4.6 – Повідомлення про помилку

# Перелiк посилань

1. Language Models are Unsupervised Multitask Learners /Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever – 2019
2. Generating Side Quests from Building Blocks / Tomas Hromada, Martin Cerny, Michal Bıda, and Cyril Brom – 2015
3. Procedural Level and Story Generation Using Tag-Based Content Selection / Jurie Horneman
4. Hierarchical Generation of Dynamic and Nondeterministic Quests in Games/ Edirlei Soares de Lima, Bruno Feijó, Antonio L. Furtado – 2015
5. McCann, B., Keskar, N. S., Xiong, C., and Socher, R. The natural language decathlon: Multitask learning as question answering - 2018.
6. A Computational Architecture to Model Human Emotion / Arun Chandra/ - 2009
7. GPT-2: 1.5B Release [Електронний ресурс]. —

Режим доступу: <https://openai.com/blog/gpt-2-1-5b-release/>

1. GPT-2 GitHub page [Електронний ресурс]. — Режим доступу: https://github.com/openai/gpt-2