Міністерство освіти і науки України

**Національний університет «Запорізька політехніка»**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Комп’ютерні системи і мережі\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(найменування кафедри)

**КУРСОВИЙ** **ПРОЕКТ**

**(РОБОТА)**

з \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Комп'ютерних Систем\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(назва дисципліни)

на тему:\_\_ Розподілена система яка використовує java клас Math\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студента (ки) \_III\_ курсу КНТ-518 групи

Спеціальності 123 комп’ютерна інженерія

освітня програма (спеціалізація) \_\_\_\_Комп’ютерні системи та мережі \_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_Костецький Д.В.\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище та ініціали)

Керівник \_\_\_\_Точилін С.Д.\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_ к.ф.-м.н., доцент\_\_\_\_\_\_\_\_\_\_\_\_

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_Оцінка: ECTS \_\_\_\_\_

Члени комісії \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (прізвище та ініціали

2021 рік

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Кафедра\_\_\_\_\_\_\_Комп’ютерні системи і мережі\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дисципліна \_\_\_\_**«**Комп'ютені системи»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Спеціальність \_\_123Комп’ютерна інженерія\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Курс 3 Група КНТ-518 Семестр 6

**ЗАВДАННЯ**

**на курсовий проект (роботу) студентові**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Костецького Дмитра Віталійовича\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(прізвище, ім’я, по батькові)

1. Тема проекту (роботи): \_\_ Розподілена система яка використовує java клас Math

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Термін здачі студентом закінченого проекту (роботи):\_\_\_1.05.2021\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Вихідні дані до проекту (роботи): Забезпечити введення та пошук даних\_\_\_\_\_\_

згідно з предметною областю; розробити додаток, за допомогою якого \_\_\_\_\_\_\_\_\_

забезпечити обробку інформації, створення форм, необхідних запитів \_\_\_\_\_\_\_\_\_\_

та звітів; доступ до інформації організувати за допомогою Web-\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

інтерфейсу.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. ЗМІСТ розрахунково-пояснювальної записки (перелік питань, що їх належить розробити): \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Вступна частина (титул, завдання, реферат, зміст, перелік умовних скороч.) \_\_\_\_\_\_\_\_\_\_\_\_

\_\_Основна частина (вступ, спеціальні розділи)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_Заключна частина (висновки, перелік посилань)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень):

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6. Дата видачі завдання:\_20.03.2021\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* + - 1. **КАЛЕНДАРНИЙ ПЛАН**

|  |  |  |  |
| --- | --- | --- | --- |
| №  з/п | Назва етапів курсового  проекту (роботи) | Строк виконання етапів курсового проекту ( роботи ) | Примітка |
| 1 | Отримання завдання на курсову роботу | 20.03.2021 |  |
| 2 | Дослідження літератури та матеріалів на задану тему | 5.04.2021 |  |
| 3 | Виконання пояснювальної записки | 10.04.2021 |  |
|  | Вступна частина | 20.04.2021 |  |
|  | Основна частина | 22.04.2021 |  |
|  | Заключна частина | 30.04.2021 |  |
| 4 | Представлення закінченої роботи на перевірку | 01.05.2021 |  |
| 5 | Захист курсової роботи |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_Костецький Д. В.\_\_­­­­\_\_\_\_\_\_\_\_\_\_\_\_

( підпис ) (прізвище та ініціали)

Керівник проекту (роботи) \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_ Точилін С.Д.\_\_\_\_\_\_\_

( підпис ) (прізвище та ініціали)

«01» Травня 2021 р.

**РЕФЕРАТ**

ПЗ: 46 с., 6 рис., 7 лістингів, 1 табл., 9 джерел, 1 додаток

JAVA, Android Studio, XML, додаток, Math, клас, EditText, TextView, Button, Andriod, функція, пакет

Об’єкт розробки- Розподілена система яка використовує java клас Math.

Мета проекту- навчитися працювати з усіма методами класу Math та застосовувати на практиці.

В пояснювальній записці розроблений додаток написаний на мові JAVA який повністю відповідає поставленому завданню.

Розроблений додаток містить форми для розв’язання основних формул тригонометрії, квадратних рівнянь та генерації випадкових чисел. При його створенні була задіяна мова програмування JAVA, клас Math що поставляється в пакеті java.lang, та середовище розробки Android Studio.

**ЗМІСТ**

[**Вступ** 6](#_Toc70448881)

[1 Мова програмування JAVA 7](#_Toc70448882)

[1.1 Історія мови 7](#_Toc70448883)

[1.2 Структура мови 10](#_Toc70448884)

[1.3 Пакети 17](#_Toc70448885)

[2 Класс Math 20](#_Toc70448886)

[2.1 Інформація про клас 20](#_Toc70448887)

[2.2 Тригонометричні функції 20](#_Toc70448888)

[2.3 Алгебраїчні функції 21](#_Toc70448889)

[2.4 Інші функції 22](#_Toc70448890)

[3 Середовище розробки Android Studio 23](#_Toc70448891)

[3.1 Історія Android Studio 23](#_Toc70448892)

[3.2 Основні можливості середовища розробки 25](#_Toc70448893)

[4 Практична частина 26](#_Toc70448894)

[4.1 Постановка задачі 26](#_Toc70448895)

[4.2 Складання технічного завдання на розробку додатку. 26](#_Toc70448896)

[4.3 Опис функцій на методів що використовувались при розробці 27](#_Toc70448897)

[4.4 Огляд можливостей розробленого додатку 33](#_Toc70448898)

[Висновок 35](#_Toc70448899)

[Перелік джерел посилань 36](#_Toc70448900)

[Додаток А 37](#_Toc70448901)

# **Вступ**

Ми живемо у вік який вимагає велику кількість обчислень при проведенні інженерних робіт, конструкторських і т.п, навіть ми маємо справу з числами, проводимо операції над ними: множимо, складаємо, зводимо в ступінь і т.п. Щось ми можемо порахувати в умі, щось вирішити на листочку. Якщо нам необхідно за найкоротший відрізок часу отримати результат обчислення, то ми користуємося калькулятором.

Калькулятор - пристрій для арифметичних обчислень різних ступенів складності.

Звичайний калькулятор виконує додавання і віднімання, множення і ділення, а також обчислює відсотки. Будь-яке число, яке ви вводите в калькулятор, перетворюється в бінарний код, а результати в бінарному коді переводяться назад в десяткову систему і відображаються на дисплеї.

Інженерний калькулятор виконує складні інженерні та наукові розрахунки. Він здатний обчислювати велику кількість різноманітних функцій: гіперболічних, статечних, показових, логарифмічних і найпростіших статистичних функцій.

Але все ж краще використовувати калькулятор написаний на якійсь мові програмування. Мови програмування як правило володіють такими засобами для ефективного виконання математичних операцій. Але для підрахунку більш складних та точних операцій у мові програмування java використовується клас Math пакету java.lang. Основною перевагою такого калькулятора є те що він може виконувати ті ж самі функції що і калькулятори описані вище, але в ту ж чергу можна в нього запрограмувати повноцінні формули і рівняння для їх швидкого підрахунку.

Для виконання курсового проекту були поставлені наступні завдання: ознайомитися з середовищем розробки Android Studio, проаналізувати те як робити підрахунки максимально точно і по можливості не округляти їх, розібратися з роботою і поведінкою класу Math мови JAVA що поставляється в пакеті java.lang при виконанні різних обчислень.

## 1 Мова програмування JAVA

### 1.1 Історія мови

Мова програмування - це формальна знакова система, призначена для запису комп'ютерних програм. Мова програмування визначає набір лексичних, синтаксичних і семантичних правил, що визначають зовнішній вигляд програми і дії, які виконає виконавець (зазвичай -ЕВМ) під її управлінням.

Java - це об'єктно-орієнтована мова програмування, розроблений компанією Sun Microsystems (в подальшому придбаної компанією Oracle) .Пріложенія Java зазвичай транслюються в спеціальній байт-код, тому вони можуть працювати на будь-якій віртуальній Java-машині незалежно від комп’ютерної архітектури. Дата офіційного випуску- 23 травня 1995 року.

Спочатку мова називалася Oak ( «Дуб»), розроблялася Джеймсом Гослінгом для програмування побутових електронних пристроїв. Через те, що мова з такою назвою вже існувала, Oak був перейменований в Java [4].

Існує 3 основні версії чому так назвали мову:

Версія № 1: Найцікавіша. Програмісти п'ють багато кави. Поки створювався нову мову програмування, було випито гігалітри кави, і тому вирішили назвати мову в честь сорту кави Java.

Версія № 2: Назва дана на честь марки кави Java, який був названий на честь острова Ява (на англ. - Java). Ця версія перегукується з версією № 1.

Версія № 3: Оскільки нова мова розроблялася для програмування побутових електронних пристроїв, то іноді пов'язують це з ілюзією на каво-машину як приклад побутового пристрою.

Програми на Java транслюються в байт-код, що виконується віртуальною java-машиною (JVM) - програмою, обробній байтовий код і передавальної інструкції обладнанню як інтерпретатор, але з тією відмінністю, що байтовий код, на відміну від тексту, обробляється значно швидше. Гідність подібного способу виконання програм - в повній незалежності байт-коду від ОС і устаткування, що дозволяє виконувати Java-додатки на будь-якому пристрої, який підтримує віртуальну машину. Додатки можна запустити на багатьох платформах. Одного разу написаний додаток не доведеться модифікувати під інші платформи: воно буде працювати без будь-яких змін на різних операційних системах і апаратних архітектурах.

Цикл розробки програмних засобів з використанням Java значно скорочується в силу того, що Java - інтерпретована мова. Процес компіляції-збірки-завантаження - тепер програму треба тільки відкомпілювати і відразу запускати. Java контролює звернення до пам'яті. Незважаючи на те, що ця мова - інтерпретована, його код оптимізується до фази виконання. Підтримка системи багатопоточності дозволяє створювати паралельно виконувані взаємодіючі легковагі процеси.

Важливою особливістю технології Java є гнучка система безпеки, в рамках якої виконання програми повністю контролюється віртуальною машиною. Будь-які операції, які перевищують встановлені повноваження програми (наприклад, спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером), викликають негайне переривання.

Часто до недоліків концепції віртуальної машини відносять зниження продуктивності. Ряд удосконалень кілька збільшив швидкість виконання програм на Java:

-застосування технології трансляції байт-коду в машинний код безпосередньо під час роботи програми (JIT-технологія) з можливістю збереження версій класу в машинному коді,

-обширне використання переносних орієнтованого коду (native-код) в стандартних бібліотеках,

-апаратні кошти, що забезпечують прискорену обробку байт-коду (наприклад, технологія Jazelle, підтримувана деякими процесорами архітектури ARM).

За даними сайту shootout.alioth.debian.org, для семи різних завдань час виконання на Java становить в середньому в півтора-два рази більше, ніж для C / C ++, в деяких випадках Java швидше, а в окремих випадках в 7 разів повільніше. З іншого боку, для більшості з них споживання пам'яті Java-машиною було в 10-30 разів більше, ніж програмою на C / C ++. Також примітно дослідження, проведене компанією Google, згідно з яким відзначається істотно нижча продуктивність і більше споживання пам'яті в тестових прикладах на Java в порівнянні з аналогічними програмами на C ++.

За даними компанії Oracle, програми на Java запускаються на 3 млрд девайсів. Це маркетингове повідомлення складно перевірити. Проте Java широко використовується і входить в число найбільш затребуваних мов, це не викликає сумніву.

Наприклад, переважна більшість великих компаній так чи інакше використовують Java. Дуже багато серверних додатків для корпорацій написані на цій мові. Наприклад, мова йде про програми для фінансових організацій, які забезпечують проведення транзакцій, фіксацію торгових операцій.

На Java написано багато веб-додатків. Популярні фреймворки, в тому числі Spring, Stuts, JSP, використовуються для створення різних додатків в інтернеті: від ecommerce-проектів до великих порталів, від освітніх платформ до урядових ресурсів. Популярна комп'ютерна гра Minecraft написана на Java.

Мобільна розробка - ще одна область використання Java. Цією мовою пишуть програми для пристроїв, що працюють під управлінням ОС Android.

На Java створюють клієнтські програми. Простий і близький розробникам приклад: IDE NetBeans написано на «Джаві».

Також Java застосовується для роботи з Big Data, розробки програм для наукових цілей, наприклад, обробки природних мов, програмування приладів - від побутових девайсів до промислових установок.

Тобто на Java можна писати різні типи додатків: веб, мобільний і десктопний софт, ігри і так далі. Традиційно у цієї мови сильні позиції в промисловому програмуванні, в сегменті великих компаній.

### 1.2 Структура мови

Для того щоб зрозуміти структуру Java, потрібно спочатку зрозуміти що таке об'єктно-орієнтоване програмування (ООП).

Об'єктно-орієнтоване програмування (скорочено ООП) - це парадигма розробки програмних систем, в якій додатки складаються з об'єктів,, кожен з яких є екземпляром певного класу, а класи утворюють ієрархію спадкування. Об'єкти - це сутності, у яких є властивості і поведінку. Зазвичай об'єкти є екземплярами якого-небудь класу. Поведінка об'єкта визначається за допомогою методів - спеціальних блоків коду, які можна викликати з різних частин програми. Використовуючи ці властивості і методи, можна значно прискорити розробку, зробити код більш читабельним. До того ж самому програмісту простіше складати код, якщо він думає з допомогою об'єктів.

Ідеологічно ООП - підхід до програмування як до моделювання інформаційних об'єктів, вирішальний на новому рівні основне завдання структурного програмування: структурування інформації з точки зору керованості, що істотно поліпшує керованість самим процесом моделювання, що, в свою чергу, особливо важливо при реалізації великих проектів.

Керованість для ієрархічних систем передбачає мінімізацію надмірності даних (аналогічну нормалізації) і їх цілісність, тому створене зручно керованим - буде і зручно розумітися. Таким чином, через тактичну задачу керованості вирішується стратегічне завдання - транслювати розуміння завдання програмістом в найбільш зручну для подальшого використання форму.

Основні принципи структурування в разі ООП пов'язані з різними аспектами базового розуміння предметної завдання, яке потрібно для оптимального управління відповідною моделлю:

-абстракція - це використання тільки тих характеристик об'єкта, які з достатньою точністю представляють його в даній системі. Основна ідея полягає в тому, щоб представити об'єкт мінімальним набором полів і методів і при цьому з достатньою точністю для розв'язуваної задачі.

-інкапсуляція - властивість мови програмування, що дозволяє користувачеві не замислюватися про складність реалізації використовуваного програмного компонента (що у нього всередині), а взаємодіяти з ним за допомогою наданого інтерфейсу (публічних методів і членів), а також об'єднати і захистити життєво важливі для компонента дані. При цьому користувачеві надається тільки специфікація (інтерфейс) об'єкта.

-успадкування - дозволяє описати новий клас на основі вже існуючого (батьківського), при цьому властивості і функціональність батьківського класу запозичуються новим класом. Це дозволяє звертатися з об'єктами класу-спадкоємця точно так же, як з об'єктами базового класу.

-поліморфізм - можливість об'єктів з однаковою специфікацією мати різну реалізацію. Коротко зміст поліморфізму можна виразити фразою: «Один інтерфейс, безліч реалізацій».

При написанні програм використовуються класи і методи. Головна їхня відмінність - в тому, що клас складається з інтерфейсу і реалізації.

Об'єкт - деяка сутність володіє певним станом і поведінкою, що має певні властивості (атрибути) і операції над ними (методи). Як правило, при розгляді об'єктів виділяється те, що об'єкти належать одному або декількох класах, які визначають поведінку (є моделлю) об'єкта. Об'єкти класів, як і дані примітивних типів, можуть передаватися в методи. Однак в даному випадку є одна особливість - при передачі об'єктів в якості значення передається копія посилання на область в пам'яті, де розташований цей об'єкт.

Клас - являє собою шаблон для створення об'єктів, що забезпечує початкові значення станів: ініціалізація полів-змінних і реалізація поведінки функцій або методів. Інші абстрактні типи даних - інтерфейси характеризуються якимись своїми, іншими особливостями. Клас є ключовим поняттям в ООП. На лістингу 1.1 приведений приклад створення класу мовою java, який містить в собі поля і змінні.

Лістинг 1.1 – Приклад створення класу

public class StudentUnivers {

String firstName;

String lastName;

int age;

String nameGroup;

String city;

int course;

void coutStudentInfo() {

System.out.printf("Name: %s \nLast Name: %s \nAge: %d\nGroup: %s\n City: %s\n Course: %d", firstName, lastName, age, nameGroup, city, course);}}

Крім звичайних методів класи можуть визначати спеціальні методи - конструктори. Конструктори викликаються при створенні нового об'єкта даного класу. Конструктори виконують ініціалізацію об'єкта. Якщо в класі не визначено жодного конструктора, то для цього класу автоматично створюється конструктор без параметрів. У лістингу 1.2 приведений приклад класу з наявним конструктором.

Лістинг 1.2 - Приклад класу з наявним конструктором

public class StudentUnivers {

String firstName;

String lastName;

int age;

String nameGroup;

String city;

int course;

public StudentUnivers (String firstName, String lastName, int age, String nameGroup, String city, int course) {

this.firstName = firstName;

this.lastName = lastName;

this.age = age;

this.nameGroup = nameGroup;

this.city = city;

this.course = course; }

void coutStudentInfo() {

System.out.printf("Name: %s \nLast Name: %s \nAge: %d\nGroup: %s\n City: %s\n Course: %d", firstName, lastName, age, nameGroup, city, course);

}}

public static void main(String[] args) {

StudentUnivers Ivan = new StudentUnivers ("Іван","Головко",19,"РТ-218","Николаев",3); }

Класи можуть успадковуватися один від одного. Клас-нащадок отримує всі поля і методи класу-батька, але може доповнювати їх власними або перевизначати вже наявні. Більшість мов програмування підтримує тільки одиничне успадкування (клас може мати тільки один клас-батько), лише в деяких допускається множинне спадкування - породження класу від двох або більше класів-батьків. Множинне успадкування створює цілий ряд проблем, як логічних, так і чисто реалізаційних, тому в повному обсязі його підтримка не поширена. На лістингу 1.3 приведений приклад наслідування класу.

Лістинг 1.3 – Приклад наслідування класів.

class Human {

String firstName;

String lastName;

int age;

Human(String firstName, String lastName, int age){

this.firstName = firstName;

this.lastName = lastName;

this.age = age; }

void coutHumanInfo(){

System.out.printf("Name: %s \nLast Name: %s \nAge: %d\n", firstName, lastName, age);

}}

class StudentUnivers extends Human{

String nameGroup;

String city;

int course;

StudentUnivers (String firstName, String lastName, int age, String nameGroup, String city, int course) {

super(firstName,lastName,age);

this.nameGroup = nameGroup;

this.city = city;

this.course = course;

}

void coutStudentInfo(){

System.out.printf("Name: %s \nLast Name: %s \nAge: %d\nGroup: %s\n City: %s\n Course: %d", firstName, lastName, age, nameGroup, city, course);

}

}

public static void main(String[] args) {

StudentUnivers Ivan = new StudentUnivers ("Іван","Головко",19,"РТ-218","Николаев",3);

}

Інтерфейс - це клас без полів і без реалізації, що включає тільки заголовки методів. Якщо якийсь клас успадковує (або, як кажуть, реалізує) інтерфейс, він повинен реалізувати всі вхідні в нього методи. Використання інтерфейсів надає відносно дешеву альтернативу множинного спадкоємства. Взаємодія об'єктів в абсолютній більшості випадків забезпечується викликом ними методів один одного.

Лістинг 1.4 – Приклад роботи інтерфейсу

public class Test {

public static void main(String[] args) {

B tempObj = new A("Клас А ");

tempObj.printHello();

tempObj.printBye();

tempObj = new B("Клас Б ");

tempObj.printHello();

tempObj.printBye();

}}

interface Interface {

void printHello();

void printBye();}

class A implements Interface {

String name;

A(String name) {

this.name = name; }

public void printHello() {

System.out.printf("%s каже привіт \n", name);

}

public void printBye() {

System.out.printf("%s каже бувай \n", name);

}

}

class B implements Interface {

private String name;

B(String name) {

this.name = name;

}

public void printHello() {

System.out.printf("З вами привітався %s \n", name);

}

public void printBye() {

System.out.printf("З вами попрощався %s \n", name);

}

}

Модифікатори доступу – мають всі члени класу: поля і методи. Модифікатори доступу дозволяють задати допустиму область видимості для членів класу, тобто контекст, в якому можна використовувати цю змінну або метод. Щоб використовувати модифікатор в Java, потрібно включити його ключове слово в визначенні класу, методу або змінної. Модифікатор повинен бути попереду решти операторів.

В Java використовуються наступні модифікатори доступу:

- public : публічний, загальнодоступний клас або член класу. Поля і методи, оголошені з модифікатором public, видно з інших класів поточного пакета і з зовнішніх пакетів.

-private : закритий клас або член класу, протилежність модифікатору public. До змінної, методу або класу, позначеного модифікатором private , можна звертатися тільки з того ж класу, де він оголошений. Для всіх інших класів позначений метод або змінна - невидимі. Це найвища ступінь закритості - тільки свій клас. Такі методи не успадковуються і не перевизначаються. Доступ до них з класу-спадкоємця також неможливий.

-protected : такий клас або член класу доступний в межах всіх класів, що знаходяться в тому ж пакеті, що і наш або в межах всіх класів-спадкоємців нашого класу. В основному даний модифікатор доступу використовують при наслідуванні класів.

-Модифікатор за замовчуванням. Якщо змінна або метод не позначені ніяким модифікатором, то вважається, що вони позначені «модифікатором за замовчуванням». Змінні і методи з таким модифікатором видно всіх класах пакету, в якому вони оголошені, і тільки їм. Цей модифікатор ще називають « package » або « package private », натякаючи, що доступ до змінних і методів відкритий для всього пакету, в якому знаходиться їх клас

Більш детально розібратися з специфікаторами доступу допоможе таблиця 1.1 на якій зображено те який мають доступ змінні методи та класи у різних ситуаціях.

Таблиця 1.1 – Специфікатори доступу

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Тип видимості | Ключове слово | Доступ | | | |
| Свій клас | Свій пакет | Клас - спадкоємець | Інші класи |
| Закритий | private | є | немає | немає | немає |
| Пакет | (Немає модифікатора) | є | є | немає | немає |
| Захищений | protected | є | є | є | немає |
| Відкритий | public | є | є | є | є |

Модифікатори класу, методу, змінної і потоку, які використовуються не для доступу. Java надає ряд модифікаторів не для доступу, а для реалізації багатьох інших функціональних можливостей:

- модифікатор static застосовується для створення методів і змінних класу. Ключове слово static використовується для створення змінних, які будуть існувати незалежно від будь-яких екземплярів, створених для класу. Тільки одна копія змінної static в Java існує незалежно від кількості екземпляру класу. Статичні змінні також відомі як змінні класу. В Java локальні змінні не можуть бути оголошені статичними. Ключове слово static при використанні для створення методів, які будуть існувати незалежно від будь-яких екземплярів, створених для класу. В Java статичні методи або методи static не використовують будь-які змінні екземпляра будь-якого об'єкта класу, вони визначені. Методи static приймають всі дані з параметрів і щось з цих параметрів обчислюється без посилання на змінні. Змінні і методи класу можуть бути доступні з використанням імені класу, за яким слідує крапка і ім'я змінної або методу.

- модифікатор final використовується для завершення реалізації класів, методів і змінних. Зміна final може бути ініціалізована тільки один раз. Посилальна змінна, оголошена як final, ніколи не може бути призначена для позначення іншого об'єкта. Однак дані всередині об'єкта можуть бути змінені. Таким чином, стан об'єкта може бути змінено, але не посилання. При створенні змінних в Java модифікатор final часто використовується з static, щоб зробити константою змінну класу. Метод final не може бути перевизначений будь-яким підкласом. Як згадувалося раніше, в Java модифікатор final захищає метод від змін в підкласі. Головним наміром зробити метод final буде те, що зміст методу не повинно бути змінено на стороні.

- модифікатор abstract необхідний для створення абстрактних класів і методів. Клас з таким модифікатором не може створити екземпляр. Якщо клас оголошений як abstract, то єдина мета для нього бути розширеним за допомогою наслідування. Клас не може бути одночасно abstract і final, так як клас final не може бути розширеним. Якщо клас містить абстрактні методи, то він повинен бути оголошений як abstract. В іншому випадку буде згенеровано помилка компіляції. Клас abstract може містити як абстрактні методи, а також і звичайні. Метод abstract є методом, оголошеним з будь реалізацією. Тіло методу (реалізація) забезпечується підкласом. Будь-клас, який розширює абстрактний клас повинен реалізувати всі абстрактні методи суперкласу, якщо підклас не є абстрактним класом.

### 1.3 Пакети

Пакет Java - є спеціальним бібліотечним модулем, який містить групу класів, об'єднаних в одному просторі імен. Їх використовуються для запобігання конфліктів з назвами, для контролю доступу, для полегшення пошуку / знаходження і використання класів, інтерфейсів. Приклад правильного звернення до класу: «ім'я пакета» + «ім'я класу».

Щоб вказати, що клас належить певному пакету, треба використовувати директиву package, після якої вказується ім'я пакета: «package назва\_пакета;»

Як правило, назви пакетів відповідають фізичній структурі проекту, тобто організації каталогів, в яких знаходяться файли з вихідним кодом. А шлях до файлів всередині проекту відповідає назві пакета цих файлів.

Класи необов'язково визначати в пакети. Якщо для класу пакет не визначений, то вважається, що даний клас знаходиться в пакеті за замовчуванням, який не має імені.

Імпорт пакетів - Якщо нам треба використовувати класи з інших пакетів, то нам треба підключити ці пакети і класи. Наприклад використовуємо клас Scanner який знаходиться в пакеті java.util для отримання даний вводяться в консоль. Що б імпортувати пакет ми прописуємо: «import java.util.Scanner;», тобто ми вказуємо повний шлях до файлу в пакеті, і створюємо об'єкт «Scanner in = new Scanner (System.in);» за допомогою якого зможемо викликати функцію nextInt (), яка забирає введені значення з консолі. Імпортувати можна необмежену кількість пакетів в програмований проект.

Основні пакети які поставляються в java:

java.lang - базова функціональність мови та основні типи

java.util - включає велику кількість допоміжних класів, широко використовуваних в інших вбудованих пакетах Java. Ці класи розташовані в пакеті java.util і використовуються для роботи з набором об'єктів, взаємодії з системними функціями низького рівня, для роботи з математичними функціями, генерації випадкових чисел і маніпуляцій з датою і часом

java.io - містить майже кожен клас, який може знадобитися для здійснення введення і виведення в Java. Всі дані потоки представлені потоком введення і адресою виведення. Потік в пакеті java.io здійснює підтримку різних даних, таких як примітиви, об'єкти, локалізовані символи і т.п;

java.nio - новий фреймворк для прикладних програмних інтерфейсів мови Java, призначених для реалізації високопродуктивних операцій введення-виведення;

java.net - операції з мережами, сокетами, DNS-запитів підтримує сімейства протоколів TCP, UDP і т.д. У складі пакету java.net є безліч класів, пов'язаних з мережею;

java.security - генерація ключів, шифрування і дешифрування;

java.sql - Java Database Connectivity (JDBC) для доступу до баз даних;

javax.swing - ієрархія пакетів для переносних GUI компонентів;

java.awt - ієрархія основних пакетів для рідних компонентів GUI надає технологію AWT для створення графічного (віконного) інтерфейсу користувача – GUI;

Для того що б створити свій власний пакет, потрыбно вибрати йому ім'я і включити оператор package разом з цим ім'ям поверх кожного вихідного файлу, який містить класи, інтерфейси, перерахування та типи анотацій, які ви хочете включити в пакет.

Оператор package повинен бути першим рядком у вихідному файлі. Може бути тільки один запит package в кожному вихідному файлі, і він застосовується до всіх типів в цьому файлі. Якщо оператор package не використано, тоді класи, інтерфейси, перерахування та типи анотацій будуть поміщені в поточний пакет за замовчуванням.

Пакети повинні розташовуватися у відповідних директоріях, тобто файл пакета (з ім'ям ІмяПакета) повинен бути збережений в папці ІмяПакета.

Вони можуть бути вкладеними (наприклад, java.util, де java- головний пакет, autil - вкладений). Відповідно ці два пакети повинні розташовуватися в каталозі java \ util.

## 2 Класс Math

### 2.1 Інформація про клас

Пакет java.lang містить класи, які є базовими в мові Java. Цей пакет містить класи, які є фундаментальними для дизайну мови програмування Java. Найважливіші класи Object, який є коренем ієрархії класу, і Class, зразки якого представляють класи під час виконання. Цей пакет також містить потрібний нам клас Math який реалізує набір базових математичних функцій. Даний клас є корисним, коли потрібно: обчислити абсолютні значення, значення тригонометричних функцій, зведення в різні ступені, витяг коренів різних ступенів, генерація випадкових чисел та іншими математичних числень. Методи реалізовані як static, тому можна відразу викликати через Math.methodName () без створення екземпляра класу. Даний клас містить дві константи: Е (константа Ейлера) і PI (число я). А тепер більш докладно про методи які містить цей (вони все статичні).

### 2.2 Тригонометричні функції

Тригонометричні функції є елементарні функції, аргументом яких є кут. За допомогою тригонометричних функцій описуються співвідношення між сторонами і гострими кутами в прямокутному трикутнику. Області застосування тригонометричних функцій надзвичайно різноманітні. Так, наприклад, будь-які періодичні процеси можна представити у вигляді суми тригонометричних функцій (ряду Фур'є). Дані функції часто з'являються при вирішенні диференціальних і функціональних рівнянь.

Всі функції класу які працюють з кутами, приймають його тільки в радіанах. Щоб перетворити його з градусів в радіани і назад, в розглянутому класі є дві спеціальні функції: toRadians () - перетворює кут, вимірюваний в градусах, в еквівалентний кут, вимірюваний в радіанах, toDegrees () - перетворює кут, вимірюваний в радіанах, в приблизно еквівалентний кут, вимірюваний в градусах.

Основні методи використовуються для розрахунку кутів:

sin () - Повертає синус кута

cos () - Повертає косинус кута

tan () - Повертає тангенс кута

asin () - Повертає арксинус

acos () - Повертає арккосинус

atan () - Повертає арктангенс

sinh () - Повертає гіперболічний синус

cosh () - Повертає гіперболічний косинус

tanh () - Повертає гіперболічний тангенс

Два самих фундаментальних елемента в математиці - це «е» (підстава натурального логарифма) і «пі» (відношення довжини кола до його діаметра). Ці дві константи часто потрібні в розрахунках. Отже в класі Math є ці дві змінні-константи: Math.PI- число «Пі» рівне 3.1415 і Math.E число «Е» рівне 2.718

### 2.3 Алгебраїчні функції

У більшості випадків всі ці методи можна реалізувати самостійно, але можна скористатися методами даного класу в які потрібно передати тільки значення і отримати результат, тим самим заощадивши свій час на реалізацію і заощадивши місце в коді.

Основні алгебраїчні функції прописані в класі Math:

sqrt () - квадратний корінь з аргументу

cbrt () - кубічний корінь з аргументу

pow () - приймає 2 числа, перше-це саме число і друге- це ступінь в яку потрібно звести.

exp () - експонента, повертає натуральний логарифм за основою e і аргументу - показником ступеня

log () - повертає натуральний логарифм аргументу

### 2.4 Інші функції

Також клас по мимо описаних тригонометричних і алгебраїчних функцій так само має кілька функцій які створені для роботи з числами: їх округлення в одну з двох сторін, знаходження більшого і меншого числа і генерації випадкового числа.Список цих функцій:

Math.rint () - повертає ціле число, яке найближчим за значенням аргументу, іншими словами - округлює дріб до цілого числа

Math.floor () - дає ціле число з нульовою дробовою частиною, найближчим до числа аргументу зліва, іншими словами - округлює дріб в меншу сторону

Math.abs () - дає абсолютне значення аргументу, простими словами - ми отримуємо модуль числа

Math.ceil () - дає ціле число з нульовою дробовою частиною, найближчим до числа аргументу справа, іншими словами - округлює дріб в більшу сторону.

Math.min () - повертає мінімальне значення з двох аргументів

Math.max () - повертає максимальне значення з двох аргументів

Math.random () - в Java використовується для генерації випадкового числа в діапазоні від 0.0 до 1.0. Різні діапазони можуть бути досягнуті за допомогою арифметики

## 3 Середовище розробки Android Studio

### 3.1 Історія Android Studio

Ринок мобільних додатків збільшується фактично в геометричній прогресії, кожен день з'являються нові мобільні додатки.

Так само, як і для створення картинок, відеороликів або будь-яких інших продуктів, для створення додатків необхідне власне інтегроване середовище розробки (IDE). Ще кілька років тому найпопулярнішим середовищем був безкоштовний Eclipse, але зараз гідну йому конкуренцію становить порівняно нова IDE Android Studio.

Android Studio - інтегроване середовище розробки (IDE) для роботи з платформою Android, анонсована 16 травня 2013 року на конференції Google I / O.

Дана IDE перебувала у вільному доступі починаючи з версії 0.1, опублікованій в травні 2013, а потім перейшла в стадію бета-тестування, починаючи з версії 0.8, яка була випущена в червні 2014 року. Перша стабільна версія 1.0 була випущена в грудні 2014 року, тоді ж припинилася підтримка плагіна Android Development Tools (ADT) для Eclipse.

Android Studio, заснована на програмному забезпеченні IntelliJ IDEA від компанії JetBrains, - офіційний засіб розробки Android додатків. Дане середовище розробки доступне для Windows, macOS і GNU / Linux. 17 травня 2017, на щорічній конференції Google I / O, Google анонсував підтримку мови Kotlin, використовуваного в Android Studio, як офіційної мови програмування для платформи Android на додаток до Java і С ++.

Завдяки вбудованому емулятора можна проводити тести продуктивності і коректності роботи що розробляються на різних системах, і в разі необхідності проводити оптимізацію. З 2014 року Android Studio є офіційним середовищем розробки продуктів під систему Android.

IDE можна завантажити і користуватися безкоштовно. У ній присутні макети для створення UI, з чого зазвичай починається робота над додатком. В Studio містяться інструменти для розробки рішень для смартфонів і планшетів, а також нові технологічні рішення для Android TV, Android Wear, Android Auto, Glass і додаткові контекстуальні модулі.

Середовище Android Studio призначене як для невеликих команд розробників мобільних додатків (навіть в кількості однієї людини), або ж великих міжнародних організацій з GIT або іншими подібними системами управління версіями. Досвідчені розробники зможуть вибрати інструменти, які більше підходять для масштабних проектів. Рішення для Android розробляються в Android Studio з використанням Java або C ++. В основі робочого процесу Android Studio закладений концепт безперервної інтеграції, що дозволяє відразу ж виявляти наявні проблеми. Тривала перевірка коду забезпечує можливість ефективного зворотного зв'язку з розробниками. Така опція дозволяє швидше опублікувати версію мобільного застосування в Google Play App Store. Для цього є також підтримка інструментів LINT, Pro-Guard і App Signing.

За допомогою засобів оцінки продуктивності визначається стан файлу з пакетом прикладних програм. Візуалізація графіка дає можливість дізнатися, чи відповідає додаток орієнтиру Google в 16 мілісекунд. За допомогою інструменту для візуалізації пам'яті розробник дізнається, коли його застосування буде використовувати занадто багато оперативної пам'яті і коли відбудеться «прибирання сміття». Інструменти для аналізу батареї показують, яке навантаження припадає на пристрій.

Android Studio сумісна з платформою Google App Engine для швидкої інтеграції в хмарі нових API і функцій. У середовищі розробки ви знайдете різні API, такі як Google Play, Android Pay і Health. Є підтримка всіх платформ Android, починаючи з Android 1.6. Є варіанти Android, які істотно відрізняються від версії Google Android. Найпопулярніша з них - це Amazon Fire OS. В Android Studio можна створювати APK для цієї ОС. Підтримка Android Studio обмежується онлайн-форумами.

### 3.2 Основні можливості середовища розробки

Android studio включає різний інструментарій, що забезпечує комфортні умови роботи на різних етапах розробки програми. Вам не потрібно кожен раз упаковувати програму, а потім встановлювати на смартфон або планшет, що б подивитися результати виконаної роботи. Більш того, вам не потрібно мобільний пристрій. Вбудований Емулятор Андроїд дозволяє подивитися, як працює додаток в потрібних для вас умовах. Інструмент аналізатор APK дозволяє переглянути кількість і розмір файлів майбутньої програми. Так само є паралельне порівняння двох пакетів, наприклад старої і нової версії.

Для створення і настройки призначеного для користувача інтерфейсу, доступний інструмент - Редактор макетів. Зміна розміру або положення елементів, виконується простим перетягуванням.

Можливості середовища розробки:

- Збірка додатків, заснована на Gradle .

- Різні види збірок і генерація кількох .apk файлів

- Рефакторинг коду

- Вбудований ProGuard і утиліта для підписування додатків.

- Підтримка розробки додатків для  пристроїв які працюють на ОС Android .

- Шаблони основних макетів і компонентів UI, з чого зазвичай починається робота над додатком.

- Розроблену програму можна запустити на пристрої за домогою usb кабелю.

Додаток створюваний в рамках даної роботи, буде розроблятися в Android Studio. Вважається, що це одна з кращих середовищ розробки програмного забезпечення для Android-пристроїв на сьогоднішній день. Вона є офіційною середовищем розробки від Google, і при цьому є безкоштовною. Середовице Android Studio надає програмісту безліч можливостей при розробці, наприклад, наявність власного емулятора, відладчика, готові шаблони, дозволяє бачити зміни в реальному часі за допомогою емулятора пристроїв, а також багато інших інструментів.

## 4 Практична частина

### 4.1 Постановка задачі

Створити програму, яка виконує обчислення в залежності від того в які були внесені вхідні данні та після підрахунку відображала результат обчислення. Також повинна мати графічний інтерфейс за допомогою якого буде виконуватись введення та виведення результату. Даний додаток повинен працювати на пристроях з операційною системою Android.

Ця програма повинна виконувати підрахунки основних тригономертичних функцій, квадратних рівнянь, деяких простих математичних виразів та генерації випадкових чисел. Не менш важливим є її точність підрахунку та оптимізація коду. Однією з головних вимог є те шо користувацький інтерфейс повинен бути простим для сприйняття та оптимізованим під різні мобільні пристрої.

### 4.2 Складання технічного завдання на розробку додатку.

Розроблювальний додаток повинен бути зручним в користуванні. На робочій поверхні екрану повинні бути розташовані такі елементи: поля для вводу виводу інформації, кнопки керування, за допомогою яких буде виконуватись обчислення та очищення полів для вводу даних.

Екран повинен бути поділений на блоки. У цих блоках знаходяться поля вводу та виводу які відносяться до певних підрахунків тема яких написана на початку блока та кнопка за допомогою якої данні беруться з цих полів та відправляються на обробку мовою java та класом Math.

Перший блок складається з основних формул тригонометрії, тобто користувач вводить радіани або градуси та натискає кнопку «Розрахувати». В залежності від того яка одиниця виміру буда введена, розраховується протилежна, тобто якщо були введені радіани то вони конвертуються в градуси та навпаки, потім виконується розрахунок значень: синуса, косинуса, тангенса, котангенса, синус подвійного кута, косинус подвійного кута, тангенс подвійного кута, котангенс подвійного кута, синус квадратний та косинус квадратний.

Другий блок відповідає за генерацію випадкового числа. У відповідні поля вводяться діапазони генерації та натискається кнопка для генерації цього числа.

Третій блок підраховує число в ступені, у перше поле потрібно ввести саме число, а у друге необхідно ввести ступінь в яку потрібно звести число, потім за допомогою натискання на кнопку, отримуємо число зведене у вказану ступінь.

Четвертий блок обчислює квадратний корень числа введеного в поле.

П'ятий блок виконує знаходження коренів квадратного рівняння, у поля вводяться три змінних, натискається кнопка «Розв'язати» та виводяться розв'язок цього рівняння, а саме корені рівняння та дискримінант.

І останньою у цьому списку блоків знаходиться кнопка яка буде очищувати всі поля які відповідають за ввод даних, розроблена вона для того щоб не витрачати час на очищення цих полів.

### 4.3 Опис функцій на методів що використовувались при розробці

При розробці додатку для програмування було використано середовище розробки Android Studio за допомогою якого можна робити додатки для платфоми Android.

Перед тим як починати писати логіку Android додатку потрібно створити графічний інтерфейс за допомогою якого буде виконуватись взаємодія людини з програмою.

Інтерейс було поділено на блоки для того щоб користувачу було зручно бачити границі тих частин, в яких знаходяться поля для розв'язку формул певних тем. Повний код розробленої програми приведено в додатках А.1 та А.2

Для кращого розуміння побудови інтерфейсу, розглянемо будову блоку який відповідає за розв'язок квадратного рівняння. На прикладі коду мови xml (лістинг 4.1). В xml-файлі описано у вигляді xml-коду розташування та візуалізація об'єктів. При запуску активності система Android автоматично розпізнає розмір екрану мобільного пристрою і призводить виведений контент у відповідність з розміткою, описаної в xml-файлі. Таким чином, одна і та ж активність буде виглядати однаково незалежно від діагоналі використовуваного пристрою. На рисунку 4.1 зображено те як цей блок виглядає на пристрої.

Основною частиною інтерфейсу є компонент **LinearLayout** він відноситься до **view group** , яка вирівнює всі дочірні елементи в одному напрямку, вертикально або горизонтально. Орієнтацію макета можна задати за допомогою атрибута **android: orientation**. Весь додаток було побудовано на цьому типі макету горизонтальної орієнтації, яка в свою чергу містить в собі макети цього ж типу, але для зручності відображення орієнтація була обрана вертикального типу.

Також **LinearLayout**  підтримує присвоєння ваги за допомогою атрибута **android:layout\_weight**. Цей атрибут присвоює значення «важливості» **view** з точки зору того, скільки місця він повинен займати на екрані. Більше значення ваги дозволяє йому розширюватися, заповнюючи весь простір в батьківському **view** . Дочірні **view** можуть встановлювати значення ваги, займаючи місце у**view group**  пропорційно їх вазі. Вага за замовчуванням - нуль. Цей атрибут використовувався в блоці розв'язку квадратного рівняння, для того щоб зменшити текстове поле с текстом «Результат», та збільшити поле для самого результату.

За допомогою компоненту TextView на екрані можна виводити різну інформацію, наприклад в цьому блоці за його допомогою виводяться його назва та результати обчислення. Реалізовано це таким чином: ввесь текст який виводиться на екран пристрою показується за допомогою цього компоненту, коли програма виконує обчислення та вже має результат розв'язку, тоді за допомогою функції setText присвоює це значення у заготовлене пусте поле TextView. Особливістю полів цього типу э те що воно лише відображає текст та не дає можливість його змінювати або копіювати.

За допомогою компоненту EditText користувач може вносити значення які програма повинна буде обчислити та вивести результат у TextView. Ці поля бувають різного типу: простий текст, електрона пошта, пароль, телефон, багаторядковий текст та числовий. У даному додатку був обраний EditText числового типу, бо для підрахунку формул нам не потрібні букви, тому їх можна заборонити. Взагалі EditText є підкласом класу TextView. Він також представляє текстове поле, але тепер уже з можливістю введення і редагування тексту. Таким чином, в EditText ми можемо використовувати всі ті ж можливості, що і в TextView.

Одним з часто використовуваних елементів є кнопки, які представлені класом android.widget.Button. Ключовою особливістю кнопок є можливість взаємодії з користувачем через натискання. Кнопка складається з тексту, який повідомляє користувачеві, яка дію відбудеться після її натискання. За допомогою компоненту Button користувач викликає java функцію, яка забирає значення з певних EditText полів та починає виконувати обчислення запрограмовані в цій функції та виводить результат у певний TextView компонент.

Лістинг 4.1- Код що описує розташування компонентів на екрані

<LinearLayout android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="horizontal">

<EditText android:id="@+id/edTextDiscr1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="3"

android:hint="a"

android:inputType="numberDecimal" />

<EditText android:id="@+id/edTextDiscr2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="3"

android:hint="b"

android:inputType="numberDecimal" />

<EditText android:id="@+id/edTextDiscr3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="3"

android:hint="c"

android:inputType="numberDecimal" />

<Button android:id="@+id/btnDiscr"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Розрахувати"/> </LinearLayout>

<LinearLayout android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginLeft="10dp"

android:orientation="horizontal">

<TextView android:id="@+id/textDiscrR"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="3"

android:text="Результат: " />

<TextView android:id="@+id/textDiscrRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

</LinearLayout>

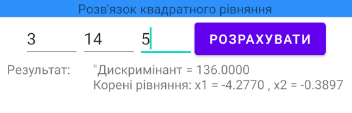


Рисунок 4.1 – Вигляд блоку який відповідає за розв'язок квадратного рівняння

Тепер розглянемо додаток з програмного боку. Для точного розрахунку всіх функцій використовується клас Math мови JAVA що поставляється в пакеті java.lang. Подивимось на код того самого блоку що вирішує квадратні рівняння. Для розв’язку даного рівняння майже достатньо стандартних математичних операцій, але для розрахунку квадратного кореню числа нам приходить на допомогу функція sqrt, яка реалізована у класі Math.

Лістинг 4.2 – Код функції що відповідає за розв'язок квадратного рівняння

public void discrim(TextView textView, double a, double b, double c)

{

double D;

D = b \* b - 4 \* a \* c;

if (D > 0) {

double x1, x2;

x1 = (-b - Math.sqrt(D)) / (2 \* a);

x2 = (-b + Math.sqrt(D)) / (2 \* a);

textView.setText(String.format("\"Дискримінант = %.4f \nКорені рівняння: x1 = %.4f , x2 = %.4f",D,x1, x2));

}

else if (D == 0)

{

double x;

x = -b / (2 \* a);

textView.setText(String.format("\"Дискримінант = %.4f \nРівняння має лише один корень: x = %.4f , x2 = %.4f",D,x));

}

else

{

textView.setText(String.format("\"Дискримінант = %.4f \nРівняння не має дійсних коренів",D));

}

}

У блоці відповідальному за основні тригонометричні функції були два поля які приймають вхідні данні- це радіани та градуси. Якщо користувач вводить у поле радіанів то додаток демонструє це число в одиницях виміру градусів, і навпаки, при введенні градусів розраховується число у радіанах. Потім після цих операцій починається виконання основної частини коду відповідальної за цю тему, а саме розрахунок значень: синуса, косинуса, тангенса, котангенса, синус подвійного кута, косинус подвійного кута, тангенс подвійного кута, котангенс подвійного кута, синус квадратний та косинус квадратний. При їх розрахунку використовувався клас Math та такі його функції: sin, cos, tan, toDegrees, toRadians, pow, round. Також в деяких розрахунках виконувалось комбінування цих функцій.

Як приклад можна привести розрахунок косинуса подвійного кута, де використовувалось комбінування трьох функцій цього класу «Math.pow(Math.cos(radian), 2) - Math.pow(Math.sin(radian), 2)». При його розрахунку було використано такі функції класу Math: cos- повертає значення косинуса вказаного в радіанах або градусах, sin- працює аналогічно cos та повертає значення синуса вказаного в радіанах або градусах та функція pow – що виконує піднесення до степеню будь-яких значень.

І останнім не описаним блоком залишається той шо генерує випадкове число за допомогою функції rand. Генерація виконується у діапазоні, який було вказано в полях цього блоку.

Також не мало важливою фішкою додатку є кнопка очищення яка розташована внизу всього інтерфейсу за допомогою якої можна очистити всі поля для вводу, тим самим це робить інтерфейс зручним, тому що людині не потрібно витрачати час на безглузді дії. Код функції яка викликається при натисканні на цю кнопку приведено у лістингу 4.3. За допомогою функції getChildCount отримуємо кількість компонентів які розташовані на екрані. Функція getChildAt дає можливість за допомогою індексу пройтися по всім компонентам, за допомогою оператору if виконується перевірка, чи відноситься переглядаємий компонент в циклі до компоненту типа EditText. Якщо він являється таким то за допомогою функції setText присвоюємо йому значення пустого рядка що фактично означає що не потрібно нічого відображати у цих полях. Ось так завдяки невеликої кількості строк коду можна зекономити не мало часу на таких простих діях.

Лістинг 4.3 Код функції очищення EditText полів

private void clearForm(ViewGroup group)

{

for (int i = 0, count = group.getChildCount(); i < count; ++i)

{

View view = group.getChildAt(i);

if (view instanceof EditText)

{

((EditText) view).setText("");

}

if (view instanceof ViewGroup && (((ViewGroup) view).getChildCount() > 0))

{

clearForm((ViewGroup) view);

}

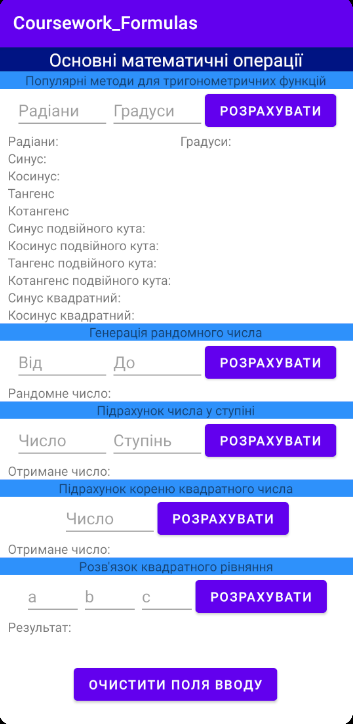
}

}

}

### 4.4 Огляд можливостей розробленого додатку

Огляд почнеться с повного зображення додатку на рисунку 4.2показано те як додаток виглядає при першому запуску.



4.2 **–** Вигляд додатку при першому запуску.

Тепер зупинимось більш детально на кожному з них. Перший на черзі блок який вираховує популярні тригонометричні функції. На рисунку 4.3 зображено як виглядає пустий блок та з розрахованими за його допомогою даними.

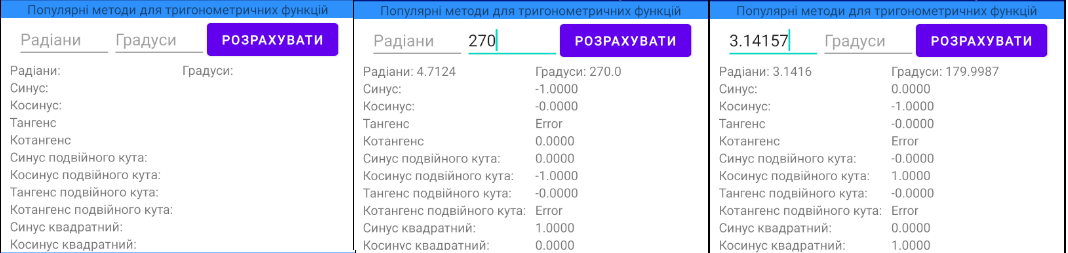
****

Рисунок 4.3 – Демонстрація розрахунку тригонометричних функцій

Наступним йде блок генерації випадкового числа. На рисунку 4.4 зображено те як він виглядає з незаповненими полями, та з готовим згенерованим числом.

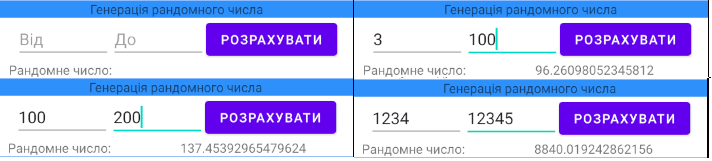
****

Рисунок 4.4 – Демонстрація блоку генерації випадкового числа

Після блоку генерації йдуть блоки підрахунку числа у ступені та підрахунку кореню квадратного заданого числа. На рисунку 4.5 зображено те як виглядають ці два блоки у пустому вигляді та з готовими розрахунками.

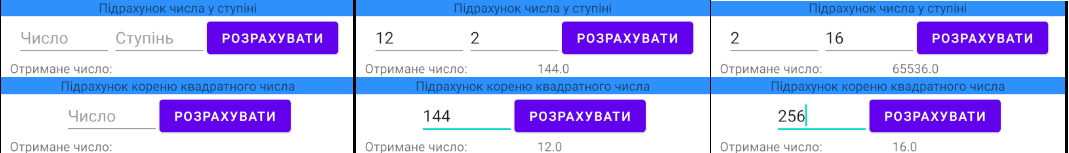


Рисунок 4.5 – Демонстрація блоків підрахунку числа у ступені та підрахунку кореню квадратного заданого числа

І останнім блоком залишається той що відповідає за розв'язок квадратного рівняння. На рисунку 4.6 зображено те як він виглядає з незаповненими полями, та з готовим розрахованим дискримімантом і корнями.

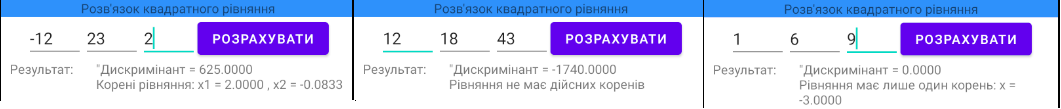


Рисунок 4.6 – Демонстрація блоку розв'язку квадратного рівняння

### Висновок

У курсовому проекті було проведено ознайомлення з середою розробки Android Studio та класом Math який постачається з пакетом java.lang та використовується для точних підрахунків. Результатом цього проекту було розроблено додаток за допомогою якого можна вираховувати основні тригонометричні операції, розв’язувати квадратні рівняння, генерувати випадкове число та деякі інших математичні операції. Програма не займає багато місця, не вимоглива до встановленого програмного забезпечення.

Перевагами даного додатку є:

· Можливість скидати результат;

· Можливість запустити на будь-якому Android пристрої бо продукт маловимогливий до системних ресурсів;

Недоліком є неможливість виконання математичних операцій які не були прописані у коді при програмуванні. Але це і мінусом вазко назвати, бо у будь який час невистачаючу формулу можна додати до розробленого додатку.

В цілому, поставлена ​​на початку курсового проекту мета була досягнута. У програмі виконуються всі необхідні функціональні вимоги щоб продемонструвати всі можливості класу Math.

### Перелік джерел посилань

1. Джеймс Гослінг, Білл Джой, Гай Стіл, Гілад Брача, Алекс Баклі. Мова програмування Java SE 8. Детальний опис, 5-е видання = The Java Language Specification, Java SE 8 Edition (5th Edition) (Java Series). - М. : «Вільямс» , 2015. - 672 с. - ISBN 978-5-8459-1875-8 .
2. Брюс Еккель. Філософія Java = Thinking in Java. - 4-е изд. - СПб. : Пітер , 2018. - 1168 с. - ISBN 978-5-496-01127-3 .
3. Герберт Шилдт. Полный справочник по Java SE 6 = Java: The Complete Reference. — 7-е изд. — М.: «Вильямс», 2007. — С. 1040. — ISBN 0-07-226385-7.
4. Кей С. Хорстманн. Java. Бібліотека професіонала, тому 2. Розширені засоби програмування. 10-е видання = Core Java. Volume II - Advanced Feature (Tenth Edition). - М. : «Вільямс» , 2017. - 976 с. - ISBN 978-5-9909445-0-3 .
5. Бенджамін Дж. Еванс, Джеймс Гоф, Кріс Ньюленд. Java: оптимізація програм. Практичні методи підвищення продуктивності додатків в JVM. - М .: Діалектика, 2019. - 448 с. - ISBN 978-5-907114-84-5.
6. Герберт Шілдт. Java. Повне керівництво, 10-е видання = Java. The Complete Reference, 10th Edition. - М .: «Діалектика», 2018. - 1488 с. - ISBN 978-5-6040043-6-4.Роджерс Р., Ломбардо Д. Android. Розробка додатків. - М. : ЕКОМ Паблишерз , 2010. - 400 с.- ISBN 978-5-9790-0113-5.
7. Голощапов А. Google Android: програмування для мобільних пристроїв. - СПб. : БХВ-Петербург , 2010. - 448 с. - ISBN 978-5-9775-0562-8 .
8. Донн Фелкер. Android: розробка додатків для чайників = Android Application Development For Dummies. - М. : Діалектика , 2011. - 336 с. - ISBN 978-5-8459-1748-5 .
9. Іан Соммервілла. Інженерія програмного забезпечення = Software Engineering. - 6-е изд. - М. : «Вільямс» , 2002. - С. 642. - ISBN 5-8459-0330-0 .

### Додаток А

Лістинг А.1 – Код файлу activity\_main.xml

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/Main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

tools:context=".MainActivity">

<ScrollView

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="vertical">

<TextView

android:id="@+id/textView2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#061783"

android:gravity="center"

android:text="Основні математичні операції"

android:textColor="@color/white"

android:textSize="20dp" />

<TextView

android:id="@+id/textView3"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#2E91FB"

android:gravity="center"

android:text="Популярні методи для тригонометричних функцій" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="horizontal">

<EditText

android:id="@+id/edTextRadian"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="5"

android:hint="Радіани"

android:inputType="numberSigned" />

<EditText

android:id="@+id/edTextDegrees"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="5"

android:hint="Градуси"

android:inputType="numberSigned" />

<Button

android:id="@+id/btnTrigon"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Розрахувати"/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginLeft="10dp"

android:orientation="vertical">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvRad"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Радіани: " />

<TextView

android:id="@+id/tvDeg"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Градуси: " />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvSin"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Синус: " />

<TextView

android:id="@+id/tvSinRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvCos"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Косинус:" />

<TextView

android:id="@+id/tvCosRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvTan"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Тангенс" />

<TextView

android:id="@+id/tvTanRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvCtan"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Котангенс " />

<TextView

android:id="@+id/tvCtanRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvSin2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Синус подвійного кута: " />

<TextView

android:id="@+id/tvSin2Res"

android:layout\_width="match\_parent" android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvCos2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Косинус подвійного кута: " />

<TextView

android:id="@+id/tvCos2Res"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvTg2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Тангенс подвійного кута:"/>

<TextView

android:id="@+id/tvTg2Res"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvCtg2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Котангенс подвійного кута: " />

<TextView

android:id="@+id/tvCtg2Res"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvSin\_2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Синус квадратний: " />

<TextView

android:id="@+id/tvSin\_2Res"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="horizontal">

<TextView

android:id="@+id/tvCos\_2"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Косинус квадратний:"/>

<TextView

android:id="@+id/tvCos\_2Res"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<TextView

android:id="@+id/textView5"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#2E91FB"

android:gravity="center"

android:text="Генерація рандомного числа" />

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="horizontal">

<EditText

android:id="@+id/edTextRandNum1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="5"

android:hint="Від"

android:inputType="numberSigned" />

<EditText

android:id="@+id/edTextRandNum2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="5"

android:hint="До"

android:inputType="numberSigned" />

<Button

android:id="@+id/btnRand"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Розрахувати"/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginLeft="10dp"

android:orientation="horizontal">

<TextView

android:id="@+id/textRandR"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Рандомне число: " />

<TextView

android:id="@+id/textRandRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<TextView

android:id="@+id/textView6"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#2E91FB"

android:gravity="center"

android:text="Підрахунок числа у ступіні" />

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="horizontal">

<EditText

android:id="@+id/edTextPowNum1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="5"

android:hint="Число"

android:inputType="numberSigned" />

<EditText

android:id="@+id/edTextPowNum2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="5"

android:hint="Ступінь"

android:inputType="numberSigned" />

<Button

android:id="@+id/btnPow"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Розрахувати"/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginLeft="10dp"

android:orientation="horizontal">

<TextView

android:id="@+id/textPowR"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Отримане число: " />

<TextView

android:id="@+id/textPowRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<TextView

android:id="@+id/textView7"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#2E91FB"

android:gravity="center"

android:text="Підрахунок кореню квадратного числа" />

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="horizontal">

<EditText

android:id="@+id/edTextSqrtNum1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="5"

android:hint="Число"

android:inputType="numberDecimal" />

<Button

android:id="@+id/btnSqrt"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Розрахувати"/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginLeft="10dp"

android:orientation="horizontal">

<TextView

android:id="@+id/textSqrtR"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="Отримане число: " />

<TextView

android:id="@+id/textSqrtRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<TextView

android:id="@+id/textView8"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:background="#2E91FB"

android:gravity="center"

android:text="Розв'язок квадратного рівняння" />

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:gravity="center"

android:orientation="horizontal">

<EditText

android:id="@+id/edTextDiscr1"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="3"

android:hint="a"

android:inputType="numberDecimal" />

<EditText

android:id="@+id/edTextDiscr2"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="3"

android:hint="b"

android:inputType="numberDecimal" />

<EditText

android:id="@+id/edTextDiscr3"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:ems="3"

android:hint="c"

android:inputType="numberDecimal" />

<Button

android:id="@+id/btnDiscr"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Розрахувати"/>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginLeft="10dp"

android:orientation="horizontal">

<TextView

android:id="@+id/textDiscrR"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="3"

android:text="Результат: " />

<TextView

android:id="@+id/textDiscrRes"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_weight="1"

android:text="" />

</LinearLayout>

</LinearLayout>

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_marginTop="30dp"

android:gravity="center"

android:orientation="horizontal">

<Button

android:id="@+id/btnClear"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:gravity="center"

android:text="Очистити поля вводу"/>

</LinearLayout>

</LinearLayout>

</ScrollView>

</LinearLayout>

Лістинг 1.2

package com.example.coursework\_formulas;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;

import android.os.Bundle;

import android.view.SoundEffectConstants;

import android.view.View;

import android.view.ViewGroup;

import android.widget.Button;

import android.widget.EditText;

import android.widget.TextView;

import android.widget.Toast;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {

Button btnTrigon, btnRand, btnPow, btnSqrt, btnClear, btnDiscr;

EditText edTextRadian, edTextDegrees, edTextRandNum1, edTextRandNum2, edTextDiscr1, edTextDiscr2, edTextDiscr3, edTextPowNum1, edTextPowNum2, edTextSqrtNum1 ;

TextView tvRad, tvDeg, tvSinRes, tvCosRes, tvTanRes, tvCtanRes, tvSin2Res, tvCos2Res, tvTg2Res, tvCtg2Res, tvSin\_2Res, tvCos\_2Res, textRandRes, textPowRes, textSqrtRes, textDiscrRes;

@Override

protected void onCreate(Bundle savedInstanceState)

{

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

edTextRadian = findViewById(R.id.edTextRadian);

edTextDegrees = findViewById(R.id.edTextDegrees);

edTextRandNum1 = findViewById(R.id.edTextRandNum1);

edTextRandNum2 = findViewById(R.id.edTextRandNum2);

edTextPowNum1 = findViewById(R.id.edTextPowNum1);

edTextPowNum2 = findViewById(R.id.edTextPowNum2);

edTextSqrtNum1 = findViewById(R.id.edTextSqrtNum1);

edTextDiscr1 = findViewById(R.id.edTextDiscr1);

edTextDiscr2 = findViewById(R.id.edTextDiscr2);

edTextDiscr3 = findViewById(R.id.edTextDiscr3);

btnTrigon = findViewById(R.id.btnTrigon);

btnTrigon.setOnClickListener(this);

btnRand = findViewById(R.id.btnRand);

btnRand.setOnClickListener(this);

btnPow = findViewById(R.id.btnPow);

btnPow.setOnClickListener(this);

btnSqrt = findViewById(R.id.btnSqrt);

btnSqrt.setOnClickListener(this);

btnDiscr = findViewById(R.id.btnDiscr);

btnDiscr.setOnClickListener(this);

btnClear = findViewById(R.id.btnClear);

btnClear.setOnClickListener(this);

tvRad = findViewById(R.id.tvRad);

tvDeg = findViewById(R.id.tvDeg);

tvSinRes = findViewById(R.id.tvSinRes);

tvCosRes = findViewById(R.id.tvCosRes);

tvTanRes = findViewById(R.id.tvTanRes);

tvCtanRes = findViewById(R.id.tvCtanRes);

tvSin2Res = findViewById(R.id.tvSin2Res);

tvCos2Res = findViewById(R.id.tvCos2Res);

tvTg2Res = findViewById(R.id.tvTg2Res);

tvCtg2Res = findViewById(R.id.tvCtg2Res);

tvSin\_2Res = findViewById(R.id.tvSin\_2Res);

tvCos\_2Res = findViewById(R.id.tvCos\_2Res);

textRandRes = findViewById(R.id.textRandRes);

textPowRes = findViewById(R.id.textPowRes);

textSqrtRes = findViewById(R.id.textSqrtRes);

textDiscrRes = findViewById(R.id.textDiscrRes);

}

@SuppressLint("SetTextI18n")

public void trigonometry()

{

try {

double radian = 0, degrees = 0;

if ((edTextRadian.getText().toString().trim().isEmpty() && edTextDegrees.getText().toString().trim().isEmpty()) || (!edTextRadian.getText().toString().trim().isEmpty() && !edTextDegrees.getText().toString().trim().isEmpty()))

{

Toast toast = Toast.makeText(getApplicationContext(), "Проверьте введённые значения(Введите только радианы или градусы)", Toast.LENGTH\_SHORT);

toast.show();

clearForm(findViewById(R.id.Main));

} else

{

if (!edTextRadian.getText().toString().trim().isEmpty())

{

radian = Double.parseDouble(edTextRadian.getText().toString());

degrees = Math.toDegrees(radian);

} else if (!edTextDegrees.getText().toString().trim().isEmpty()) {

degrees = Double.parseDouble(edTextDegrees.getText().toString());

radian = Math.toRadians(degrees);

}

resInTextView(tvDeg, roundAvoid(degrees, 4));

tvDeg.setText("Градуси: " + Double.toString(roundAvoid(degrees, 4)));

resInTextView(tvRad, roundAvoid(radian, 4));

tvRad.setText("Радіани: " + Double.toString(roundAvoid(radian, 4)));

resInTextView(tvSinRes, Math.sin(radian));

resInTextView(tvCosRes, Math.cos(radian));

resInTextView(tvTanRes, Math.tan(radian));

resInTextView(tvCtanRes, 1 / Math.tan(radian));

resInTextView(tvSin2Res, 2 \* Math.sin(radian) \* Math.cos(radian));

resInTextView(tvCos2Res, Math.pow(Math.cos(radian), 2) - Math.pow(Math.sin(radian), 2));

resInTextView(tvTg2Res, (2 \* Math.tan(radian)) / (1 - Math.pow(Math.tan(radian), 2)));

resInTextView(tvCtg2Res, (1 / ((2 \* Math.tan(radian)) / (1 - Math.pow(Math.tan(radian), 2)))));

resInTextView(tvSin\_2Res, Math.pow(Math.sin(radian), 2));

resInTextView(tvCos\_2Res, Math.pow(Math.cos(radian), 2));

}

} catch (Exception e)

{

errFunc();

}

}

public void resInTextView(TextView textView, double res)

{

try {

if (res < 10000 && res > -10000) {

textView.setText(String.format("%.4f", res));

} else {

textView.setText("Error");

}

} catch (Exception e) {

errFunc();

}}

public static double roundAvoid(double value, int places)

{

double scale = Math.pow(10, places);

return Math.round(value \* scale) / scale;

}

public void randMath(TextView textView, double a, double b)

{

double random\_number = a + (double) (Math.random() \* b);

textView.setText(Double.toString(random\_number));

}

public void powMath(TextView textView, double a, double b)

{

double pow\_number = Math.pow(a, b);

textView.setText(Double.toString(pow\_number));

}

public void sqrtMath(TextView textView, double a)

{

double sqrt\_number = Math.sqrt(a);

textView.setText(Double.toString(sqrt\_number));

}

public void errFunc()

{

Toast toast = Toast.makeText(getApplicationContext(), "Что-то не так, проверьте введённые значения", Toast.LENGTH\_SHORT);

toast.show();

}

public void discrim(TextView textView, double a, double b, double c){

double D;

D = (b \* b) - (4 \* a \* c);

System.out.println(D);

if (D > 0.0) {

double x1, x2;

x1 = (-b - Math.sqrt(D)) / (2 \* a);

x2 = (-b + Math.sqrt(D)) / (2 \* a);

textView.setText(String.format("\"Дискримінант = %.4f \nКорені рівняння: x1 = %.4f , x2 = %.4f",D,x1, x2));

}

else if (D <0.0)

{

textView.setText(String.format("\"Дискримінант = %.4f \nРівняння не має дійсних коренів",D));

}

else {

double x;

x = -b / (2 \* a);

System.out.println("В нуле");

System.out.println("В нуле");

System.out.println("В нуле");

textView.setText(String.format("\"Дискримінант = %.4f \nРівняння має лише один корень: x = %.4f",D,x));

}

}

@Override

public void onClick(View v)

{

try {

switch (v.getId()) {

case R.id.btnTrigon:

trigonometry();

break;

case R.id.btnRand:

randMath(textRandRes, Double.parseDouble(edTextRandNum1.getText().toString()), Double.parseDouble(edTextRandNum2.getText().toString()));

break;

case R.id.btnPow:

powMath(textPowRes, Double.parseDouble(edTextPowNum1.getText().toString()), Double.parseDouble(edTextPowNum2.getText().toString()));

break;

case R.id.btnSqrt:

sqrtMath(textSqrtRes, Double.parseDouble(edTextSqrtNum1.getText().toString()));

break;

case R.id.btnDiscr:

discrim(textDiscrRes, Double.parseDouble(edTextDiscr1.getText().toString()), Double.parseDouble(edTextDiscr2.getText().toString()), Double.parseDouble(edTextDiscr3.getText().toString()));

break;

case R.id.btnClear:

clearForm(findViewById(R.id.Main));

break;

}

} catch (Exception e)

{

errFunc();

}

}

private void clearForm(ViewGroup group)

{

for (int i = 0, count = group.getChildCount(); i < count; ++i)

{

View view = group.getChildAt(i);

if (view instanceof EditText)

{

((EditText) view).setText("");

}

if (view instanceof ViewGroup && (((ViewGroup) view).getChildCount() > 0))

clearForm((ViewGroup) view);

}

}

}