Міністерство освіти і науки України

Департамент освіти і науки Дніпропетровської облдержадміністрації

Дніпропетровське територіальне відділення МАН України

відділення: комп’ютерні науки

секція : мультимедійні системи, навчальні

та ігрові програми

СТВОРЕННЯ МОБІЛЬНОГО НАВЧАЛЬНОГО ДОДАТКУ ДЛЯ ПОШУКУ ХІМІЧНИХ ІЗОМЕРІВ

Роботу виконав:

Болотов Всеволод Володимирович,

учень 11 класу

комунальний заклад освіти

«Дніпропетровський обласний

ліцей-інтернат

фізико-математичного профілю»

Науковий керівник:

Іщеряков Сергій Михайлович,

кандидат технічних наук,

доцент кафедри комп’ютерних наук Державного університету телекомунікацій

Дніпро – 2020

ТЕЗИ

науково-дослідницької роботи

«Створення мобільного навчального додатку для пошуку хімічних ізомерів»

учня 11 класу

комунального закладу освіти

«Дніпропетровський обласний ліцей-інтернат

фізико-математичного профілю»

Болотова Всеволода Володимировича

Дніпропетровське територіальне відділення МАН України

Науковий керівник:

Іщеряков Сергій Михайлович,

кандидат технічних наук,

доцент кафедри комп’ютерних наук Державного університету телекомунікацій

Мета дослідження - створити мобільний додаток для допомоги учням у вивчення теми «структурні ізомери в хімії», який мав би простий та зрозумілий інтерфейс, розробити необхідні алгоритми та дослідити ефективність застосування додатку.

Актуальність: тема ізомерів є досить важливою у наш час, а додатки для її вивчення учнями фактично відсутні. Це, а також відсутність у відкритому доступі алгоритмів та порад до написання подібних додатків і роблять роботу актуальною.

Завдання :

* обрати інструменти розробки;
* визначити або розробити алгоритми для використання у програмі;
* створити програмну реалізацію;
* здійснити аналіз та апробацію даної програми та визначити її ефективність

Результати:

* досягнута мета роботи, розроблено навчальну програму для андроїд-пристроїв, що є зручною та зрозумілою в користуванні;
* узагальнено алгоритми створення структурних ізомерів;
* успішно проведена апробація роботи та доведена ефективність використання створеного додатку.

ЗМІСТ

[ЗМІСТ 2](#_Toc30426928)

[ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ 3](#_Toc30426929)

[ВСТУП 5](#_Toc30426930)

[ОСНОВНА ЧАСТИНА 7](#_Toc30426931)

[1.АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ 7](#_Toc30426932)

[2.РОЗРОБКА ДОДАТКУ 9](#_Toc30426933)

[2.1ОБҐРУНТУВАННЯ ВИБОРУ МОВИ ПРОГРАМУВАННЯ ТА ЗАСОБІВ РОЗРОБКИ 9](#_Toc30426934)

[2.2 ОПИС ТА ТЕРМІНОЛОГІЯ ДОДАТКУ 10](#_Toc30426935)

[2.2.1 ОПИС 10](#_Toc30426936)

[2.2.2. ТЕРМІНОЛОГІЯ ДОДАТКУ : 11](#_Toc30426937)

[2.3 ОПИС ТА РОЗРОБКА АРХІТЕКТУРИ ДОДАТКУ 11](#_Toc30426938)

[2.3.1 ГРАФІЧНА ЧАСТИНА 11](#_Toc30426939)

[2.3.2. ПРОГРАМНА ЧАСТИНА 13](#_Toc30426940)

[2.4 РОЗРОБКА АЛГОРИТМІВ 14](#_Toc30426941)

[2.4.1. МОЖЛИВІСТЬ ІСНУВАННЯ СПОЛУК 14](#_Toc30426942)

[2.4.2 ПОЯСНЕННЯ МОЖЛИВОСТІ ІСНУВАННЯ СПОЛУК 15](#_Toc30426943)

[2.4.3 ІЗОМОРФІЗМ ГРАФІВ 17](#_Toc30426944)

[2.4.4 ГЕНЕРАЦІЯ ІЗОМЕРІВ 18](#_Toc30426945)

[2.4.5 ВІЗУАЛІЗАЦІЯ 20](#_Toc30426946)

[3. АПРОБАЦІЯ 21](#_Toc30426947)

[3.1 ХІД АПРОБАЦІЇ 21](#_Toc30426948)

[3.2 РЕЗУЛЬТАТИ АПРОБАЦІЇ 22](#_Toc30426949)

[3.3 ІДЕЇ ЩОДО ПОДАЛЬШОГО РОЗВИТКУ 22](#_Toc30426950)

[ВИСНОВКИ 23](#_Toc30426951)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 24](#_Toc30426952)

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ ТА ТЕРМІНІВ

ArrayList – реалізація динамічного масиву у мові програмування java. Динамічний масив – це масив, що може змінювати розмір під час виконання програми.

Android – операційна система і платформа для мобільних телефонів, створена корпорацією Google на базі ядра Linux.

Enum, або перелічуваний тип даних – тип даних що складається з множини іменованих значень які називаються елементами, членами чи енумераторами типу.

IDE – інтегроване середовище розробки, комплексне рішення для написання програмного забезпечення.

Multiset (або Мультимножина) – це структура даних, що відповідає аналогічному поняттю у математиці, де мультимножина – це множина в якій для кожного елемента запам'ятовується не лише його входження, але й кількість входжень.

Java – об'єктно-орієнтована мова програмування, випущена як основний компонент платформи Java. Синтаксис мови запозичений від C та C++. Java-програми запускаються у середовищі віртуальної машини Java.

Атом (в контексті цієї роботи також елемент) – з хімічної точки зору найменша, електронейтральна, хімічно неподільна частинка речовини. Атоми характеризуються різними параметрами, але у контексті цієї роботи розглядається лише така характеристика, як валентність.

Вале́нтність – це властивість атомів одного хімічного елемента з'єднуватися з певним числом атомів інших хімічних елементів.

Гідроге́н – хімічний елемент з атомним номером 1. Має валентність 1. У хімії позначається латинською буквою «H» .

Зв’язаний граф – граф, у якому між будь-якими двома вершинами існує шлях.

Ізомери – це хімічні сполуки, що мають однакову брутто-формулу, тобто складаються з однакових атомів, але мають різну будову, через що й мають різні хімічні та фізичні властивості. Ізомерія буває структурна та просторова. У цій роботі розглядається структурна ізомерія.

Ізоморфізм графів – поняття, яким характеризується два(або більше) графа, якщо між множинами їх вершин можливо встановити взаємно однозначну відповідність, за якої дві вершина одно графа суміжні тоді і тільки тоді, коли відповідні вершини у другому графі теж суміжні.

Інцидентність – поняття, що використовується щодо вершини та ребра, якщо вершина зв’язана цим ребром з іншою вершиною.

Карбон – хімічний елемент з атомним номером 6. Має валентність 4. У хімії позначається латинською буквою «C» .

Мультиребра – два або більше ребр, що інциденті одним і тим же двом вершинам.

Незважений граф – граф, у якому у ребр нема ваги.

Неорієнтований граф – граф, у якому ребрам напрямок не задано.

Нітроген – хімічний елемент з атомним номером 7. Має валентність 3. У хімії позначається латинською буквою «N» .

Оксиге́н – хімічний елемент з атомним номером 8. Має валентність 2. У хімії позначається латинською буквою «O» .

Орієнтований граф – граф, у якому ребрам задано напрямок.

Петля (у теорії графів) – ребро, що починається та завершується у тій самій вершині.

Ступінь вершини – кількість ребр, що інциденті вершині.

Хімі́чна сполу́ка – речовина, молекули якої складаються з атомів двох або більше різних хімічних елементів, сполучених між собою. Сполука має певний хімічний склад і їй можна приписати точну хімічну формулу.

Ци́кл (в теорії графів) – послідовність з різних зв’язаних вершин, в якому перша та остання вершина збігається.

ВСТУП

Створення та розвиток комп’ютерів, комп’ютерних програм та мережі Інтернет ще у попередньому столітті дали значні можливості для розширення меж наявних у людства знань та дало змогу швидко та легко отримувати будь-яку інформацію за лічені секунди. З часом людство створило широкий спектр програм, що полегшують та покращують наше життя і без яких ми інколи навіть не можемо його уявити.

Але, незважаючи на доступність, комфортність та ефективність використання комп’ютерних технологій, на жаль, досі існують галузі, які майже не зазнали діджиталізації. Однією з таких галузей є сфера освіти. Незважаючи на той факт, що учні все більше часу проводять у електронному просторі[1], шкільна освіта ще мало у нього інтегрована[2].

З цією проблемою кожен може зустрітися у повсякденному житті. Зокрема, під час вивчення на уроках хімії у 9 – 10 класі тему «ізомери», я зіштовхнувся з повною відсутністю навчальних додатків, що могли б допомогти мені розібратися у цьому питанні.

Взагалі, спектр навчальних хімічних програм є відносно вузьким. Усі хімічні програми, наприклад [3] для персональних комп’ютерів призначені для професійних хіміків, тому, по-перше, майже усі вони платні, по-друге, мають складний інтерфейс, тож не підходять для учнів, які бажають розібратися у заданій темі. Мобільні додатки – навпаки, зазвичай мають простий інтерфейс, але, на жаль, я не знайшов жодного додатку, який би був призначений для вивчення вищезазначеної теми. Єдиний мобільний додаток, що містить цю тему – це [4]. Але у ньому наявні лише декілька тестів на необхідну тему, до того ж англійською мовою, тобто він теж не підходить для вивчення зазначеної теми.

Тож, відсутність у вільному доступі у мережі Інтернет аналогічних додатків та алгоритмів для їх написання й зумовлює актуальність цієї дослідницької роботи.

Мета дослідження – створити мобільний додаток для допомоги учням у вивчення теми «структурні ізомери в хімії», який мав би простий та зрозумілий інтерфейс, розробити необхідні алгоритми та дослідити його ефективність.

Для досягнення мети було вирішено розв’язати наступні задачі:

* обрати інструменти розробки;
* визначити або розробити алгоритми для використання у програмі;
* створити програмну реалізацію;
* здійснити аналіз та апробацію даної програми та визначити її ефективність.

Об’єктом дослідження є явище структурної ізомерії, а предметом дослідження є додаток, що має допомагати учню у поясненні та вивченні цього явища.

Наукова новизна полягає у тому, що було запропоновано принципово новий додаток, який має простий та зрозумілий інтерфейс, та розроблено алгоритми для його функціонування.

Результати роботи можуть бути використані для допомоги учням у вивченні явища структурної ізомерії, а також можуть бути застосовані програмістами при розробці програм аналогічної тематики для професійних хіміків.

ОСНОВНА ЧАСТИНА

1.АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

Зазвичай для пошуку та дослідження нових сполук та ізомерів хіміки використовують експериментальні методи[5], тобто спочатку отримують певну речовину, а потім вже її аналізують. Але, не дивлячись на це, для пошуку структурних ізомерів отриманих сполук хіміки розробили наступний алгоритм[6] (його розглядають на прикладі сполуки С6Н14 , її називають гексаном) :

1)Спочатку зображуємо молекулу лінійного ізомеру (її карбоновий скелет), як зображено на малюнку 1.1

(213 байт)

Мал. 1.1. Молекула лінійного ізомеру

2)Потім ланцюг скорочуємо на один атом карбону і цей атом приєднуємо до будь-якого атому карбону ланцюга як відгалуження від нього, виключаючи крайні положення (мал. 1.2 та 1.3):

(268 байт)     (267 байт)

Мал. 1.2. Ланцюги після скорочення на один атом

Якщо приєднати атом карбону до одного з крайніх положень, то хімічну будову ланцюга не зміниться (мал. 1.3):

(240 байт)

Мал. 1.3. Ланцюг після переміщення атому в крайнє положення

Крім того, потрібно стежити, щоб не було повторів. Так, структура на малюнку 1.4 ідентична тій, що на малюнку 1.2

(258 байт)

Мал. 1.4. Вигляд структури при повторенні

3)Коли всі положення основного ланцюгу вичерпано, скорочуємо ланцюг ще на один атом карбону. Тепер в бічних відгалуженнях розмістяться два атома карбону. Тут можливі такі поєднання атомів (мал.1.5):

(307 байт)  (321 байт)

Мал.1.5. Можливі поєднання атомів

4)Далі за можливістю потрібно перейти до пункту 2, тобто знову зменшити ланцюг і т. д.

5) Після побудови карбонового скелету у ізомерах необхідно доповнити всі атоми карбону зв'язками з воднем, враховуючи, що карбон чотиривалентний.

Таким чином, ми отримаємо усі 5 ізомерів заданої сполуки (мал 1.6).

(526 байт)

 (473 байт) (475 байт)

(501 байт) (493 байт)

Мал 1.6. Ізомери гексану

Але, на жаль, даний алгоритм має декілька значних вад:

* він пояснює лише пошук ізомерів для сполук, що не мають подвійних зв’язків,
* він не пояснює, як працювати зі сполуками, що мають атоми крім карбону та гідрогену у своєму складі.

Тож для досягнення мети роботи необхідно узагальнити цей алгоритм для роботи з будь-якими елементами та сполуками.

Після генерації ізомерів результат необхідно візуалізувати. Оскільки хімічні сполуки дуже зручно представляти у вигляді графів, то для візуалізації необхідно використовувати алгоритм візуалізації графів.

2.РОЗРОБКА ДОДАТКУ

2.1ОБҐРУНТУВАННЯ ВИБОРУ МОВИ ПРОГРАМУВАННЯ ТА ЗАСОБІВ РОЗРОБКИ

В якості мови програмування при розробці в наш час найчастіше використовують Java [7]. Це обумовлено наступними факторами:

1. Java – кросплатформенна мова програмування. Додатки, написані цією мовою, легко переносяться на пристрої, які керуються іншими операційними системами.
2. Java – сучасна та швидка мова програмування.
3. У стандартній бібліотеці Java є багато вбудованих алгоритмів які полегшують написання коду та спрощують його.
4. Java має автоматичне керування споживанням пам’яті, яке полегшує написання коду та економить оперативну пам’ять, що робить програму менш вимогливою до ресурсів пристрою.

В наш час велику популярність здобули мобільні пристрої під керуванням операційної системи Android. Програми для них дуже зручно писати за допомогою середовища Android Studio [8].

Android Studio– це інтегроване середовище (IDE) для роботи з платформою Android, анонсоване 16 травня 2013 року на конференції Google I/O менеджером з продукції корпорації Google - Еллі Паверс (Ellie Powers).

8 грудня 2014 року компанія Google випустила перший стабільний реліз Android Studio 1.0. Android Studio, заснована на програмному забезпеченні IntelliJ IDEA від компанії JetBrains, є офіційним засобом розробки Android додатків. Дане середовище доступне для Windows, MacOS і Linux. Мова програмування, яка використовується у цьому середовищі – Java.

На основі вищезазначеного, було вирішено в якості мови програмування обрати Java, а в якості середовища розробки – Android Studio.

2.2 ОПИС ТА ТЕРМІНОЛОГІЯ ДОДАТКУ

2.2.1 ОПИС

Основне призначення додатку – навчання, тож у користувача наявні наступні можливості :

1. ознайомитись із теоретичними відомостями на задану тему;
2. задати хімічний склад сполуки (з яких томів вона складається), з якою користувач бажає працювати;
3. спробувати намалювати усі можливі ізомери сполуки склад якої задано у п.2;
4. побачити всі можливі ізомери сполуки, заданої у п.2 ;

При спробі намалювати усі можливі ізомери (п.3), користувач має наступні можливості:

1. додати новий атом, з яким може працювати;
2. якщо вже додано більше 2-х атомів, між ними користувач може; встановити зв’язок, довгим дотиком на кожному з атомів;
3. видалити один з доданих атомів, перемістивши його до іконки «смітник»;
4. видалити зв’язок між 2-ма атомами, спочатку натиснувши на смітник, а потім на зв’язок;
5. користувач може переміщувати атоми екраном, натиснувши на відповідний атом і проводячи пальцем до місця, куди бажає перемістити;
6. коли користувач вважає, що він намалював ізомер заданої сполуки, він може це перевірити. Якщо він намалював правильно, програма про це повідомить. Якщо не правильно – про це теж повідомить;
7. після того, як користувач намалював усі можливі ізомери заданої сполуки, з’являється діалогове вікно з відповідним повідомленням;

При бажанні подивитись можливі ізомери користувач має наступний інтерфейс:

1. Малюнок ізомера, який користувач може подивитись та проаналізувати
2. Кнопки перемикання для перегляду попереднього чи наступного ізомерів.
3. Номер поточного ізомера та загальна кількість ізомерів на поточний час.

Через те, що ізомери генеруються не миттєво, доки цей процес триває,

користувачеві про це повідомляється.

2.2.2. ТЕРМІНОЛОГІЯ ДОДАТКУ :

Найголовнішими у цьому додатку були наступні поняття:

* атом (або елемент) – складова частина будь-якої молекули;
* молекула (або сполука, ізомер) – група атомів, що пов’язані між собою.

Дуже часто під час написання додатку робота з атомами та молекулами виражалася у термінах теорії графів, а саме:

* атоми позначались як вершини;
* зв’язки між атомами позначались як ребра;
* сполуки позначались як граф, що складається вершин та ребр.

З загально хімічних міркувань, графи, що є уособленням сполук у даному додатку, мають мати наступні властивості:

* граф може як містити цикли, так і не містити;
* граф може містити мультиребра, а може й не містити (мультиребра відповідають подвійним, потрійним і т. д. зв’язкам у хімії);
* граф має бути зв’язаним;
* граф не має петель;
* ступінь кожної вершини має дорівнювати валентності атому, якому вона відповідає;
* граф є неорієнтованим;
* граф є незваженим.

2.3 ОПИС ТА РОЗРОБКА АРХІТЕКТУРИ ДОДАТКУ

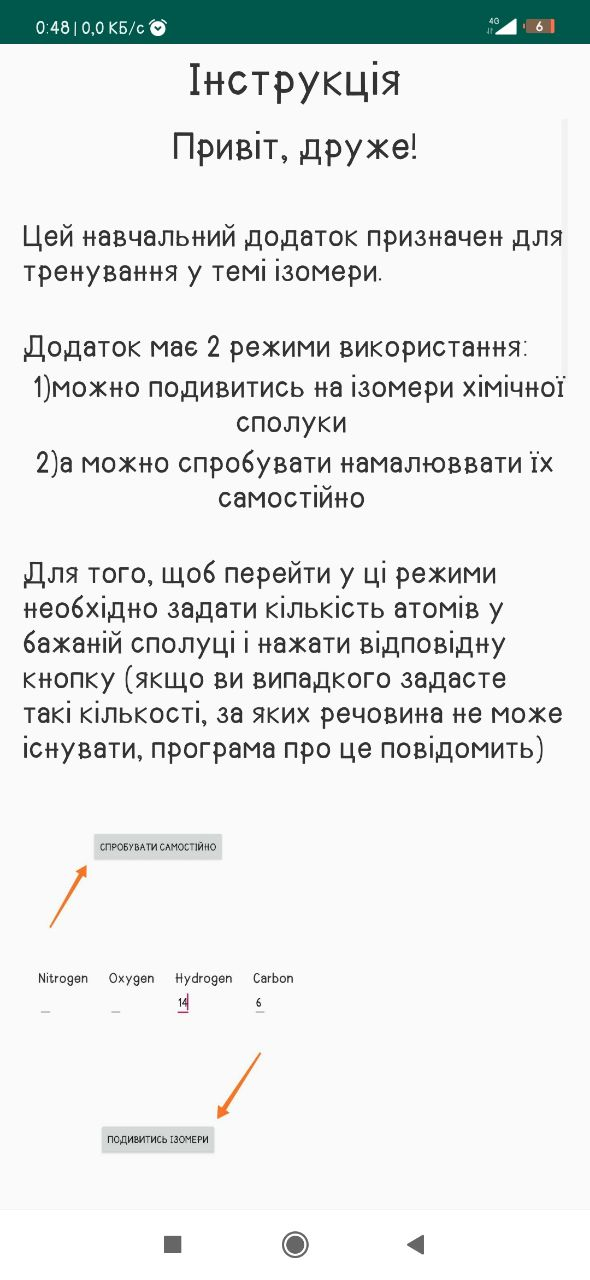
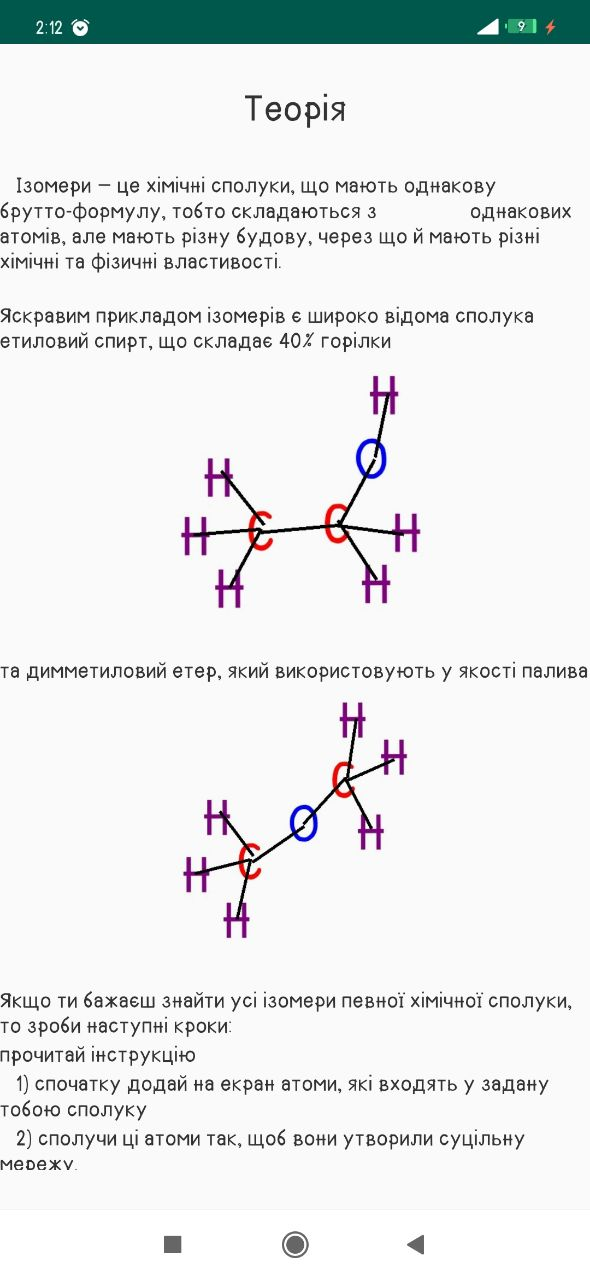
2.3.1 ГРАФІЧНА ЧАСТИНА

Графічна частина представлена вісьмома класами :

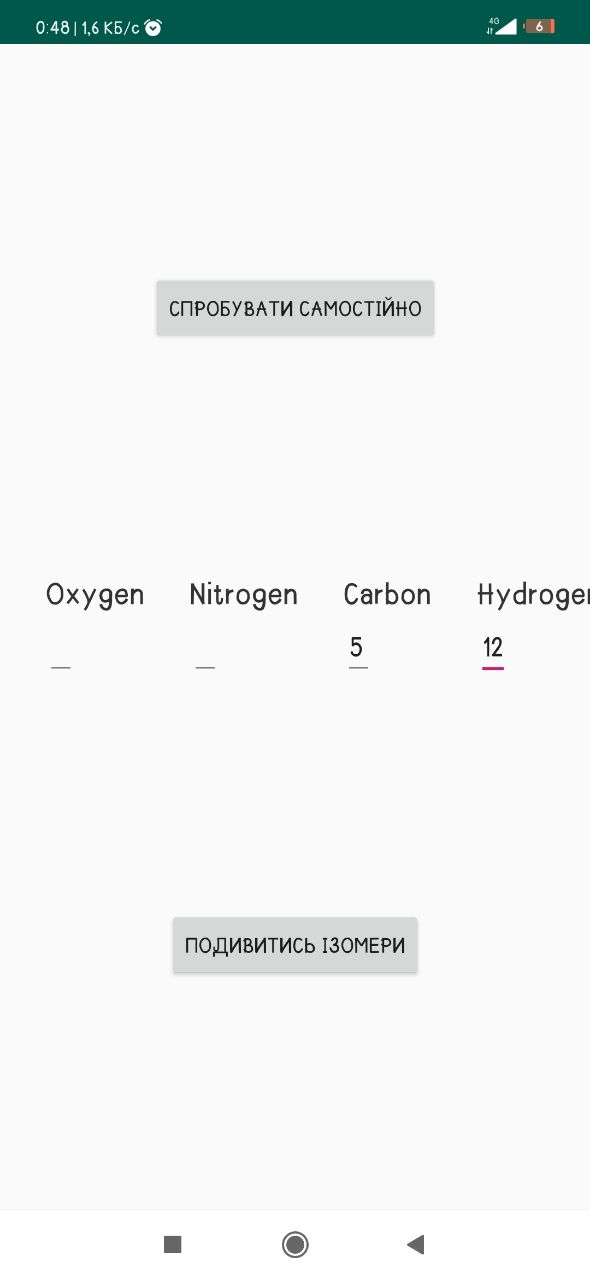
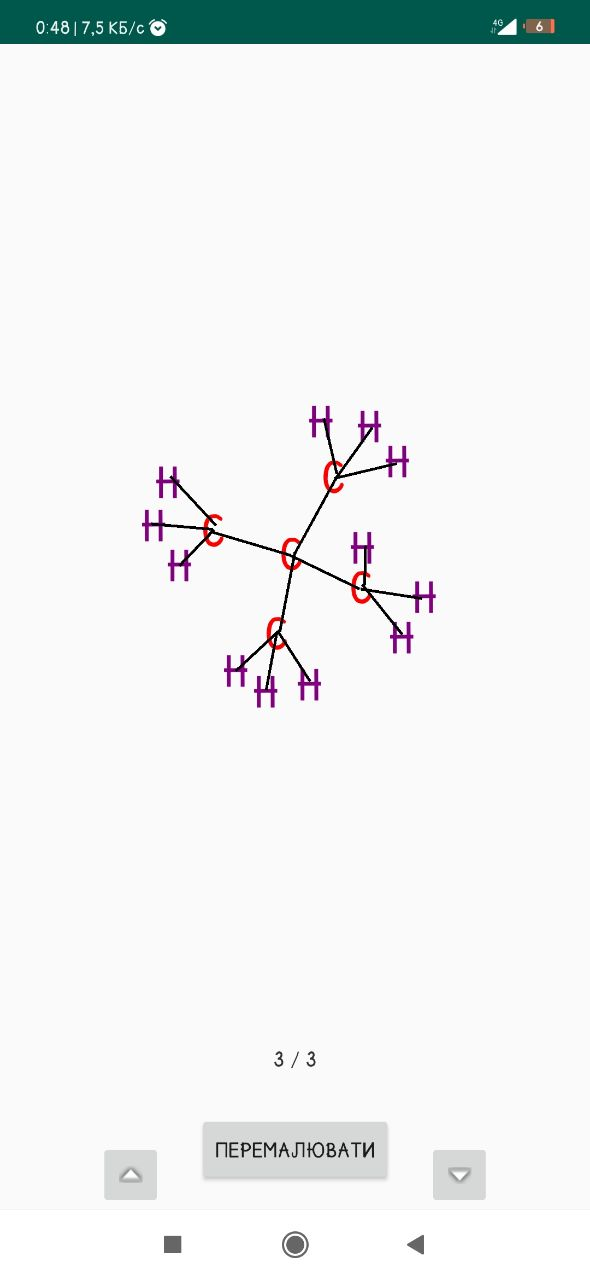
1. MainActivivty.java
2. InstructionActivity.java
3. TheoryActivity.java
4. ElementChooseActivity.java
5. DisplayerActivity.java
6. TrainActivity.java
7. ElementHelper.java
8. ImageHelper.java

Перші шість класів є нащадками класу AppCompatActivity і призначені для взаємодії користувача з додатком (малюнки: 2.1 - 2.6). Інтерфейс та можливості цих класів були описані у п.2.2.1.

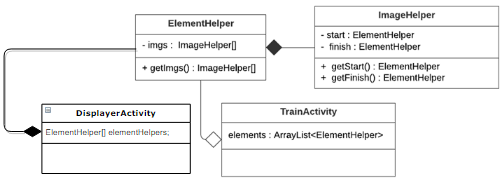
Класи ElementHelper.java та ImageHelper.java використовуються для пов’язування об’єктів TextView та ImageView, які доступні користувачеві з атомами та зв’язками між цими атомами відповідно (мал.2.7).

Мал. 2.1. MainCtivity Мал. 2.2. InstructionActivity Мал 2.3. TheoryActivity

Мал.2.4. ElementChooseActivity Мал.2.5. DisplayerActivity Мал.2.6. TrainActivity



Мал. 2.7. Взаємозв’язок між класами

2.3.2. ПРОГРАМНА ЧАСТИНА

Програмна частина реалізується класами з пакета Logic, зокрема:

1) Enum Elements, що реалізує відповідні хімічні елементи і складається з наступних констант :

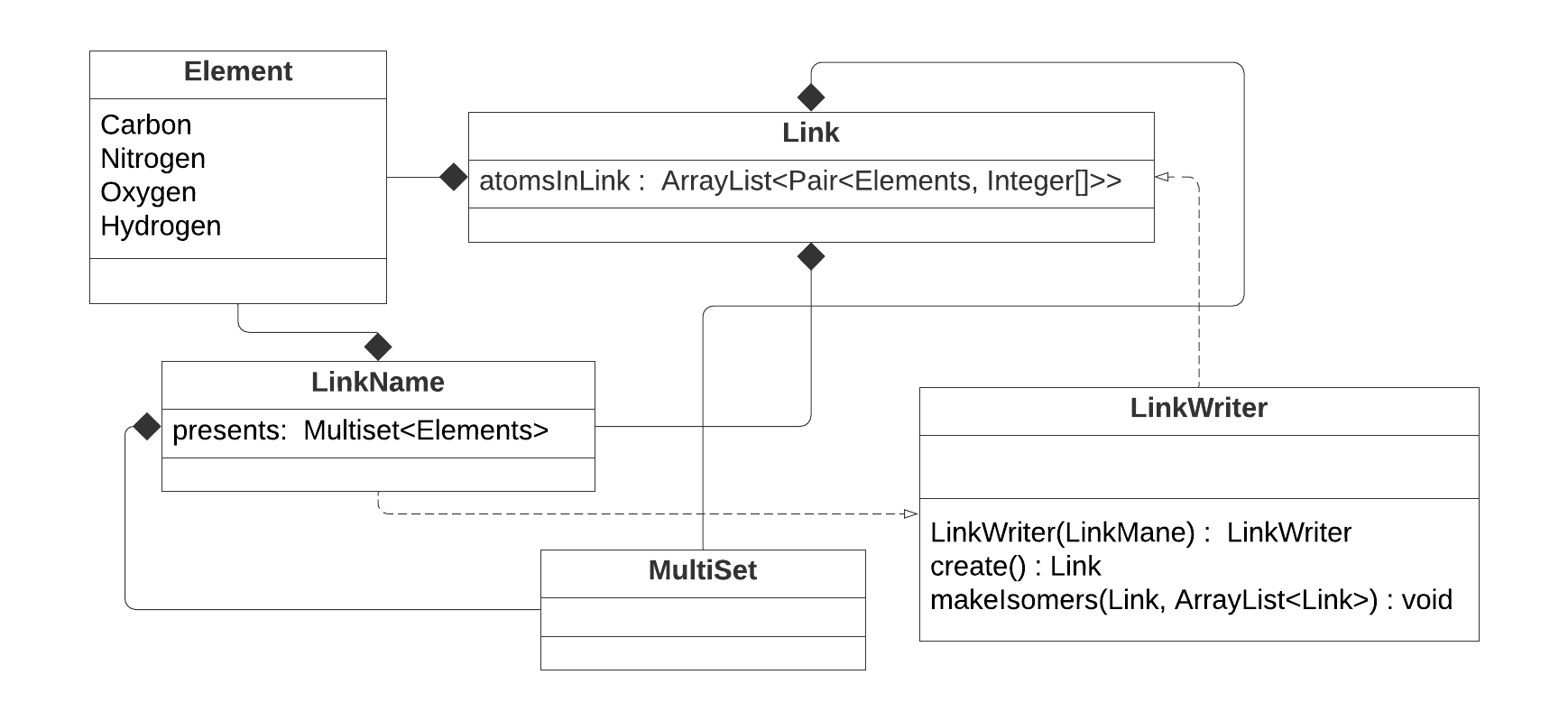
* Carbon;
* Nitrogen;
* Oxygen;
* Hydrogen.

2) Класом LinkName, що фактично відповідає за склад сполуки;

3) Класом Link, що відповідає сполуці, тобто містить інформацію про зв’язки атомів між атомами у сполуці;

4) Класом LinkWriter, що виконує основні функції програми, зокрема, містить методи create() для створення нового ізомеру з LinkName та makeIsomers() для пошуку усіх ізомерів заданої сполуки. (мал.2.8)

Також через відсутність реалізації такої структури даних, як MultiSet у вбудованих бібліотеках Java, використано реалізацію з сайту за посиланням[9]



Мал. 2.8 Взаємозв’язок між класами

2.4 РОЗРОБКА АЛГОРИТМІВ

2.4.1. МОЖЛИВІСТЬ ІСНУВАННЯ СПОЛУК

За логікою програми, користувач має задати кількість кожного виду атомів у сполуці, ізомери якої він бажає побачити. Після цього необхідно перевірити, чи може теоретично існувати така сполука.

Сполука може теоретично існувати, якщо з її елементів можливо створити зв’язаний граф, у якого ступінь кожної вершини дорівнює валентності атома, що у ній розташований.

Оскільки граф повинен бути зв’язаним, усі його атоми, валентність яких більше одиниці, мають бути з’єднані між собою, а одновалентні атоми повинні приєднуватись до багатовалентних і не мають бути з’єднані між собою (що є очевидним, бо інакше граф буде не зв’язаним).

Проведемо наступні міркування: спочатку перед генеруванням в нас усі атоми не пов’язані один з одним, і на місці невикористаних зв’язків наявні «вільні місця» . Тож загальна кількість таких «вільних місць» може бути визначена за формулою:

excess =,

де – це валентність i-го атома, n – кількість атомів. Так як усі атоми з валентністю більшою за один мають бути зв’язаними хоча б одинарними зв’язками, то в нас утвориться або

* два крайні атоми (які будуть зв’язані лише з одним іншим атомом) і всі інші атоми будуть не крайніми(вони будуть пов’язані з 2-ма атомами)

або

* буде більше крайніх атомів, але деякі некрайні атоми будуть пов’язані з більшою кількістю атомів.

У будь-якому разі для поєднання неодновалентних атомів нам необхідно зв’язків. Тож, враховуючи, що один зв’язок пов’язує два атоми, на зв’язування з одновалентними в нас залишається

,

де - це вільні місця, що в нас залишаються, а – кількість атомів з валентністю > 1.

Тепер на місця, що залишились, необхідно приєднати одновалентні елементи. Вочевидь, що якщо таких елементів більше ніж «вільних місць», то створити цю сполуку неможливо.

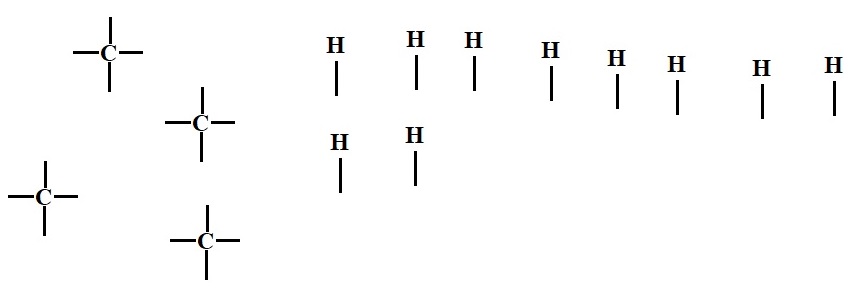
Тепер розглянемо випадок, коли нам вистачило місця. Якщо після приєднання одновалентних елементів місця взагалі не залишилось, то все гаразд.

Якщо після приєднання залишається місце, то вочевидь, воно знаходиться у неодновалентних елементах. Тож нам необхідно пов’язати вільні місця у атомах між собою. (більш наочно буде показано на прикладі). Очевидно, що оскільки при з’єднанні буде використано по одному зв’язку з атомів, що зв’язується, то повністю використати місце, що залишилось після приєднання одновалентних атомів можливо за умови, що в нас залишиться парна кількість «вільних місць» (якщо кількість непарна, то створити сполуку неможливо). Тож створити сполуку можливо, якщо кількість зв’язків, що залишилась, є парною.

2.4.2 ПОЯСНЕННЯ МОЖЛИВОСТІ ІСНУВАННЯ СПОЛУК

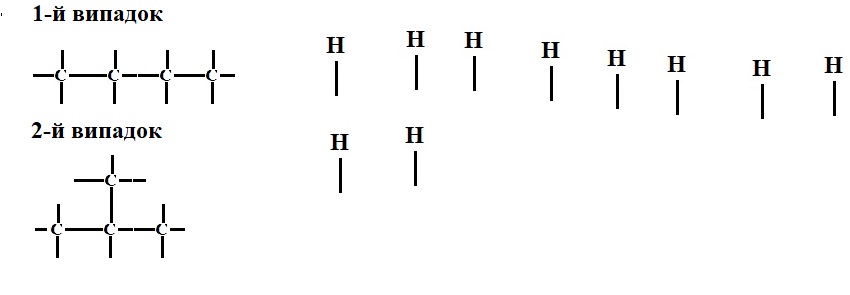
Візьмемо для прикладу сполуку бутан C4H10.

Спочатку усі атоми роз’єднані, що схематично можна зобразити, як малюнку 2.9. Загальна кількість “вільних місць” вираховується за наведеною вище формулою та складає

Мал. 2.9. Схематичне представлення початкового стану молекули у програмі

Тепер нам необхідно «пов’язати» багатовалентні атоми між собою.

Як і зазначалось вище, в нас або два крайні атоми, тоді усі інші зв’язані з двома. Або більше крайніх, але тоді некрайні зв’язані з більшою кількістю атомів. (мал. 2.10)



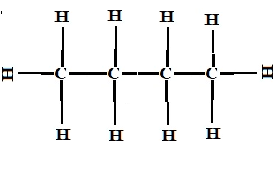
Мал. 2.10. Варіанти розміщення багатовалентних атомів і атоми гідрогену

Як бачимо, другий випадок ми можемо отримати, просто від’єднавши правий атом і приєднавши його до центрального, при цьому кількість місць, що залишиться для атомів гідрогену, не зміниться і буде дорівнювати

Тепер нам залишилось приєднати 10 атомів гідрогену (див. мал. 2.10, праворуч). Оскільки кожен зв’язок забирає по 1 «місцю» у кожного атома, що він зв’язує, то при з’єднанні буде витрачено , тобто залишиться

.

Тобто ми можемо створити сполуку, структурну формулу(схема розміщення атомів) якої зображено на малюнку 2.11 .

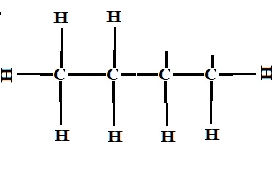


Мал. 2.11. Структурна формула можливого ізомеру бутану

Зауважимо, що за більшої кількості одновалентних атомів існування такої сполуки було б неможливим.

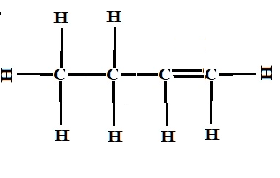
За меншої непарної кількості (наприклад дев’яти) одновалентних атомів, став би непарним числом, і створення такої сполуки також було б неможливим.

А ось за меншої парної кількості одновалентних атомів (наприклад, восьми), був парним додатним числом. У цьому випадку = 2, тож, у двох атомів залишаться «вільні місця». (мал 2.12)



Мал. 2.12. Структура молекули, якщо – парне додатне число

У цьому випадку можливо з’єднати 2 праві атоми карбону. Між ними утвориться подвійний зв’язок, або мультиребро мовою графів. (мал. 2.13)



Мал. 2.13 Сполука з подвійним зв’язком

Таким чином, ми з’ясовуємо, чи можливо створити цю сполуку. І якщо можливо, створюємо перший ізомер.

2.4.3 ІЗОМОРФІЗМ ГРАФІВ

Ізомери за хімічним визначенням – це сполуки, що мають однаковий склад, але різну будову.

Виходячи з цього визначення, необхідно знайти алгоритм, який би визначав, чи однакова будова двох графів, тобто, чи є вони ізоморфними.

За визначенням, 2 графи є ізоморфними, якщо існує взаємооднозначна відповідність між вершинами та ребрами, яка зберігає суміжність та інцидентність (тобто графи відрізняються лише назвами вершин).

Для визначення ізоморфізму графів було розроблено наступний алгоритм:

1)Виходячи з поняття ізоморфізму графів, якщо графи ізоморфні, кожній вершині у першому графі обрати відповідає вершина у другому графі, тож несуттєво, з якої саме вершини починати встановлювати відповідність. Тож оберемо довільну вершину у першому графі. Нехай це буде нульова вершина.

2) Тепер потрібно знайти відповідну їй вершину у другому графі. Вочевидь, атоми у відповідних вершинах мають бути однаковими. Оскільки невідомо, яка сама вершина є відповідною, розглянемо усі можливі по черзі.

3) Обрано вершини у обох графах. Далі дивимось на усіх «сусідів» цих атомів. Якщо «сусіди» співпадають кількісно і якісно (тобто однакова кількість однакових атомів), то повторюємо цю дію також для «сусідів». Таким чином, якщо графи однакові, то ми зможемо знайти кожному атому відповідний. Якщо ж це неможливо зробити, то графи різні, тож і ізомери теж різні.

Оціночно, алгоритм буде працювати за час O(n \* m^2), де n – кількість вершин, m – кількість ребер.

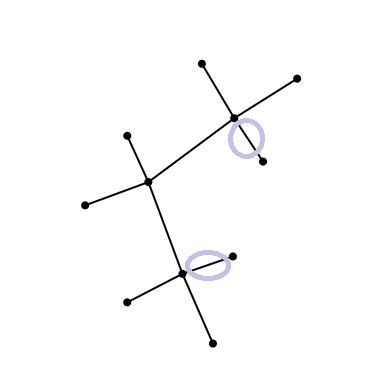
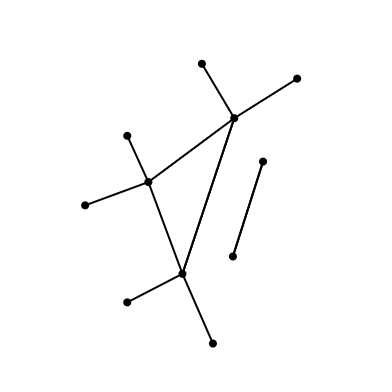
2.4.4 ГЕНЕРАЦІЯ ІЗОМЕРІВ

Генерація ізомерів певної речовини у даній роботі – це створення нових графів, що не є ізоморфними один одному, але відповідають вимогам, зазначеним у пункті 2.2.2.

За допомогою алгоритму, що описаний у пункті 2.4.2, створено перший ізомер. Тепер ми маємо створити усі інші ізомери. Новий ізомер можливо створити, якщо у наявному змінимо хоча б два зв’язки. Далі для отриманого ізомера повторюємо попередні дії. Вочевидь, при створенні нового ізомеру, потрібно перевіряти, чи не створювали ми такий самий.

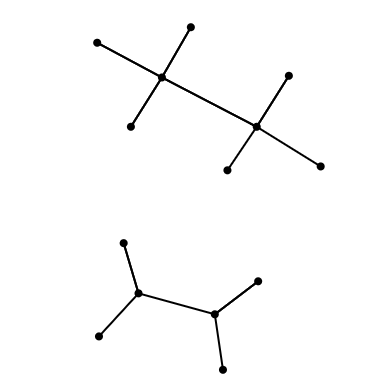
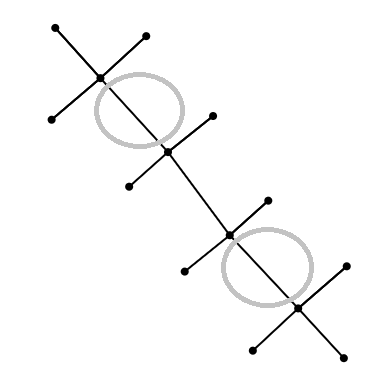
При цьому варто зауважити:

* якщо у обох зв’язках будуть присутні одновалентні елементи, то вони утворять між собою ізольовану пару (мал. 2.14), тож у таких обмінах нема сенсу. З цього випливає, що робити зміну варто лише у разі якщо хоча б один з атомів неодновалентний.

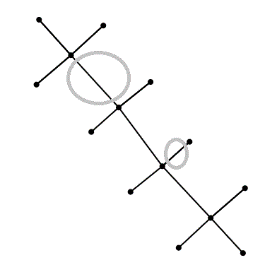
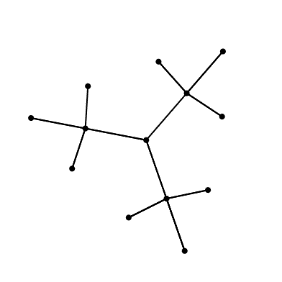
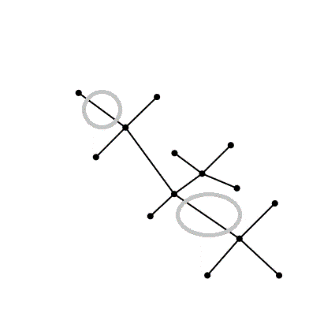
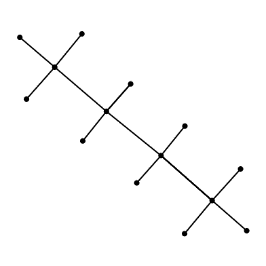
Мал. 2.14 Випадок утворення ізольованої пари одновалентних елементів

* аналогічна ситуація, якщо елемент не має подвійних (і більше) зв’язків та не є складовою циклу (мал. 2.15 ).



Мал.2.15. Випадок утворення ізольованих частин при обміні зв’язками між двома атомами, якщо один з них не має подвійних (і більше) зв’язків та не є складовою циклу

При цьому варто зауважити, що можливість поміняти місцями крайні елементи (неодновалентні) залишається (мал. 2.16).

Мал.2.16 Обмін місцями крайніх елементів.

Тож для генерації нових ізомерів здійснюються наступні дії :

1. Створюємо ArrayList ізомерів і додаємо до нього перший створений ізомер.
2. Для кожного ізомера, що є в ArrayList’і виконуємо наступні дії:
   1. Визначаємо атоми, що входять до циклів
   2. По черзі беремо неодновалентні атоми сполуки та виконуємо з ним наступне:
      1. Дивимось на усі зв’язки атома.
      2. Якщо атом не містить подвійних зв’язків, сполучений більше ніж з одним неодновалентним елементом та не є складовою циклів, повертаємось до п. 2.2 і беремо наступний атом.
      3. Визначаємо, які зв’язки атома можемо змінювати. Якщо атом є складовою циклу, чи є сполученим з лише одним атомом, то можливо змінювати усі його зв’язки. В іншому разі – лише неодинарні.
      4. Здійснюємо перебір усіх інших атомів та усіх їх зв’язків.
      5. Міняємо місцями можливі зв’язки першого та другого атомів. Таким чином, утворено новий ізомер
      6. Перевіряємо утворений ізомер на зв’язність. Якщо він зв’язний переходимо до наступного пункту. Якщо ні – повертаємось до пункту
      7. Після кожної зміни перевіряємо, чи не було вже такого самого ізомера. Якщо не було, додаємо до ArrayList’а ізомерів.

2.4.5 ВІЗУАЛІЗАЦІЯ

Після генерації ізомери необхідно візуалізувати. З-поміж існуючих алгоритмів візуалізації[10] найкраще для даного додатку використати алгоритм силової візуалізації графу, який полягає у представлення графа як фізичної системи[11], через високу якість отриманого зображення за невеликої кількості вершин та ребр.

Принцип дії алгоритму полягає у тому, що усім вершинам умовно надають електричний заряд, а зв’язані вершини умовно з’єднувати пружинами. Після цього вершини розташовують у довільних місцях і починають розраховувати сили, що діють у системі, а саме: сили відштовхування між усіма атомами, сили притягання між сполученими атомами та сила тертя, що сповільнює рух усіх атомів. Під дією цих сил атоми починають рухатися, і через невеликий проміжок часу встановлюється стан рівноваги, рух зупиняється і користувач бачить якісне зображення ізомера.

3. АПРОБАЦІЯ

3.1 ХІД АПРОБАЦІЇ

Для перевірки правильності роботи алгоритмів мною разом із вчителем хімії проведено перевірку діяльності програми у значній кількості різноманітних випадків. Перевірка не виявила помилок.

Для апробації цієї програми зроблено наступні кроки:

1. Трьом групам учнів КЗО «Дніпропетровський обласний ліцей-інтернат фізико-математичного профілю», що складалися з 5 осіб дев’ятого, десятого та одинадцятого класів, були дані завдання, які порадив викладач хімії, за темою роботи. Після перевірки результати були занесені у перші рядки таблиць 3.1 – 3.3.
2. Учням цих груп було запропоновано скористатись мобільним додатком, описаним у цій роботі, для покращення своїх знань та навичок.
3. Через 2 дні знов було зібрано усі три групи. Їм було роздано нові завдання, аналогічні попереднім. Після перевірки результати було занесено у другі рядки таблиць 3.1 – 3.3.
4. Потім було проаналізовано результати кожної групи до використання додатку та після та визначив середній результат кожної групи у кожному випадку.

Таблиця 3.1

Результати учнів 9-го класу

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Учень | 1 | 2 | 3 | 4 | 5 | Середній результат |
| Результат до використання додатку | 60% | 53% | 28% | 35% | 74% | 50% |
| Результат після використання додатку | 73% | 64% | 53% | 64% | 80% | 66,8% |

Таблиця 3.2

Результати учнів 10-го класу

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Учень | 6 | 7 | 8 | 9 | 10 | Середній результат |
| Результат до використання додатку | 63% | 70% | 70% | 89% | 45% | 67,4% |
| Результат після використання додатку | 74% | 92% | 77% | 100% | 45% | 77,6% |

Таблиця 3.3

Результати учнів 11-го класу

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Учень | 11 | 12 | 13 | 14 | 15 | Середній результат |
| Результат до використання додатку | 45% | 70% | 83% | 54% | 60% | 62,4% |
| Результат після використання додатку | 58% | 89% | 94% | 57% | 76% | 74,8% |

Як видно з данних наведених таблиць, середні результати учнів після використання додатку у кожному разі покращувалися, що свідчить про ефективність програми.

3.2 РЕЗУЛЬТАТИ АПРОБАЦІЇ

Апробація довела ефективність використання додатку.

Після апробації додаток був допрацьований, у нього були внесені наступні зміни:

* додано підказки та пояснення інтерфейсу;
* виправлено помилки, виявлені при тестуванні;
* додано блок з теоретичними відомостями.

3.3 ІДЕЇ ЩОДО ПОДАЛЬШОГО РОЗВИТКУ

* створити базу даних згенерованих ізомерів;
* оптимізувати алгоритми;
* створити клієнт-серверну частину, де сервер буде визначити, ізомери яких сполук генерувати і відправляти до клієнтів (на пристрої з встановленим цим додатком). На пристрої клієнта ізомери згенеруються та відправляться на сервер, де будуть заноситися у базу даних. Таким чином можна зробити велику базу даних ізомерів, яку зможуть використовувати хіміки.

ВИСНОВКИ

1. Проведено аналіз існуючих програм на задану тематику та розглянуто існуючий алгоритм пошуку ізомерів.
2. Визначено мету роботи, предмет та об’єкт дослідження.
3. Розроблено алгоритми для ефективної роботи додатку, а саме:

* алгоритм визначення можливості існування сполуки, який також створює її перший ізомер;
* алгоритм визначення ізоморфізму графів;
* алгоритм пошуку ізомерів будь-якої хімічної сполуки;

1. Знайдено та успішно реалізовано алгоритм для візуалізації отриманих ізомерів.
2. Спираючись на аналіз наявних мов програмування, для реалізації проекту обрано Java, середовище розробки Android Studio.
3. Досягнута мета роботи, розроблено навчальну програму для андроїд-пристроїв, що є зручною та зрозумілою в користуванні.
4. Проведено апробацію додатку – тестування знань серед учнів КЗО «Дніпропетровський обласний ліцей-інтернат фізико-математичного профілю» з хімії за темою роботи до та після використання додатку; оброблено результати, враховано відгуки. Помічено позитивну динаміку серед учнів, що використали цей додаток. Покращено якість знань учнів з хімії.
5. Тестування підтвердило ефективність використання цього додатку, що, в свою чергу, свідчить про правильність та надійність розроблених алгоритмів.
6. Результати роботи можуть бути використані для покращення вивчення хімії у школі за темою роботи, а також при розробці мобільних додатків та комп’ютерних програм аналогічного напрямку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Скільки часу діти проводять перед екраном [Електронний ресурс]. Режим доступу: <https://www.bbc.com/ukrainian/science/2015/03/150329_children_screens_she>
2. Цифрові технології в освіті [Електронний ресурс]. Режим доступу http://rozumniki.com/info/news/tsyfrovi\_tekhnologiyi\_v\_osviti\_tse\_ne\_trend\_a\_vymoga\_chasu/
3. Find Molecula (програма для персональних комп’ютерів) [Електронний ресурс]. Режим доступу <https://findmolecule.com/?orig=capterra>
4. Chirality 2(мобільний додаток ) [Електронний ресурс]. Режим доступу <https://play.google.com/store/apps/details?id=rmit.edu.au.oliverjones&hl=en>
5. Методи дослідження у хімії [Електронний ресурс]. – Режим доступу: <https://miyklas.com.ua/p/himija/7-klas/pochatkovi-khimichni-poniattia-39391/fizichni-vlastivosti-rechovin-39142/re-656bc8c7-77f2-4158-9cfe-6e53a2f9b709>
6. Приемы построения структурных формул изомеров [Електронний ресурс]. – Режим доступу: <http://orgchem.ru/chem2/u231_1.html>
7. Рейтинг мов програмування [Електронний ресурс]. – Режим доступу: https://spectrum.ieee.org/static/interactive-the-top-programming-languages
8. Офіційна документація Android. [Електронний ресурс]. – Режим доступу: <https://developer.android.com>
9. MultiSet implementation in java [Електронний ресурс]. – Режим доступу: <https://www.techiedelight.com/multiset-implementation-java/>
10. Алгоритми візуалізації графів [Електронний ресурс] – Режим доступу <https://habr.com/ru/company/ods/blog/464715/>
11. Силові методи візуалізації графів [Електронний ресурс]. – режим доступу <http://www.cyberforum.ru/blogs/32521/blog1977.html>