ВИЩИЙ НАВЧАЛЬНИЙ ЗАКЛАД УКООПСПІЛКИ
«ПОЛТАВСЬКИЙ УНІВЕРСИТЕТ ЕКОНОМІКИ І ТОРГІВЛІ»

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ БІЗНЕСУ ТА СУЧАСНИХ ТЕХНОЛОГІЙ

ФОРМА НАВЧАННЯ ДЕННА
КАФЕДРА МАТЕМАТИЧНОГО МОДЕЛЮВАННЯ ТА СОЦІАЛЬНОЇ ІНФОРМАТИКИ

Допускається до захисту

Завідувач кафедри _____О.О. Ємець

«_____»_____202_ р.

ПОЯСНЮВАЛЬНА ЗАПИСКА
ДО БАКАЛАВРСЬКОЇ РОБОТИ
на тему
РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ ТРЕНАЖЕРА З ТЕМИ «ТЕОРЕМИ ДОДАВАННЯ ТА МНОЖЕННЯ ЙМОВІРНОСТЕЙ ВИПАДКОВИХ ПОДІЙ» ДИСТАНЦІЙНОГО НАВЧАЛЬНОГО КУРСУ «ТЕОРІЯ ЙМОВІРНОСТЕЙ І МАТЕМАТИЧНА СТАТИСТИКА»

зі спеціальності 122 «Комп'ютерні науки»

Виконавець роботи Дудник Дмитро Анатолійович _____«___»_____2021р.

Науковий керівник к.ф.-м.н., доц., Парфьонова Тетяна Олександрівна
_____ «___»_____2021р.

ПОЛТАВА 2021 р.

РЕФЕРАТ

Обсяг пояснювальної записки: 58 сторінок, з яких основна частина 45 сторінок та 13 сторінок додатку, літературних джерел – 8 назв, рисунків – 16, блок-схем – 3 сторінки.

Предмет роботи: програмне забезпечення тренажера з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика».

Мета роботи: створення програмного забезпечення тренажера з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика».

Методи дослідження: були проаналізовані літературні джерела, інформація з яких була використана у розробці прикладів задач з теми «Теореми додавання та множення ймовірностей випадкових подій», що реалізовані у навчальному тренажері. Також, ця інформація була застосована для створення алгоритму навчального тренажеру, програмного забезпечення розробленого на його основі та аналізу виконаної роботи.

Методи розробки: для реалізації програмного забезпечення було використано програмний каркас Electron та мови JavaScript, HTML і CSS. Для стиснення файлів та пакування проекту було застосовано gulp та electron-builder.

Розроблено програмне забезпечення тренажера з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика».

Ключові слова: НАВЧАЛЬНИЙ ТРЕНАЖЕР, ТЕОРІЯ ЙМОВІРНОСТЕЙ, ТЕОРЕМИ ДОДАВАННЯ ТА МНОЖЕННЯ ЙМОВІРНОСТЕЙ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, СКОРОЧЕНЬ І ТЕРМІНІВ

Умовні позначення, символи, скорочення, терміни	Пояснення умовних позначень, символів, скорочень, термінів
CSS	Спеціальна мова стилю сторінок та інтерфейсів, що використовується для опису їхнього зовнішнього вигляду [2]
C_n^m	Кількість сполучень з n по m
Electron	Програмний каркас для розробки графічних застосунків для настільних операційних систем за допомогою веб-технологій [3]
HTML	Мова розмітки гіпертексту, за допомогою якої розмічаються сторінки та інтерфейси [4]
JavaScript	Динамічна, об'єктно-орієнтована прототипна мова програмування, що використовується для керування даними та взаємодії з користувачем на сторінках та інтерфейсах мовою HTML [4]
$P(A)$	Ймовірність події A
$P_A(B)$	Ймовірність події B, за умови що відбулась подія A
SCSS	Скриптова метамова, яка інтерпретується в CSS. Призначений для підвищення рівня абстракції коду та спрощення файлів [2]

ВСТУП

Навчальний тренажер – нині актуальне програмне забезпечення для учбового процесу університетів, дистанційних курсів та інших закладів освіти, яке підвищує якість самостійної підготовки учнів.

Досліджуючи вже наявні тренажери на дистанційному курсі було виявлена відсутність навчального тренажера з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика». Можна відмітити несистематичний підхід до структури та оформлення деяких тренажерів. Ці факти породжують потребу у розробці нового навчального тренажера.

Мета роботи: створення програмного забезпечення тренажера з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика».

Завдання роботи:

- оглянути вже наявні тренажери на дистанційному курсі,
- виконати постановку задачі,
- сформувати задачі з теми «теореми додавання та множення ймовірностей випадкових подій», що мають бути опрацьованими у тренажері,
- дослідити теоретичні відомості щодо теми,
- скласти алгоритм роботи програми,
- створити блок-схему роботи складеного алгоритму,
- визначити засоби розробки,
- розробити навчальний тренажер,
- протестувати розробку,
- проаналізувати роботу та написати звіт.

Об'єкт роботи: створення навчальних тренажерів для дистанційного навчального курсу «Теорія ймовірностей і математична статистика».

Предмет роботи: програмне забезпечення тренажера з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика».

Методи дослідження: були проаналізовані літературні джерела, інформація з яких була використана у розробці прикладів задач з теми «Теореми додавання та множення ймовірностей випадкових подій», що реалізовані у навчальному тренажері. Також, ця інформація була застосована для створення алгоритму навчального тренажеру, програмного забезпечення розробленого на його основі та аналізу виконаної роботи.

Методи розробки: для реалізації програмного забезпечення було використано програмний каркас Electron та мови JavaScript, HTML та CSS. Для стиснення файлів та пакування проекту було застосовано gulp та electron-builder.

Новизна роботи: розроблений навчальний тренажер з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика» є програмним забезпеченням, що реалізує повний обсяг висунутих до нього вимог та задовольняє потребу в його створенні, яка випливає з відсутності навчального тренажеру з такої теми на дистанційному курсі.

Практична цінність: розроблений навчальний тренажер з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика» впроваджений на дистанційному курсі з дисципліни «Теорія ймовірностей і математична статистика» в Полтавському університеті економіки та торгівлі.

Структура пояснювальної записки: перелік умовних позначень, вступ, постановка задачі, інформаційний огляд, теоретична частина, практична частина, висновки, перелік літературних джерел, додатки.

Обсяг пояснювальної записки: 58 сторінок, з яких основна частина 45 сторінок та 13 сторінок додатку, літературних джерел – 8 назв, рисунків – 16, блок-схем – 3 сторінки.

РОЗДІЛ 1. ПОСТАНОВКА ЗАДАЧІ

1.1 Постанова задачі та її словесний опис

Поставлена мета потребує виконання наступних завдань:

- оглянути вже наявні тренажери на дистанційному курсі,
- виконати постановку задачі,
- сформувати задачі з теми «теореми додавання та множення ймовірностей випадкових подій», що мають бути опрацьованими у тренажері,
- дослідити теоретичні відомості щодо теми,
- скласти алгоритм роботи програми,
- створити блок-схему роботи складеного алгоритму.
- визначити засоби розробки,
- розробити навчальний тренажер,
- протестувати розробку,
- проаналізувати роботу та написати звіт.

До програмного забезпечення тренажера висунуті наступні вимоги:

- огляд теоретичних відомостей з теми «теореми додавання та множення ймовірностей випадкових подій»,
- покроковий та інтерактивний розв'язок типових задач з теми «теореми додавання та множення ймовірностей випадкових подій» з поясненням усіх етапів та перевіркою знань користувача,
- у разі помилки повідомити користувача та наштовхнути його на правильну відповідь,
- робота на популярних операційних системах університету, таких як windows7x86, із зворотною сумісністю та можливістю змінити платформу,
- сучасний та естетично привабливий інтерфейс, який би не відлякував користувачів і водночас не відволікав їх від навчання.

1.2 Перелік математичних задач, що увійдуть до тренажера

Використовуючи літературу, що стосується теми «Теореми додавання та множення ймовірностей випадкових подій» [5-6], було сформовано наступні задачі.

Задача 1. Пошук ймовірності суми несумісних подій.

В ящику знаходиться 7 червоних та 3 чорних кулі. З ящика виймають навмання 2 кулі. Яка ймовірність того, що вони однокольорові?

Задача 2. Пошук ймовірності суми сумісних подій.

Для отримання кредиту підприємець звертається до двох банків. Ймовірність того, що перший банк не відмовить в наданні кредиту становить 0,8, інший – 0,85. Знайти ймовірність того, що хоча б один банк дасть згоду на кредитування.

Задача 3. Пошук ймовірності добутку незалежних подій.

Три студенти одночасно складають іспит. Ймовірність складання іспиту на «відмінно» першим студентом – 0,5; другим – 0,7; третім – 0,4. Знайти ймовірність того, що три студенти складуть іспит на «відмінно».

Задача 4. Пошук ймовірності добутку залежних подій.

У відділі технічного контролю швейної фабрики представлено на огляд 60 жіночих суконь, з них 30 фасону №1, 20 фасону №2 та 10 фасону №3. Визначити ймовірність того, що взяті навмання три сукні всі виявляться фасону №1.

Задача 5. Пошук ймовірності комбінуванням теорем суми та добутку подій.

Три студенти одночасно складають іспит. Ймовірність складання іспиту на «відмінно» першим студентом – 0,5; другим – 0,7; третім – 0,4. Знайти ймовірність того, що двоє із цих студентів складуть іспит на «відмінно».

РОЗДІЛ 2. ІНФОРМАЦІЙНИЙ ОГЛЯД

2.1 Огляд робіт, де розглянуте аналогічне до теми роботи завдання

Були оглянуті магістерські та дипломні роботи студентів, а також наявні навчальні тренажери на дистанційному курсі, завдання яких найбільш наближене до теми роботи та реалізують широкий обсяг функцій та рішень.

З них для детального аналізу обрано роботи Белінської В. В. [7], та Жайворонка Я. І. [8], які більше інших відповідають висунутим вимогам.

Розглянемо навчальний тренажер Белінської В. В. з теми «Розподіли дискретних випадкових величин та їх числові характеристики» дистанційного навчального курсу «Теорія ймовірностей та математична статистика». У стартовому вікні можна обрати теоретичну або практичну частину. Теоретичний розділ має перелік тестових запитань. Практичний розділ містить підрозділи на певні підтеми, що ведуть до розв'язку декількох задач.

Розглянемо навчальний тренажер Жайворонка Я. І. з теми «Коди із виявленням помилок» дистанційного навчального курсу «Теорія інформації та кодування». Після запуску він зустрічає користувача вікном із кнопками «Почати», «Мова» та «Вихід». Якщо натиснути на першу, то у ньому з’явиться перелік з восьми розділів. Якщо обрати один з них, то розпочнеться етап з вибором теоретичної та практичної частини. Вони мають декілька кроків, які стосуються теми розділу. Тестування для теоретичної частини та розв’язку задач для практичної. На усіх сторінках з безпосередньо тестами або задачами є кнопка «Допомога», яка відкриває файл з лекцією до поточної теми в онлайн-джерелі.

2.2 Позитивні аспекти оглянутих робіт

Навчальний тренажер Белінської В. В. з теми «Розподіли дискретних випадкових величин та їх числові характеристики» дистанційного навчального курсу «Теорія ймовірностей та математична статистика» реалізовує розв’язок задач із введенням обчислень та охоплює свою тему.

Присутні повідомлення про помилки з поясненнями до поточного кроку задачі або тесту.

Слід відмітити можливість обрати мову програми з трьох доступних.

Навчальний тренажер Жайворонка Я. І. з теми «Коди із виявленням помилок» дистанційного навчального курсу «Теорія інформації та кодування» реалізовує розв’язок задач із введенням обчислень та охоплює свою тему.

Програма мультиплатформна, її можна встановити на різні операційні системи.

Тренажер підтримує декілька мов.

Присутні візуальні виділення та повідомлення про правильність відповіді.

2.3 Вади розробок з оглянутих робіт

Зазначимо такі недоліки навчального тренажеру Белінської В. В.

Теоретична та практична частина відірвані одна від одної. В практичній частині розв'язок задач складається лише безпосередньо з полів для ведення обчислень та відповіді, немає детального опису кроків, теоретичного матеріалу, які користувач мусить опрацювати щоб зрозуміти та розв'язати приклад. Вірогідно, що такий формат не сприятиме засвоєнню знань учнем.

Слід зазначити також відсутність додаткових пояснень у разі помилки обчислень у практичній частині. Згідно алгоритму тренажера переважна більшість таких повідомлень наводять користувачу лише формулу, за якою можна розв’язати задачу. Це може призвести до того, що учень застряне на цьому етапі без можливості пройти далі.

Відмітимо також стилістично не витриманий дизайн, великий розмір шрифту та його низьку контрастність в тренажері, а саме темно-сірий колір тексту на сірому фоні. Це може погіршити читабельність та викликати надлишкову втому очей.

Також, згідно опису алгоритму, в навчальному тренажері відсутнє перемішування варіантів відповідей у тестах.

Відсутня можливість перегляду попередніх кроків.

Розглянемо недоліки навчального тренажеру Жайворонка Я. І. В цілому, робота набула вади, аналогічні до тренажеру Белінської В. В.

Теоретичні та практичні частини розділені. Ймовірно, що учні не зрозуміють як розв’язати задачу через те, що їм не пояснюють кроків які необхідно пройти для цього. Це знецінює тренажер як самостійне джерело для навчання, адже учень буде змушений звернутися до викладача для додаткових пояснень або переглянути теоретичні відомості в сторонньому ресурсі.

Звернемо увагу і на таку проблему цього тренажера, як заплутану навігацію між розділами. Було б доцільно уникнути стартове меню та зразу відобразити теми на вибір. Кнопка «Вихід» нефункціональна, тому що дублює призначення стандартних засобів керування вікнами. Кожен зайвий клік це втрата часу та концентрації користувача.

Підказки до кроків практичної частини, які з’являються за помилки, неінформативні та відсилають до кроків теоретичної частини. Це незручно та може заплутати користувача. Тим більше, що учень може застряти на якомусь з етапів. Однак тестові запитання в іншій частині зовсім не мають підказок та позиціонують себе як перевірку знань з їх оцінкою у балах. Вочевидь, що це не головне призначення навчального тренажера.

Зауважимо, що дизайн тренажеру в цілому сумнівний та не сприяє концентрації, текст не контрастний, що заважає його сприйняттю.

Також, згідно опису алгоритму, в цьому тренажері відсутнє перемішування варіантів відповідей у тестах.

2.4 Висновки аналізу робіт попередників

Розглянуті приклади наводять на наступні висновки та зауваження, які слід врахувати під час розробки навчального тренажера.

Практична та теоретична частина мають бути одним цілим та доповнювати одна одну. Тренажер в першу чергу мусить навчити користувача як саме потрібно мислити, щоб розв'язати задачу, а не просто перевірити його знання.

Повідомлення та підказки мають бути інформативними, наштовхувати на відповідь, або давати її, якщо користувач довго не може розв'язати завдання.

Структура інтерфейсу має тяжіти до мінімалізму, уникати зайвих розділів та кнопок. В цілому, скоротити кількість дій, які необхідно робити для користування.

Дизайн має бути привабливим, але водночас і не відволікати учнів, бути інтуїтивно зрозумілим. Також слід врахувати різні тонкощі, що стосуються читабельності тексту, підбору кольорів, розмірів і тому подібного.

Отже, зважаючи на усе це, можна зробити висновок, шо є потреба у новому та сучасному тренажері, який би враховував зауваження та пропозиції, які були сформульовані вище та осягав тему «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика»

РОЗДІЛ 3. ТЕОРЕТИЧНА ЧАСТИНА

3.1 Алгоритмізація задачі за темою роботи

У стартовому вікні знаходиться інформація про: тему, університет, виконавця та наукового керівника.

Посередині стартового вікна знаходяться кнопки, що ведуть до відповідних розділів:

- Задачі про пошук ймовірності суми подій,
- Задачі про пошук ймовірності добутку подій,
- Задача про пошук ймовірності комбінуванням теорем суми та добутку подій.

У розділі «Задачі про пошук ймовірності суми подій» користувачеві пропонується вибрати між задачами:

- Задача про пошук ймовірності суми несумісних подій,
- Задача про пошук ймовірності суми сумісних подій.

У розділі «Задачі про пошук ймовірності добутку подій» користувачеві пропонується вибрати між задачами:

- Задача про пошук ймовірності добутку незалежних подій,
- Задача про пошук ймовірності добутку залежних подій.

Після того, як користувач обере задачу з наведених вище розділів, він розпочне покроковий інтерактивний розв’язок вибраної задачі.

Якщо ж буде обрано розділ «Задача про пошук ймовірності комбінуванням теорем суми та добутку подій.» то користувач відразу розпочне розв’язок задачі на цю тему.

Інтерфейс вікна з розв’язкам задачі складається із:

- закріпленого зверху блоку з умовою задачі,
- блоку з кроками розв’язування задачі.

На кожному кроці користувач має дати відповідь на тестове завдання або записати свої обчислення задачі у відповідні поля.

Частина кроків може бути супроводжена додатковими поясненнями та відомостями.

Перевірка відповіді відбувається у режимі реального часу – як тільки користувач вибере варіант відповіді або заповнить усі поля прикладу. Для навігації в полях рекомендується використовувати кнопку «Tab».

Якщо відповідь правильна, то вона візуально відмічається зеленим кольором та даний блок стає недоступним для зміни. Після чого здійснюється перехід до наступного кроку.

Якщо відповідь неправильна, то вона візуально відмічається червоним кольором та по центру програми з'являться вікно, яке повідомляє про помилку та наводить підказку до правильної відповіді.

3.1.1. Алгоритми задач пошуку ймовірності суми подій

Розглянемо наступну задачу №1 та її розв'язок. На її основі буде розроблений алгоритм задачі про пошук ймовірності суми несумісних подій.

Умова задачі: В ящику знаходиться 7 червоних та 3 чорних кулі. З ящика виймають навмання 2 кулі. Яка ймовірність того, що вони однокольорові?

Розв'язання. Нехай подія A – поява двох червоних куль, а подія B – поява двох чорних куль. Події A та B несумісні. Число всіх єдиноможливих, рівноможливих і несумісних випадків дорівнює числу пар, які можна утворити з десяти різних куль.

$n = C_{10}^2 = \frac{9*10}{2} = 45$ – число загальних випадків.

Число випадків, сприятливих для появи двох червоних куль, дорівнює

$$C_7^2 = \frac{7*6}{2} = 21$$

Число випадків, сприятливих для появи двох червоних куль, дорівнює

$$C_3^2 = \frac{3*2}{2} = 3$$

Тоді маємо розв'язок:

$$P(A+B) = P(A) + P(B) = \frac{21}{45} + \frac{3}{45} = \frac{24}{45}$$

Перенесемо її у покрокове розв'язування відповідно алгоритму тренажера:

У кожному тесті перша відповідь правильна.

Умова задачі: В урні знаходиться 7 червоних та 3 чорних кулі. З ящика виймають навмання 2 кулі. Яка ймовірність того, що вони однокольорові?

Крок 1. Нехай:

Подія *A* з урни витягли 2 червоні кулі

Подія *B* з урни витягли 2 чорних кулі

Подія *M* - витягнуто однокольорові кулі

Згадаємо деякі теоретичні відомості.

Випадкові події *A* та *B* називаються сумісними, якщо:

- поява однієї події не виключає появи іншої.
- поява однієї події виключає появу іншої.
- поява однієї події залежить від появи іншої.
- поява однієї події відбувається лише одночасно з появою іншої.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Випадкові події *A* та *B* називаються сумісними, якщо поява однієї події не виключає появи іншої.»

Крок 2. В заданій задачі події *A* та *B*:

- несумісні
- сумісні

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Події *A* та *B* несумісні, так як за умовою задачі подія *A*, за якої витягнуто дві кулі червоного кольору, виключає можливість відбутися події *B*, за якої витягнуто дві кулі чорного кольору і навпаки».

Крок 3. Сумою двох подій *A* та *B* називають подію, що полягає у появі:

- події *A* або події *B*, або обох цих подій одночасно,
- обох цих подій,
- події *A* або події *B*.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Сумою двох подій *A* та *B* називають подію, що полягає у появі хоча б однієї з них».

Крок 4. Добутком двох подій *A* та *B* називають подію, що полягає у появі:

- в одночасній появі цих подій.
- обох цих подій.
- події *A* або події *B*.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Добутком двох подій *A* та *B* називають подію, що полягає у появі обох цих подій одночасно».

Крок 5. Подію *M* можна виразити через події *A* та *B* таким чином:

- $A+B$
- $A*B$
- $A+B-(A*B)$
- $A*B-(A+B)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Подію *M* можна виразити через події *A* та *B* як їх суму за означенням суми ймовірностей подій».

Крок 6. Ймовірність суми несумісних подій:

- $P(A+B)=P(A)+P(B)$
- $P(A+B)=P(A)*P(B)$
- $P(A+B)=P(A)+P(B)-P(A*B)$
- $P(A+B)=P(A+B)-P(A*B)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Ймовірність суми несумісних подій: $P(A+B)=P(A)+P(B)$.»

Крок 7. За яким означення ймовірностей обчислюється *P(A)* та *P(B)*?

- Класичне,
- Статистичне,
- Геометричне.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «*P(A)* та *P(B)* обчислюється за класичним означення ймовірностей»

Крок 8. Число загальних випадків – число всіх рівноможливих і несумісних випадків, тобто кількість пар, які можна утворити з 10 куль.

Що із нижче зазначеного необхідно обчислити для того, щоб знайти число загальних випадків?

- Кількість сполучень C_n^m
- Кількість розміщень A_m^n
- Кількість перестановок P_n

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Для того, щоб знайти число загальних випадків необхідно обчислити кількість сполучень C_n^m».

Крок 9. Яка формула використовується для обчислення кількості сполучення?

- $C_n^m = \dfrac{n!}{m!(n-m)!}$
- $C_n^m = \dfrac{n!}{n!(n-m)!}$
- $C_n^m = \dfrac{m!}{m!(n-m)!}$
- $C_n^m = \dfrac{n!}{m!(m-n)!}$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Для обчислення кількості сполучення використовується формула $C_n^m = \dfrac{n!}{m!(n-m)!}$. »

Крок 10. Отже, число загальних випадків:

$C_{10}^2 = \dfrac{\square * \square}{\square} = \square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Обчисліть кількість сполучень за формулою з попереднього кроку та впишіть у комірки».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $C_{10}^2 = \frac{9*10}{2} = 45$».

Крок 11. Для того щоб розрахувати ймовірність подій *A* та *B* потрібно знайти число випадків, сприятливих для появи двох червоних куль або двох чорних куль відповідно.

Кількість сприятливих випадків для появи двох червоних куль позначається як:

- C_7^2
- C_3^2
- C_{10}^2
- C_7^3

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Кількість сприятливих випадків можна позначити як кількість сполучень з 2 по 7».

Крок 12. Кількість сприятливих випадків для появи двох червоних куль позначається як:

- C_3^2
- C_7^2
- C_{10}^2
- C_7^3

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Кількість сприятливих випадків можна позначити як кількість сполучень з 2 по 3».

Крок 13. Знайти число випадків, сприятливих для появи двох червоних куль: $C_7^2 = \frac{\square * \square}{\square} = \square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Обчисліть кількість сполучень за формулою з попереднього кроку та впишіть у комірки».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $C_7^2 = \frac{7*6}{2} = 21$».

Крок 14. Знайти число випадків, сприятливих для появи двох чорних куль:

$$C_3^2 = \frac{\square * \square}{\square} = \square$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Обчисліть кількість сполучень за формулою з попереднього кроку та впишіть у комірки».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $C_3^2 = \frac{3*2}{2} = 3$».

Крок 15. Знайти: ймовірність витягнути червоні кулі:

$$P(A) = \frac{\square}{\square}$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Впишіть знайдені кількості сполучень у комірки, щоб сформувати ймовірність».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(A) = \frac{21}{45}$».

Крок 16. Ймовірність витягнути чорні кулі:

$$P(B) = \frac{\square}{\square}$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Впишіть знайдені кількості сполучень у комірки, щоб сформувати ймовірність».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(B) = \frac{3}{45}$».

Крок 17. Отже ймовірність витягнути однокольорові кулі:

$$P(M) = P(A + B) = P(A) + P(B) = \frac{\square}{\square} + \frac{\square}{\square} = \frac{\square}{\square}$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Впишіть знайдені ймовірності у комірки та розрахуйте результат виразу».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(A+B)=P(A)+P(B)=\frac{21}{45}+\frac{3}{45}=\frac{24}{45}$».

Розглянемо алгоритм розв’язку задачі №2 про пошук ймовірності суми сумісних подій. Він складається з 9 кроків.

Далі в описі алгоритму в тестових запитаннях лише перша відповідь правильна. В тренажері відповіді перемішуються.

Умова задачі: для отримання кредиту підприємець звертається до двох банків. Ймовірність того, що перший банк не відмовить в наданні кредиту становить 0,8, інший – 0,85. Знайти ймовірність того, що хоча б один банк дасть згоду на кредитування.

Крок 1. Нехай:

Подія *M* - банк надав кредит

Подія *A* - перший банк надав кредит

Подія *B* - другий банк надав кредит

Згадаємо деякі теоретичні відомості.

Випадкові події *A* та *B* називаються сумісними, якщо:

- появa однієї події не виключає появи іншої.
- появa однієї події виключає появу іншої.
- появa однієї події залежить від появи іншої.
- появa однієї події відбувається лише одночасно з появою іншої.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Випадкові події *A* та *B* називаються сумісними, якщо появa однієї події не виключає появи іншої».

Крок 2. В заданій задачі події *A* та *B*:

· сумісні

· несумісні

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Події *A* та *B* сумісні, так як згода одного банку не виключає згоду іншого».

Крок 3. Сумою двох подій *A* та *B* називають подію, що полягає у появі:

· події *A* або події *B*, або обох цих подій одночасно.

· обох цих подій.

· події *A* або події *B*.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Сумою двох подій *A* та *B* називають подію, що полягає у появі хоча б однієї з них».

Крок 4. Подію M можна виразити через події *A* та *B* таким чином:

· $A+B$

· $A*B$

· $A+B-(A*B)$

· $A*B-(A+B)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Подію *A* можна виразити як суму подій *A* та *B*».

Крок 5. Отже, ймовірність появи хоча б однієї із двох сумісних подій дорівнює:

· $P(A+B)=P(A)+P(B)-P(A*B)$

· $P(A+B)=P(A*B)-P(A)-P(B)$

· $P(A+B)=P(A)+P(B)$

· $P(A+B)=P(A)*P(B)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Ймовірність появи хоча б однієї із двох сумісних подій дорівнює сумі ймовірностей цих подій без імовірності добутку цих подій»

Крок 6. Добутком двох подій *A* та *B* називають подію, що полягає у появі:

· обох цих подій одночасно

· події *A* або події *B*

· події *B* за умови, що відбулась подія *A*

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Добутком двох подій *A* та *B* називають подію, що полягає у появі обох подій».

Крок 7. В заданій задачі події *A* та *B*:

· незалежні

· залежні

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Події *A* та *B* незалежні, тому що ймовірність згоди кожного банку не залежить від того, чи дав згоду інший банк».

Крок 8. Тому, щоб знайти добуток *A* та *B* потрібно використати формулу:

· $P(A*B)=P(A)*P(B)$

· $P(A*B)=P(A)*P_A(B)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Ймовірність добутку подій *A* та *B* дорівнює добутку ймовірностей цих подій».

Крок 9. Отже, формула має такий вигляд:

$P(M)=P(A)+P(B)-(P(A)*P(B))$.

Тому, ймовірність події *M* обчислюється так:

$P(M)=\square+\square-\square*\square=\square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Впишіть ймовірності з умови задачі у комірки та розрахуйте результат виразу».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(A)=0,8+0,85-0,8*0,85=0,97$».

3.2 Розробка блок-схеми, яка підлягає програмуванню

На рисунках 3.1-3.4 зображені блок-схеми алгоритму навчального тренажера, а саме: блок-схема головного алгоритму роботи програми тренажера (рис. 3.1), блок-схему обробки питання з тестом (рис. 3.2), блок-схему обробки питання з полями (рис. 3.3) та блок-схему виведення помилок (рис. 3.4).

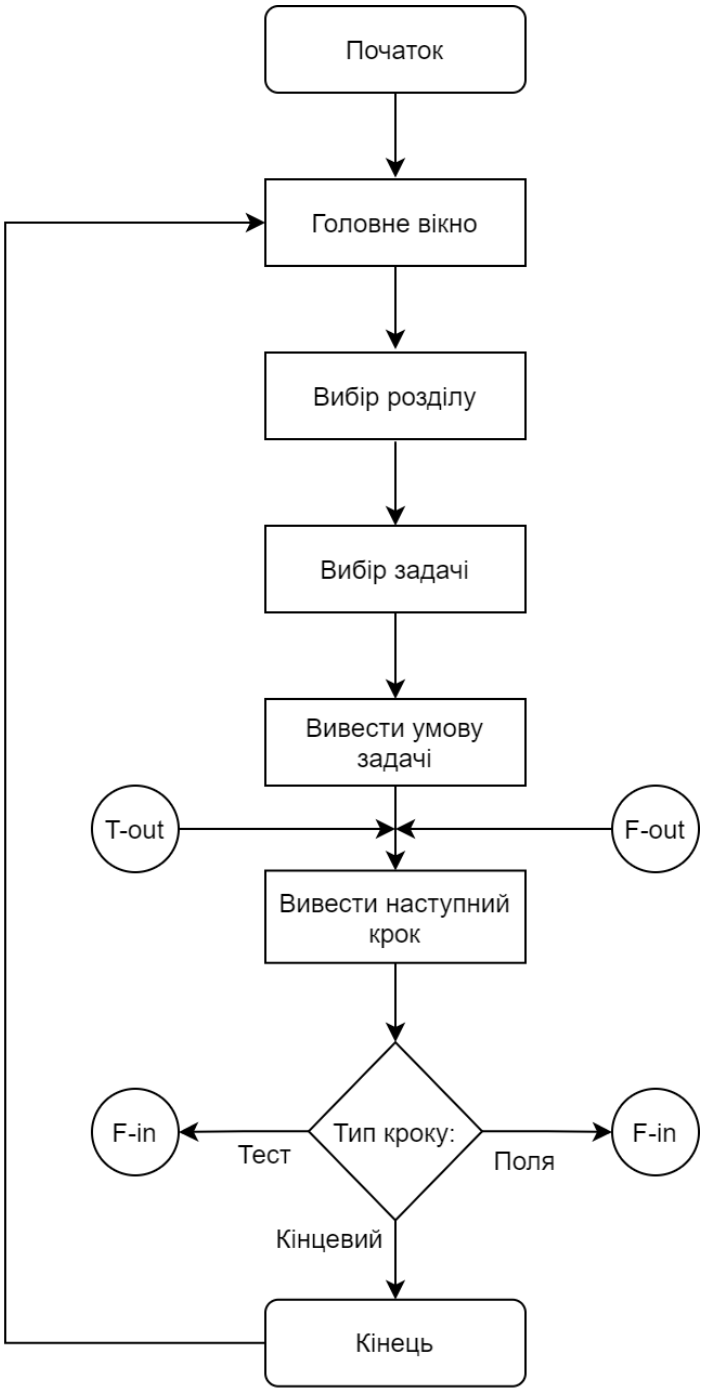


Рисунок 3.1 – Блок-схема програми

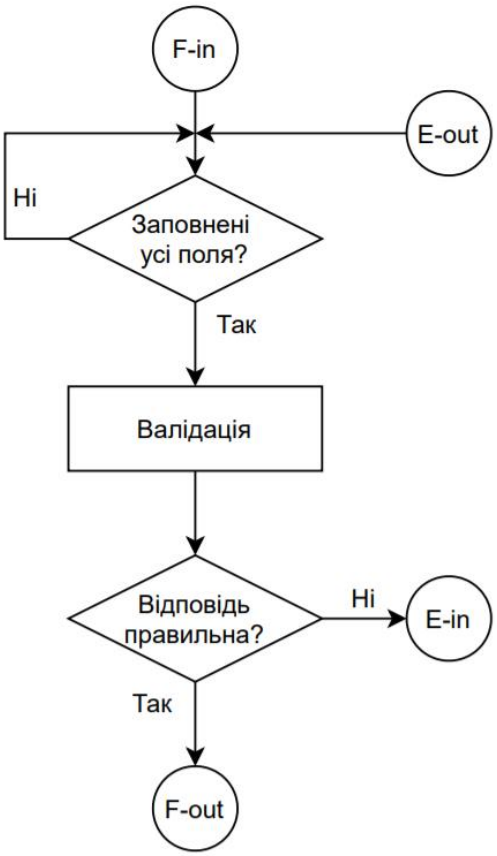


Рисунок 3.2 – Блок-схема F-in/F-out – обробник кроку типу «Поля»

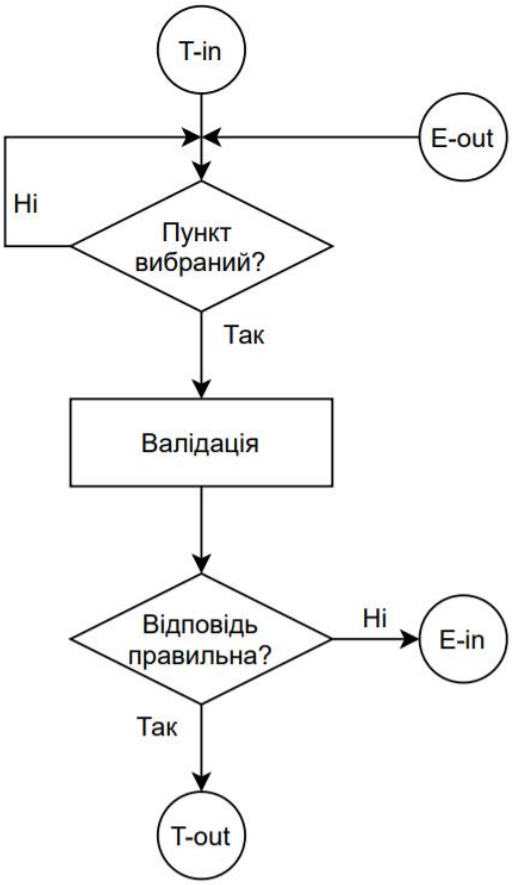


Рисунок 3.3 – Блок-схема T-in/T-out – обробник кроку типу «Тест»

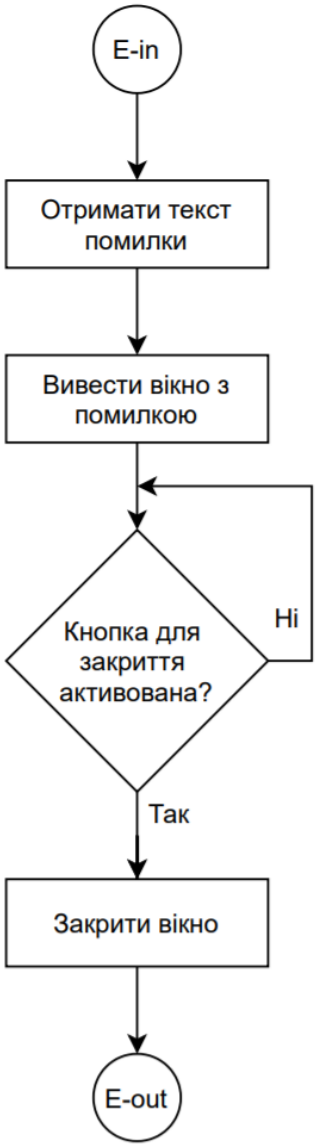


Рисунок 3.4 – Блок-схема E-in/E-out – Виведення помилок

3.3 Обґрунтування вибору програмних засобів

Навчальний тренажер, як програма для широкого кола користувачів, перш за все повинна мати графічний інтерфейс. Окрім цього, згідно з вимогами, він мусить бути й привабливим. Тому, для його реалізації було розглянуто декілька платформ та технологій, які можна використати у розробці, а саме:

- Qt – сучасний інструментарій розробки програмного забезпечення мовою програмування C++,
- GTK – набір інструментів для створення графічних інтерфейсів користувача багатьма мовами, що разом із Qt є одними із найпопулярніших,
- Electron – програмний каркас для розробки графічних застосунків за допомогою веб-технологій.

Зрештою, було обрано Electron та низку веб-технології. Попри свої недоліки, такі як збільшена потреба в ресурсах комп'ютера, Electon має вагомий функціонал, який дозволить з найвищою точністю реалізувати задуми щодо вигляду графічного інтерфейсу. Окрім цього він дає можливість використати досвід роботи із попередніми аналогічними проектами.

Вочевидь, вибір платформи, що використовує веб-технології, спонукає до їх використання у розробці. Для розмітки сторінок використовується HTML, для їх стилізації – CSS. А за поведінку програми відповідає JavaScript. Загалом, ці мови кодування та програмування забезпечують необхідну гнучкість у розробці сучасних інтерфейсів.

Для інтерпретації SCSS в CSS та стиснення CSS та JavaScript був застосований Gulp – таск-менеджер для автоматичного виконання завдань, що закодовані мовою JavaScript у файлі gulpfile.js (додаток В), виконання яких можна запустити з консолі.

Для пакування програми у портативний виконуваний файл було використано electron-builder – консольний застосунок, який призначений для цього.

РОЗДІЛ 4. ПРАКТИЧНА ЧАСТИНА

4.1 Опис програми

Якщо користувач запустить програму, то по центру екрана відкриється головне вікно (рис. 4.1). Посередині нього знаходяться три кнопки, що ведуть до відповідних розділів із задачами:

- Задачі про пошук ймовірності суми подій,
- Задачі про пошук ймовірності добутку подій,
- Задача про пошук ймовірності комбінуванням теорем суми та добутку подій.

Якщо навести на одну з них курсором, то вона плавно здіймається вгору на декілька пікселів.

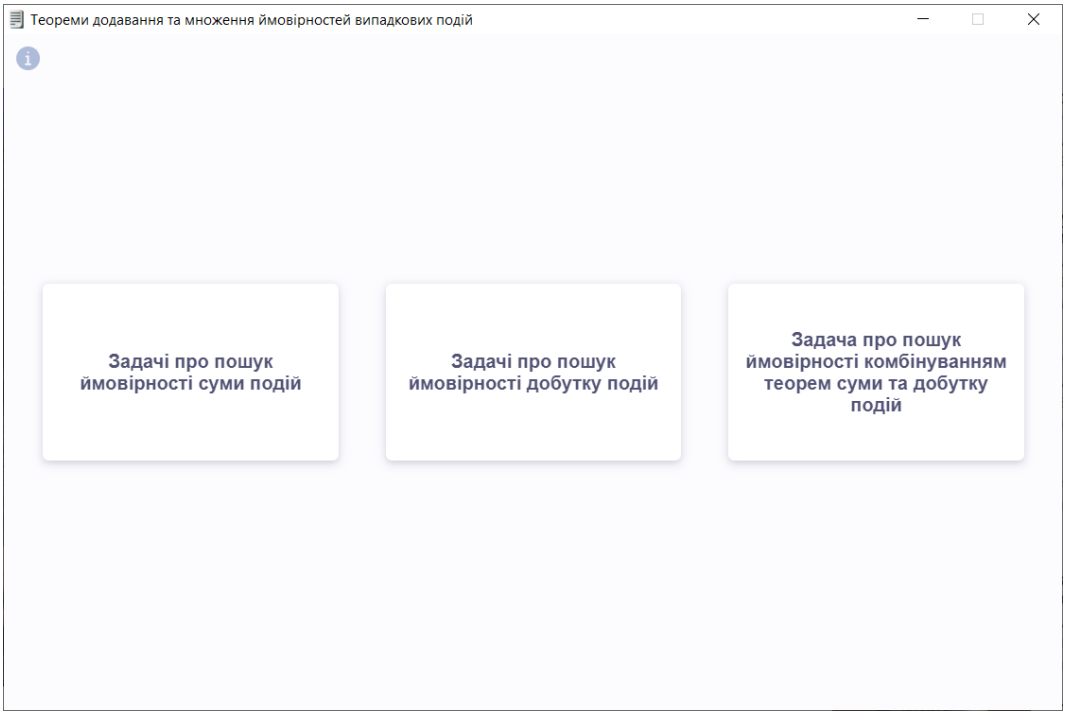


Рисунок 4.1 – Головне меню

Зверху у лівому куті знаходиться кнопка «Інфо», якщо натиснути на неї то у вікні зверху та знизу плавно з’явиться текст з інформацією про: тему, виконавця та наукового керівника (рис. 4.2). Повторне натискання кнопки ховає цей текст.

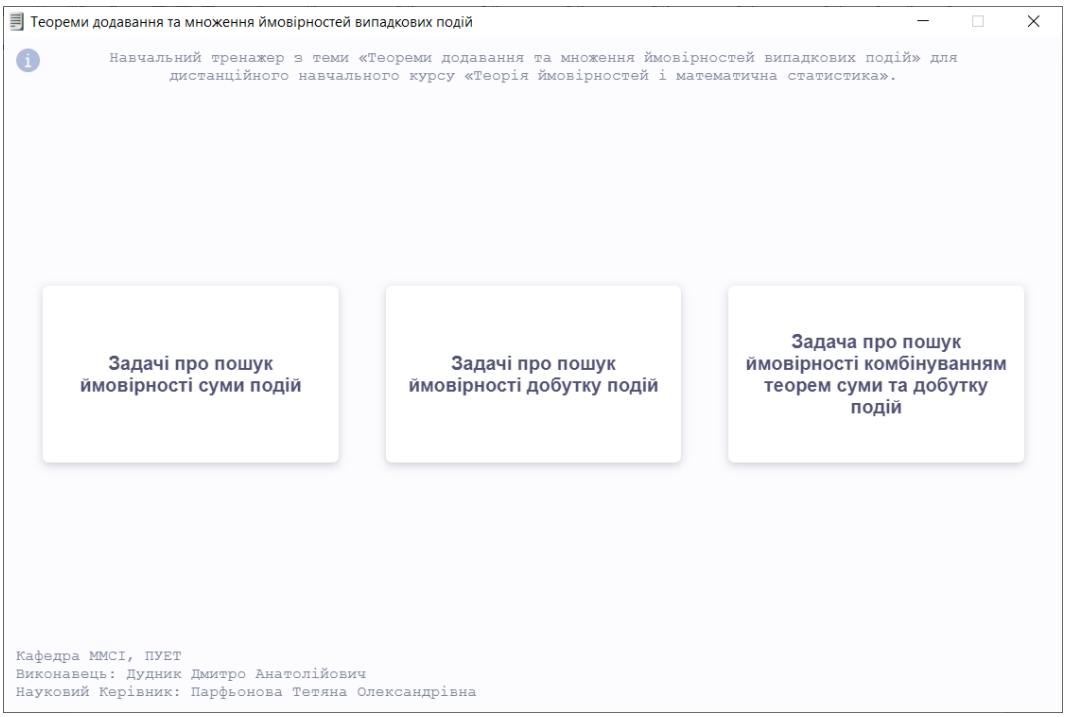


Рисунок 4.2 – Текст з інформацією про програму

Якщо обрати розділ, то вміст вікна зміниться на кнопки з описом задач на вибір (рис. 4.3).

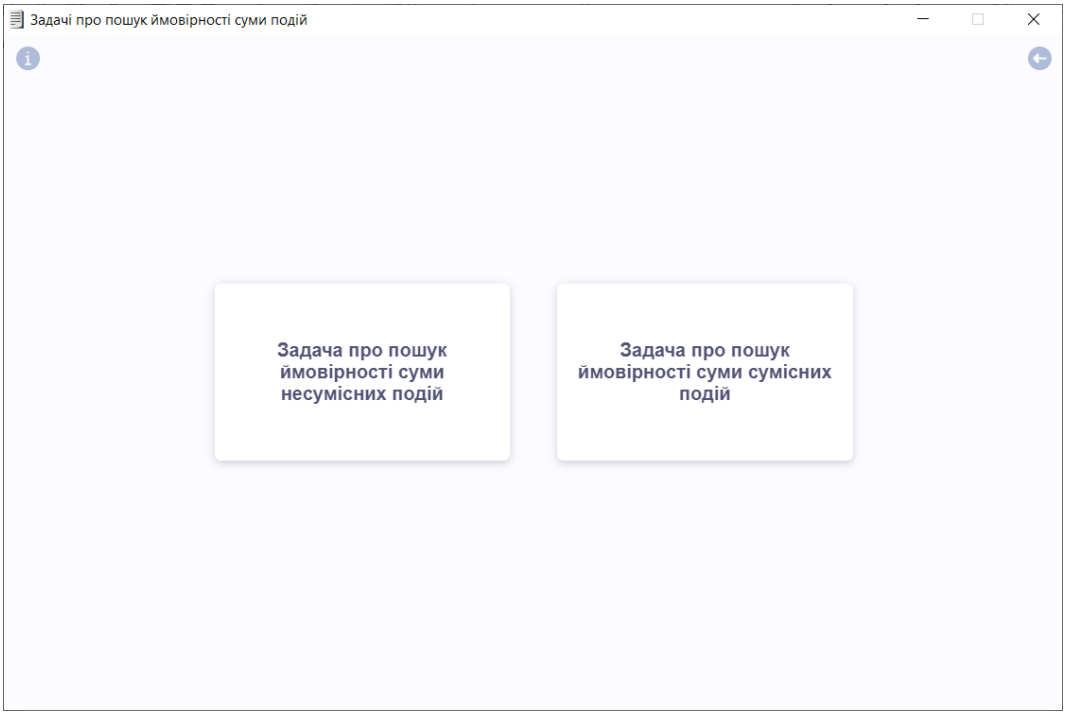


Рисунок 4.3 – Розділ з обраннями задачі

Тут присутня кнопка «Інфо» як і у попередньому вікні. Зверху у правому куті розташована кнопка «Назад», що веде до попереднього розділу, якщо на неї натиснути.

Коли користувач вибере одну із задач, то розпочнеться її покроковий розв’язок (рис. 4.4).

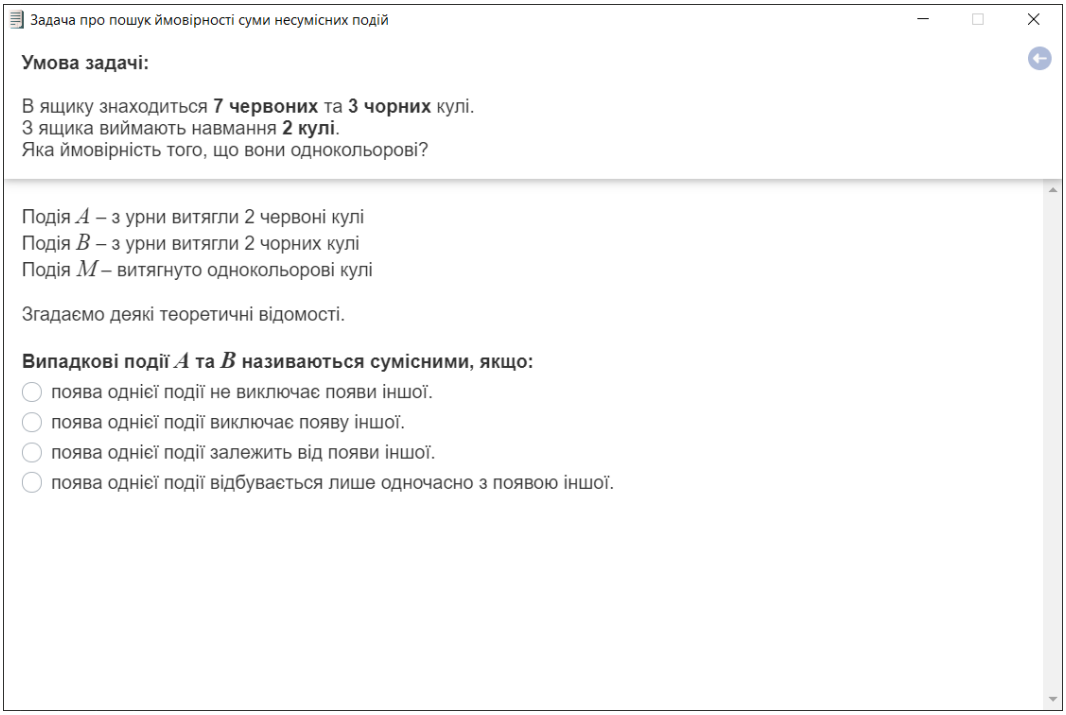


Рисунок 4.4 – Вікно із розв'язком задачі

Як і у минулому розділі, тут є кнопка «Назад».

Верхню частину вікна займає закріплений блок із умовою задачі. Ключова інформація у ній виділена жирним накресленням шрифту.

Іншу ж частину вікна займає розв’язок задачі. У цьому блоку кожен крок з'являється поступово один за одним, після того як був пройдений попередній. Його можна вільно прогортати, переглядаючи попередні етапи розв’язку задачі.

Кроки можна поділити на два типи: тести та поля для введення обчислень.

Деяка частина з них доповнюються поясненнями або відомостями.

Відповідь перевіряється як тільки користувач обере варіант відповіді у тесті, а бо заповнить усі поля у введені обчислень.

Якщо відповідь правильна, то вона візуально відмічається зеленим кольором та цей крок стає недоступним для зміни (рис. 4.5). Після чого здійснюється перехід до наступного кроку .

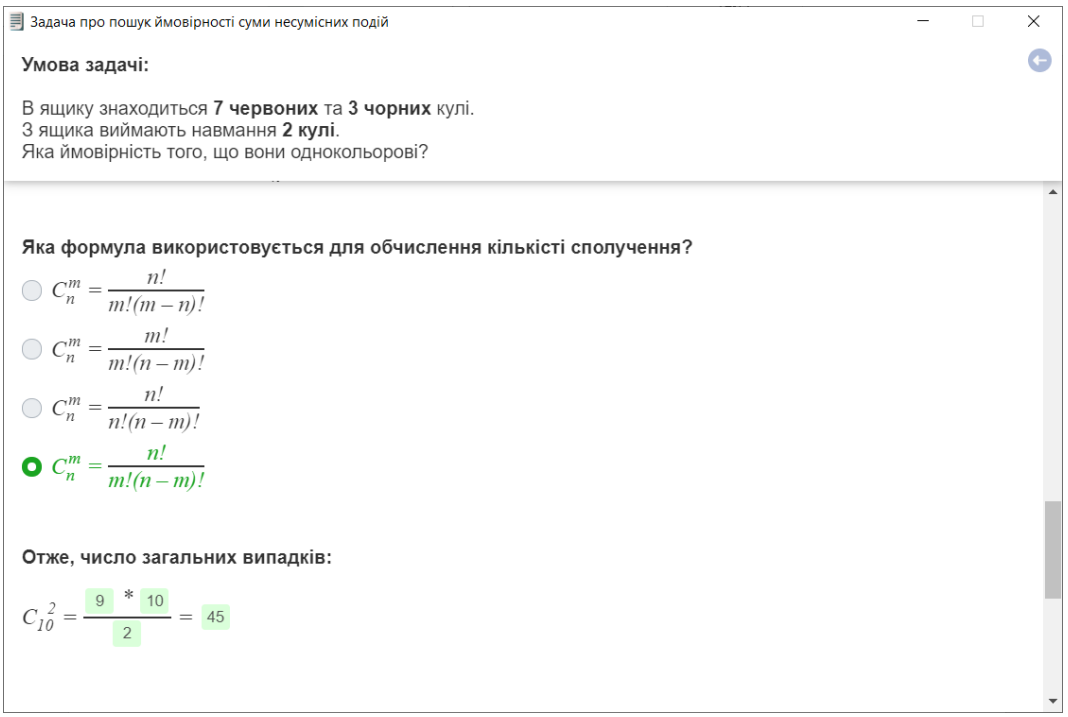


Рисунок 4.5 – Правильні відповіді

Якщо відповідь неправильна, то вона візуально відмічається червоним кольором та по центру програми з’являться вікно, яке повідомляє про помилку та наводить підказку до правильної відповіді або пояснення (рис. 4.6).

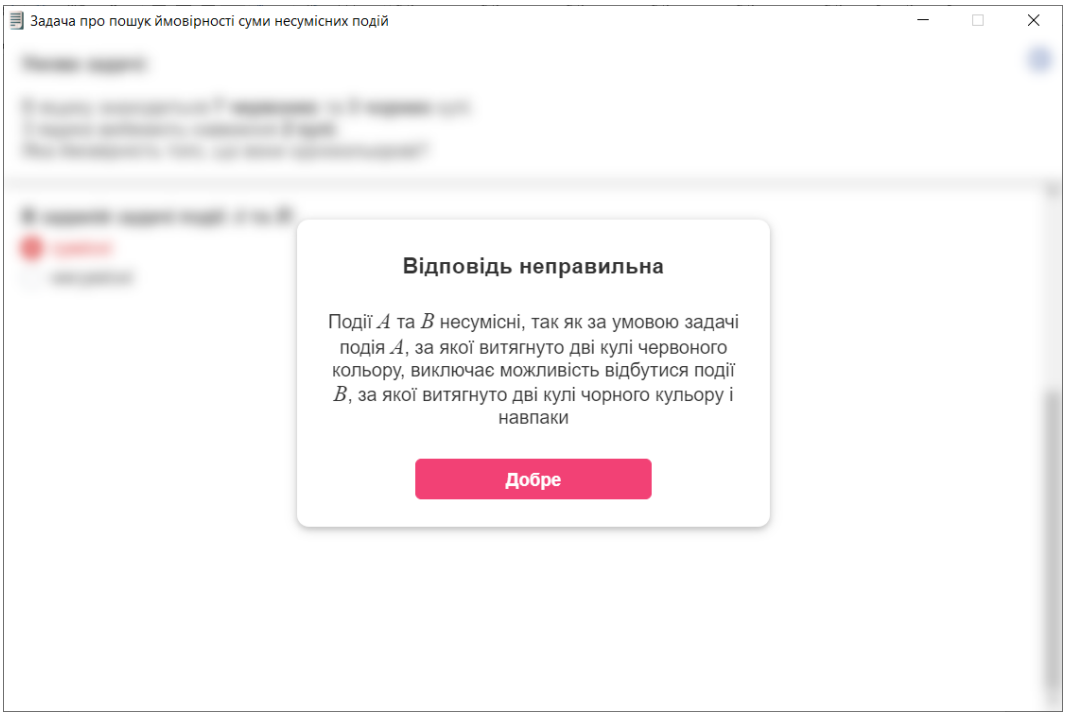


Рисунок 4.6 – Повідомлення про помилку

Крок із тестом це запитання з варіантами відповідей. З кожним новим проходженням задачі ці варіанти перемішуються між собою (рис. 4.7).

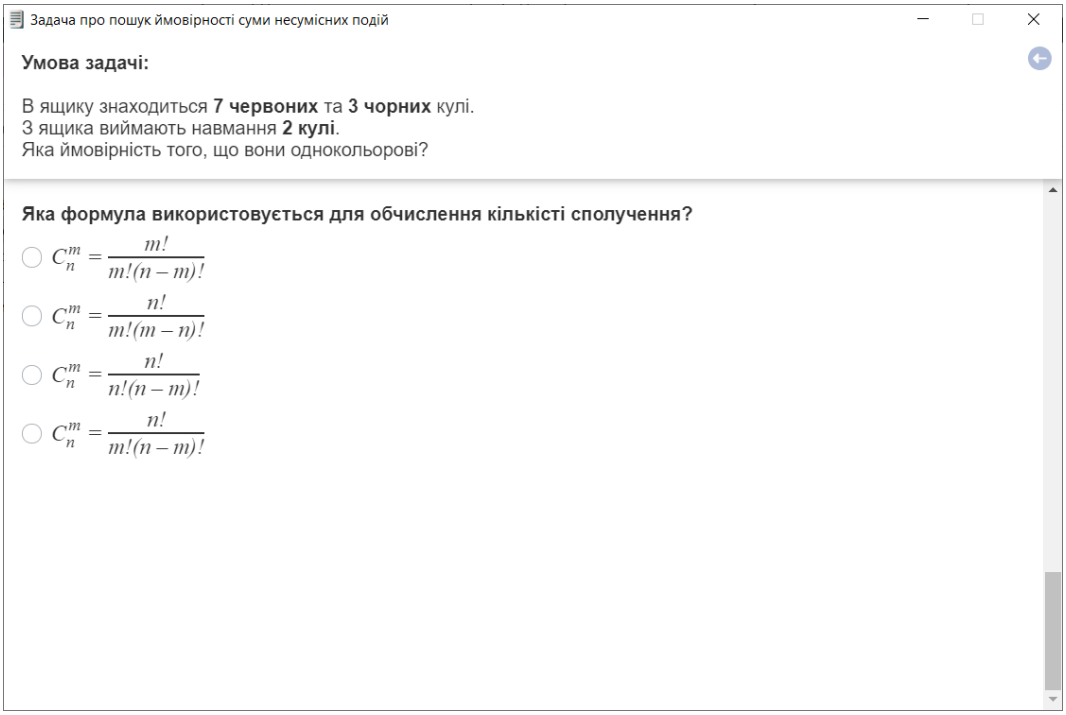


Рисунок 4.7 – Крок із тестом

Крок із введенням обчислень це набір полів, які утворюють структуру, куди користувачу потрібно вводити свою відповідь (рис. 4.8).

Для навігації між комірками доцільно використовувати кнопку «Tab»

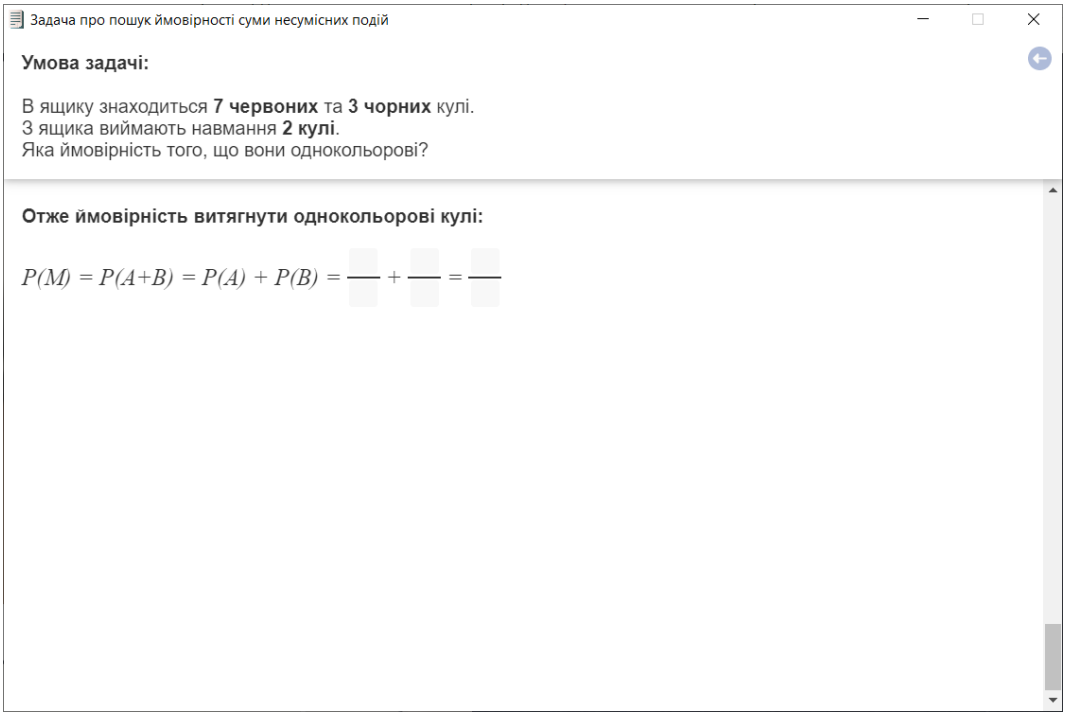


Рисунок 4.8 – Крок з комірками для введення обчислень

Цей тип кроку, окрім першої підказки (рис. 4.9), супроводжений додатковою другою (рис. 4.10), яка з’являється, якщо користувач помиляються вдруге . В ній знаходиться відповідь з уже заповненим виразом. Це гарантує, що учень не застряне на одному етапі. Також, ймовірно, шо він зрозуміє хід обчислень, які необхідно було виконати, коли він побачить готовий розв’язок цього кроку.

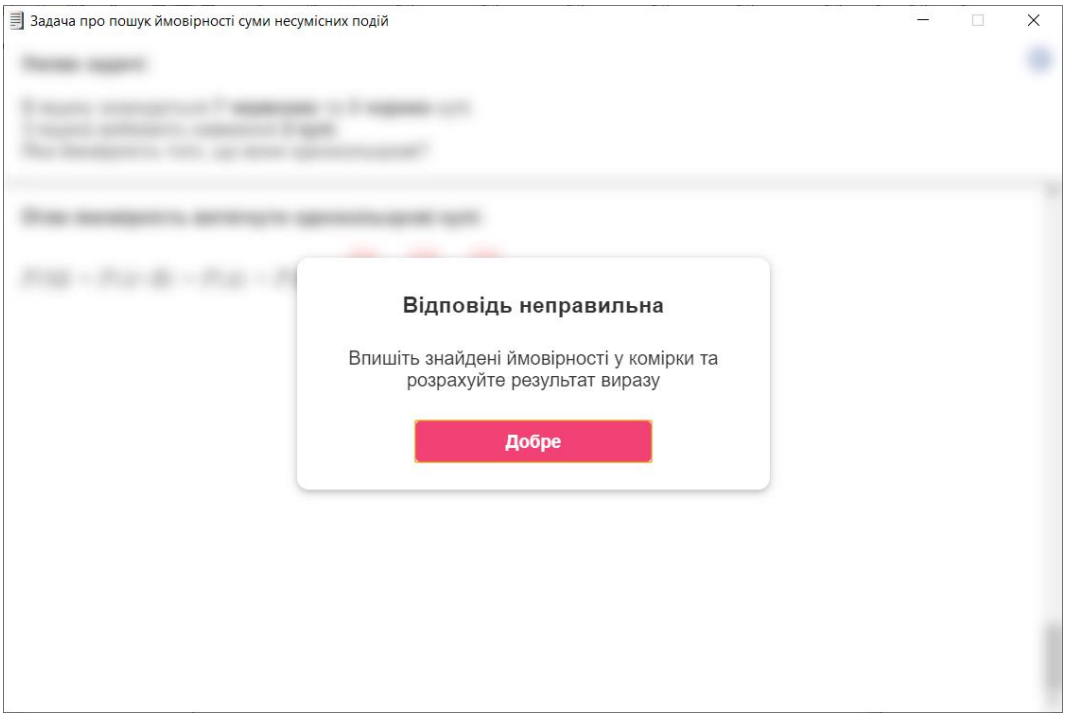


Рисунок 4.9 – Перше повідомлення про помилку з підказкою

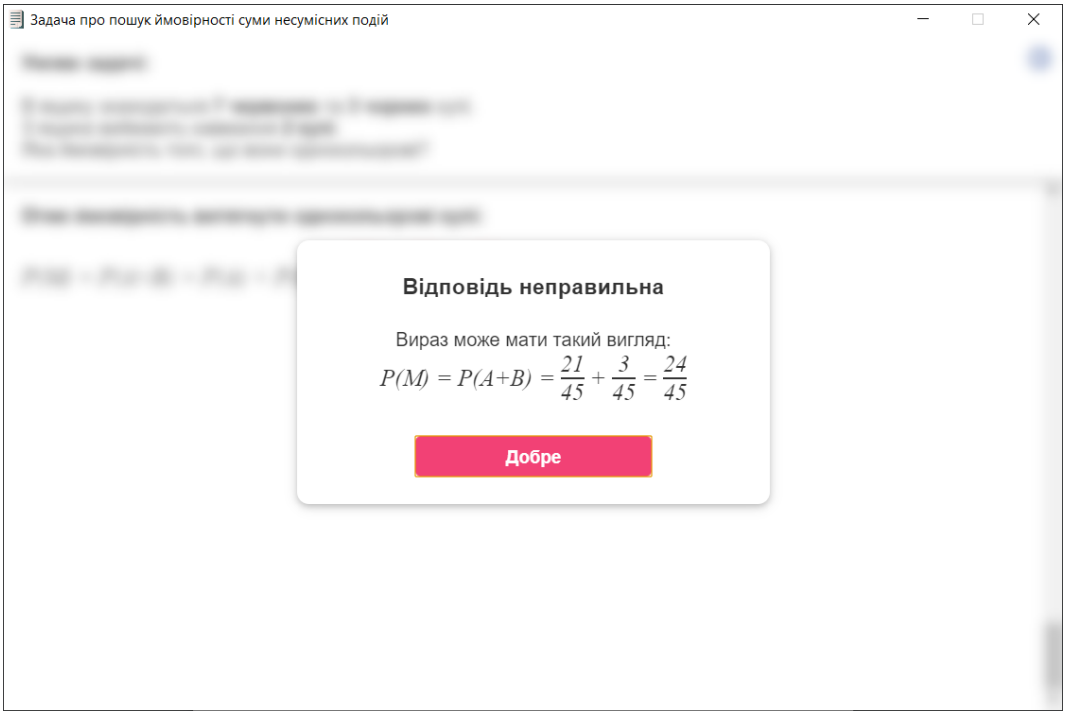


Рисунок 4.10 – Додаткове повідомлення про помилку з підказкою

Коли користувач пройде усі етапи та розв’яже задачу, то з’явиться останній кінцевий крок, у я кому знаходиться повідомлення, що завдання виконане та посилання назад до стартового вікна із вибором розділу задач.

Рисунок 4.11 – Кінцевий крок

4.2 Інструкція користувача

Для початку навчання в тренажері потрібно запустити файл «training.exe».

Оберіть розділ, який вас цікавить. Після чого виберіть задачу з переліку. У будь якому місці програми, окрім головного меню, у верхньому лівому куті знаходиться кнопка «назад», що повертає до попереднього розділу.

В розділі з розв’язком задачі вгорі можна побачити її умову. Іншу частину вікна займають кроки розв’язку. Виберіть варіант тесту або заповніть поля, щоб відповісти. Під час введення даних у поля рекомендовано використовувати кнопку «Tab» для переходу курсору у наступну комірку.

Якщо відповідь правильна, то фокус буде переведено до наступного питання. Ви завжди можете прогорнути частину вікна вгору щоб побачити минулі кроки.

Якщо відповідь неправильна, то з’явиться вікно, яке повідомить про помилку та дасть підказку.

4.3 Опис процесу програмної реалізації

Було створено проект на основі Electron. Згідно документації[], увесь код програми тренажера знаходиться в робочій директорії src (додаток А).

- Файл index.js – ініціалізація вікна та його параметри.
- Файл index.html – меню з розділами задач, що зустрічає користувача відразу після запуску.
- Файл main.js – основний код програми, що керує нею.
- Файл main.min.js – зжатий код з файлу main.js
- Файли main.scss та form.scss містять код стилів, що зроблений за допомогою препроцесору CSS
- Файл style.min.css - оброблений та зжатий код стилів мовою CSS з файлів main.scss та form.scss
- Директорія img – містить зображення, що використовуються в інтерфейсі
- Файли sum.html, product.html та comb.html – файли з розміткою меню для розділів, що ведуть безпосередньо до задач.
- Файли sum.html, product.html та comb.html – файли з розміткою меню для розділів, що ведуть безпосередньо до задач.
- Файли depended.html, independed.html, task1.html task2.html compatible.html incompatible.html – файли з розміткою задач.

Розмітка типового кроку з тестом.

```
<div class="logic-box">
   <p><b>Питання</b></p>
   <form class="test" id="id">
      <input class="text-alert" type="hidden" value="Підказка">
      <p><input name="id" type="radio" value="1"><label>Правильна</label></p>
      <p><input name="id" type="radio" value="0"><label>Хибна</label> </p>
      <p><input name="id" type="radio" value="0"><label>Хибна</label> </p>
   </form>
</div>
```

Приклад розмітки кроку з полями.

```
<div class="logic-box">
   <p><b>Питання </b></p>
   <form class="field-box math" data-type="all" data-res="42">
      <input class="text-alert" type="hidden" value="Перша підказка">
      <input class="text-alert2" type="hidden" value="Друга підказка">
      <i>C</i>
      <p class="equals">=</p>
      <input name="field" type="text" data-true="21"> //поле для введення
      <i>*</i>
      <input name="field" type="text" data-true="2"> //поле для введення
      <p class="equals">=</p>
      <input name="field" type="text" data-true="42"> //поле для введення
   </form>
</div>
```

Код програми, що керує поведінкою тренажера, написаний мовою JavaScript. Розглянемо його складові.

Знаходимо елементи та присвоюємо їх змінним.

```
let currentStep = 0 //номер поточного кроку
let steps = document.querySelectorAll(".logic-box") //масив усіх кроків
let content = document.querySelector(".content") // блок з розв’язком
let tests =  document.querySelectorAll(".test") //масив усіх тестів
let fields =  document.querySelectorAll(".field-box") //масив усіх запитань з полями
let alertOk =  document.querySelector("#alert-ok") //кнопка Добре
let popupWrapper = document.querySelector("#popup-wrapper") //вікно з помилкою
let alertText = document.querySelector('#alert') // текст вікна з помилкою
let info =  document.querySelector("#info") //кнопка Інфо
let infoText = document.querySelectorAll(".infoText") //інформація про програму
let infoOpen = false // закрита чи відкрита інформація про програму
```

Функція, що перемикає видимість інформації про програму. Також встановлюємо її виклик, якщо буде натиснута кнопка «Інфо».

```
function toggleInfo(){
   if(infoOpen){
      infoText.forEach(function (text){
         text.style.opacity = 0
         infoOpen = false
      })
   }else{
      infoText.forEach(function (text){
         text.style.opacity = 1
         infoOpen = true
      })
   }
}
if (info) info.addEventListener("click", toggleInfo)
```

Функція, що показує вікно з повідомленням про помилку. Вона приймає текст підказки, яке присвоюється вікну. Після чого встановлюється видимість вікна.

```
function openAlert(text){
   alertText.innerHTML  = text
   popupWrapper.style.display = "flex"
   setTimeout(function() {
      popupWrapper.style.opacity = 1
   }, 200);
}
```

Функція, що приховує вікно з повідомленням про помилку. В ній встановлюється невидимість вікна. Також назначаємо виклик функції, якщо буде натиснута кнопка «Добре».

```
function closeAlert(){
    popupWrapper.style.opacity = 0;
    setTimeout(function() {
        popupWrapper.style.display = "none"
    }, 200);
}
if (alertOk) alertOk.addEventListener("click", closeAlert);
```

Функція для переходу до наступного кроку. В ній визначаємо висоту блоку поточного кроку та висоту блоку content та присвоюємо їх суму як нову висоту блоку content. Після чого перемикаємо видимість наступного кроку. А завдяки деяким стилям цей блок орієнтований знизу вгору. Таким чином, з кожним новим кроком вони виникають в одному й тому ж місці, з можливістю прогорнути блок догори, щоб переглянути пройдені етапи.

```
function scrolling() {
    let height = steps[currentStep].offsetHeight //висота блоку поточного кроку
    let next = steps[++currentStep] //наступний крок
    let contentHeight = content.offsetHeight //висота усього блоку з розв’язком
    next.classList.add("visible")
    content.style.height = contentHeight + height + "px"
}
```

Функція, шо перевіряє тести. Приймає об’єкт тесту. В ній знаходимо потрібні елементи та присвоюємо їх змінним. З отриманої інформації перевіряємо, чи значення вибраного варіанту відповіді дорівнює одиниці.

Якщо так, то відповідь правильна, вона виділяється зеленим. Цей тест стає заблокованим для подальшого редагування. Після чого викликається функція, що здійснює перехід до наступного кроку.

Якщо ж відповідь хибна, то вона виділяється червоним. Далі, викликаємо функцію, що показує вікно з повідомленням про помилку, текст якого, передаємо в аргументи.

```
function checkTest(item){
   let answer = item.querySelector("input:checked"); //обраний варіант відповіді
   let inputs = item.querySelectorAll("input"); //масив усіх варіантів відповідей
   let textAlert = item.querySelector(".text-alert").value;//текст для вікна з помилкою
   setTimeout(function() {
      if(answer.value === '1'){
         answer.classList.remove("wrong");
         answer.classList.add("right");
         setTimeout(function() { scrolling()}, 500);
         inputs.forEach(function (input){
            input.disabled = true;
         })
      }else{
         answer.classList.add("wrong");
         openAlert(textAlert)
      }
   }, 500);
}
```

Призначаємо виклик вищезгаданої функції для кожного тесту, якщо в ньому буде обрано відповідь.

```
tests.forEach(function (item){
   item.addEventListener("change",() => checkTest(item))
})
```

Функція, шо перевіряє крок з полями. Приймає об’єкт цього кроку. Спочатку в ній перевіряємо, чи заповнені усі поля у цьому прикладі. Якщо так, то знаходимо потрібні елементи та присвоюємо їх змінним. Також присутня перевірка того, чи була уже введена неправильна відповідь. Якщо так, то буде обраний інший текст повідомлення про помилку. Викликаємо функцію для перевірки рівняння, та передаємо їй його тип, масив з введеними відповідями та правильну відповідь.

Якщо відповідь правильна, то поля виділяються зеленим та стають заблокованим для подальшого редагування. Після чого викликається функція, що здійснює перехід до наступного кроку.

Якщо ж відповідь хибна, то вона виділяється червоним. Введені відповіді видаляються. Далі викликаємо функцію, що показує вікно з повідомленням про помилку, текст якого, передаємо в аргументи.

```
function checkFields(item){
    let isAll = true
    let inputs = item.querySelectorAll("input[type='text']");
    inputs.forEach(function (input){
        if(input.value === ''){
            isAll = false;
        }
    });
if(isAll) {
        let type = item.dataset.type //тип прикладу
        let res = item.dataset.res //правильна відповідь
        let set = [] //масив для правильних відповідей в комірках
        let answer = [] //масив для відповідей в комірках
        let textAlert = item.querySelector(".text-alert").value //текст підказки
        if(item.classList.contains("wrong")){
            let textAlert2 = item.querySelector(".text-alert2")
            if(textAlert2) textAlert = textAlert2.value; //текст другої підказки
        }
        inputs.forEach(function (input) {
            answer.push(Number.parseFloat(input.value.replace(/,/, '.')));
            set.push(Number.parseFloat(input.dataset.true));
        });
        if(checkEquation(type, res, set, answer)){
            item.classList.remove("wrong");
            item.classList.add("right");
```

```
            inputs.forEach(function (input){
                input.disabled = true;
            })
            setTimeout(function() { scrolling()}, 500);
        }else{
            item.classList.add("wrong");
            openAlert(textAlert)
            inputs.forEach(function (input){
                input.value = null;
            })
        }
    }
```

Призначаємо виклик вищезгаданої функції для кроку з полями, якщо в ньому буде заповнена комірка.

```
fields.forEach(function (item){
    item.addEventListener("change",() =>  checkFields(item))
})
```

Функція для перевірки прикладів. Приймає тип, масив відповідей, та правильну відповідь рівняння. Повертає true, якщо відповідь правильна, та false, якщо вона хибна. В ній закодовані варіанти обчислень відповідно до типу виразу.

```
function checkEquation(type, res, set, answer){
    let userFormul = 0
    let userResult = 0
    if (type === "simple"){
        for (var i = 0; i < answer.length; i++) {
            answer[i] = Number.parseFloat(answer[i]).toFixed(2);
            set[i] = Number.parseFloat(set[i]).toFixed(2);
            if(!(answer[i] == set[i])){
                console.log(answer[i],set[i])
                return false
```

```
            }
        }
        return true
    }
    if (type === "all") {
        userFormul = (answer[0] * answer[1]) / answer[2]
        userResult = answer[3]
    }
    if (type === "inc-add") {
        userFormul = (answer[0] / answer[1]) + (answer[2] / answer[3])
        userResult = answer[4] / answer[5]
    }
    if (type === "pos") {
        userFormul = answer[0] / answer[1]
        userResult = userFormul;
    }
    if (type === "comp-add") {
        userFormul = answer[0] + answer[1] - (answer[2] * answer[3])
        userResult = answer[4]
    }
    if (type === "dep-prod") {
        userFormul = cutNum((answer[0] / answer[1]) * (answer[2] / answer[3]) *
(answer[4] / answer[5]), 2)
        userResult = cutNum(answer[6], 2)
    }
    if (type === "indep-prod") {
        userFormul = answer[0] * answer[1] * answer[2]
        userResult = answer[3]
    }
    if (type === "comb-add") {
        userFormul = answer[0] + answer[1] + answer[2]
```

```
        userResult = answer[3]
    }

    res = Number.parseFloat(res).toFixed(2);
    userFormul = Number.parseFloat(userFormul).toFixed(2);
    userResult = Number.parseFloat(userResult).toFixed(2);
    return userResult == userFormul && userFormul == res;
}
```

Функція, що перемішує варіанти відповідей та її виклик для усіх тестів.

```
function shuffle(elems) {
    allElems = (function(){
        var ret = [], l = elems.length;
        while (l--) { ret[ret.length] = elems[l]; }
        return ret;
    })();
    var shuffled = (function(){
        var l = allElems.length, ret = [];
        while (l--) {
            var random = Math.floor(Math.random() * allElems.length),
                randEl = allElems[random].cloneNode(true);
            allElems.splice(random, 1);
            ret[ret.length] = randEl;
        }
        return ret;
    })(), l = elems.length;
    while (l--) {
        elems[l].parentNode.insertBefore(shuffled[l], elems[l].nextSibling);
        elems[l].parentNode.removeChild(elems[l]);
    }
}
document.addEventListener("DOMContentLoaded", shuffleTests);
```

```
function shuffleTests(){
    tests.forEach(function (item){
        shuffle(item.querySelectorAll('p, .math-row'))
    })
}
```

4.4 Тестування програми

Створений навчальний тренажер протестований у версіях Windows7x86 та вище. Загалом, розробка відповідає висунутим до неї вимогам. Програма працює згідно з алгоритмом. А текст задач та їх кроків розв'язку узгоджується з ним. Перевірка відповідей працює справно, як і очікувалось.

Для тестування під час розробки було використано вбудований інструмент, що призначений для цього (рис 4.12).

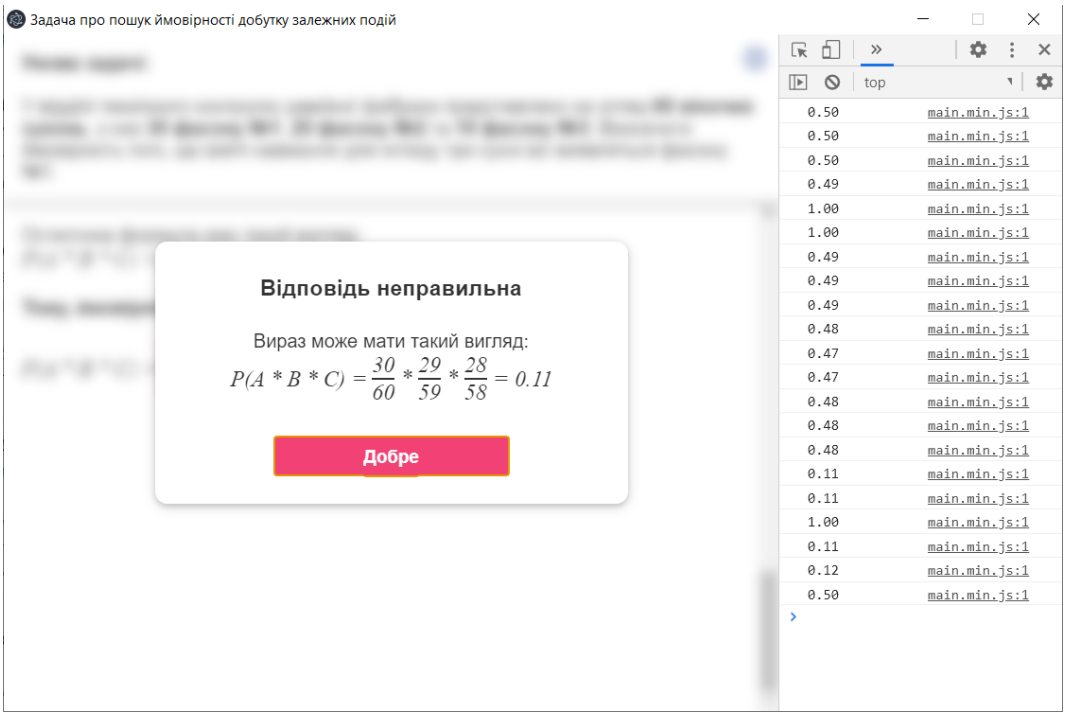


Рисунок 4.12 – Тестування програми

ВИСНОВКИ

Створено програмне забезпечення тренажера з теми «Теореми додавання та множення ймовірностей випадкових подій» для дистанційного навчального курсу «Теорія ймовірностей і математична статистика».

Були виконані наступні завдання:

- оглянуто вже наявні тренажери на дистанційному курсі,
- виконано постановку задачі,
- сформовано задачі з теми «теореми додавання та множення ймовірностей випадкових подій», що мають бути опрацьованими у тренажері,
- досліджено теоретичні відомості щодо теми,
- складено алгоритм роботи програми,
- створено блок-схему роботи складеного алгоритму,
- визначено засоби розробки,
- розроблено навчальний тренажер,
- протестовано розробку,
- проаналізовано роботу та написано звіт.

Проект дотримався усіх поставлених вимог:

- огляд теоретичних відомостей з теми «теореми додавання та множення ймовірностей випадкових подій»,
- покроковий та інтерактивний розв'язок типових задач з теми «теореми додавання та множення ймовірностей випадкових подій» з поясненням усіх етапів та перевіркою знань користувача,
- у разі помилки повідомити користувача та наштовхнути його на правильну відповідь,
- робота на популярних операційних системах університету, таких як windows7x86, із зворотною сумісністю та можливістю змінити платформу,
- сучасний та естетично привабливий інтерфейс, який би не відлякував користувачів і водночас не відволікав їх від навчання.

СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Ємець О.О. Методичні рекомендації до виконання бакалаврської роботи для студентів спеціальності 122 «Комп'ютерні науки та інформаційні технології» освітня програма «Комп'ютерні науки» галузь знань – 12 «Інформаційні технології»/ О.О. Ємець,–Полтава; ПУЕТ, 2017, - 71 с.
2. Grant Keith J. CSS in Depth / Keith J. Grant – Manning Publications Co., 2018. – 472 с. – ISBN 9781617293450.
3. Kinney S. Electron in Action / S. Kinney – Manning Publications Co., 2018. – 376 с. – ISBN 9781617294143.
4. Meyer J. HTML5 and JavaScript Projects / J. Meyer – Apress Media, 2018. – 425 с. – ISBN 978-1-4842-3864-6.
5. Огірко О. І. Теорія ймовірностей та математична статистика: навчальний посібник / О. І. Огірко, Н. В. Галайко. – Львів: ЛьвДУВС, 2017. – 292 с.
6. Ємець О.О. Теорія ймовірностей і математична статистика. Конспект лекцій. Ч.1. Елементи теорії ймовірностей.: навч. Посібник / Укл. Ємець. – Полтава, 2001. – 34 с.
7. Белінська В. В. Створення програмного забезпечення тренажера з теми «Розподіли дискретних випадкових величин та їх числові характеристики» дистанційного навчального курсу «Теорія ймовірностей та математична статистика»/ В. В. Белінська, Т. О Парфьонова. – Полтава: Кафедра ММСІ ПУЕТ, 2021. – 63 с. – Режим доступу: http://dspace.puet.edu.ua/handle/123456789/10305
8. Жайворонок Я. І. Розробка програмного забезпечення для тренажера дистанційного навчального курсу «Теорія інформації та кодування» з теми «Коди із виявленням помилок»/ Я. І. Жайворонок, Т. О Парфьонова. – Полтава: Кафедра ММСІ ПУЕТ, 2021. – 113 с. – Режим доступу: http://dspace.puet.edu.ua/handle/123456789/10045

ДОДАТОК А

Алгоритми задач

Розглянемо алгоритм розв’язку задачі про пошук ймовірності добутку залежних подій. Він складається з 8 кроків.

Далі в описі алгоритму в тестових запитаннях лише перша відповідь правильна. В тренажері відповіді перемішуються.

Умова задачі: у відділі технічного контролю швейної фабрики представлено на огляд 60 жіночих суконь, з них 30 фасону №1, 20 фасону №2 та 10 фасону №3. Визначити ймовірність того, що взяті навмання для огляду три сукні всі виявляться фасону №1.

Крок 1. Нехай:

Подія *X* – всі сукні виявляться фасону №1

Подія *A* – перша сукня виявляться фасону №1

Подія *B* – друга сукня виявляться фасону №1

Подія *C* – третя сукня виявляться фасону №1

Згадаємо деякі теоретичні відомості.

Добутком двох подій *A* та *B* називають подію, що полягає у появі:

- обох цих подій одночасно,
- події *A* або події *B*,
- події *B* за умови, що відбулась подія *A*.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Добутком двох подій *A* та *B* називають подію, що полягає у появі обох цих подій одночасно».

Крок 2. В заданій задачі події *A*, *B* та *C*:

- залежні
- незалежні

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Події *A*, *B* та *C* залежні, тому що кожна витягнута сукня впливає на ймовірність витягти наступні сукні певного фасону».

Крок 3. Отже, оскільки події *A*, *B* та *C* залежні, то подію *X* можна виразити через події *A*, *B* та *C* таким чином:

- $X = A * B * C$
- $X = A + B + C$
- $X = A * B_A * C_{AB}$
- $X = A + B_A + C_{AB}$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Подію *X* можна виразити як добуток подій *A*, *B* та *C*».

Крок 4. Ймовірність добутку подій *A*, *B* та *C*:

- $P(A * B * C) = P(A) * P_A(B) * P_{AB}\left(C\right)$
- $P(A * B * C) = P(A) * P(B_A) * P\left(C_{AB}\right)$
- $P(A * B * C) = P(A) * P(B) * P\left(C\right)$
- $P(A * B * C) = A * B_A * C_{AB}$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Ймовірність добутку подій *A*, *B* та *C* дорівнює добутку умовних ймовірностей цих подій, тобто: $P(A * B * C) = P(A) * P_A(B) * P_{AB}\left(C\right)$, де ймовірність події *B* залежить від події *A*, а ймовірність події *C* залежить від *A* та *B*».

Крок 5. Знайдемо ймовірності кожної події. Ймовірність події *A*:

$$P(A) = \frac{\square}{\square}$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «За умовою задачі відомо, що перед вибором першої сукні є 30 суконь фасону №1 з усього 60 суконь. Отже ймовірність події *A* дорівнює відношенню кількості суконь фасону №1 до загальної кількості суконь».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(A) = \frac{30}{60}$ ».

Крок 6. Ймовірність події *B*:

$$P_A(B) = \frac{\square}{\square}$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Відомо, що перед вибором другої сукні було вже взято 1 сукню фасону №1. Враховуючи це, є 29 суконь фасону №1 з усього 59 суконь. Отже, ймовірність події *B* дорівнює відношенню кількості суконь фасону №1 до загальної кількості суконь».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P_A(B) = \frac{29}{59}$ ».

Крок 7. Ймовірність події *C*:

$$P_{AB}(C) = \frac{\square}{\square}$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Відомо, що перед вибором третьої сукні було вже взято 2 сукні фасону №1. Враховуючи це, є 28 суконь фасону №1 з усього 58 суконь. Отже ймовірність події B дорівнює відношенню кількості суконь фасону №1 до загальної кількості суконь».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P_{AB}(C) = \frac{28}{58}$ ».

Крок 8. Остаточна формула має такий вигляд:

$$P(A*B*C) = P(A)*P_A(B)*P_{AB}(C).$$

Тому, ймовірність події X обчислюється так:

$$P(A*B*C) = \frac{\square}{\square} * \frac{\square}{\square} * \frac{\square}{\square} = \frac{\square}{\square} = \square$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Впишіть знайдені ймовірності у комірки та розрахуйте результат виразу».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(A*B*C)=\frac{30}{60}*\frac{29}{59}*\frac{28}{58}=\frac{24360}{205320}=0.11$».

Розглянемо алгоритм розв'язку задачі про пошук ймовірності добутку незалежних подій. Він складається з 7 кроків.

Далі в описі алгоритму в тестових запитаннях лише перша відповідь правильна. В тренажері відповіді перемішуються.

Умова задачі: Три студенти одночасно складають іспит. Ймовірність складання іспиту на «відмінно» першим студентом – 0,5; другим – 0,7; третім – 0,4. Знайти ймовірність того, що три студенти складуть іспит на «відмінно».

Крок 1. Нехай:

Подія *X* – всі студенти складуть іспит на «відмінно»

Подія *A* – перший студент склав іспит на «відмінно»

Подія *B* – другий студент склав іспит на «відмінно»

Подія *C* – третій студент склав іспит на «відмінно»

Згадаємо деякі теоретичні відомості.

Випадкові події *A* та *B* називаються сумісними, якщо:

- поява однієї події не виключає появи іншої.
- поява однієї події виключає появу іншої.
- поява однієї події залежить від появи іншої.
- поява однієї події відбувається лише одночасно з появою іншої.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Випадкові події *A* та *B* називаються сумісними, якщо поява однієї події не виключає появи іншої».

Крок 2. В заданій задачі події *A*, *B* та *C*:

- сумісні
- несумісні

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Події *A*, *B* та *C* сумісні, так як іспит одного студента не виключає іспит іншого».

Крок 3. Добутком двох подій *A* та *B* називають подію, що полягає у появі:

· в одночасній появі цих подій.

· обох цих подій.

· події *A* або події *B*.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Добутком двох подій *A* та *B* називають подію, що полягає у появі обох цих подій одночасно».

Крок 4. В заданій задачі події *A*, *B* та *C*:

· незалежні

· залежні

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Події *A*, *B* та *C* незалежні, тому що ймовірність скласти іспит кожного студента не залежить від того склали інші студенти іспит чи ні».

Крок 5. Отже, оскільки події *A*, *B* та *C* незалежні між собою, то подію X можна виразити через події *A*, *B* та *C* таким чином:

· $X = A * B * C$

· $X = A + B + C$

· $X = A * B_A * C_{AB}$

· $X = A + B_A + C_{AB}$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Подію *X* можна виразити як добуток незалежних подій *A*, *B* і *C*».

Крок 6. ймовірність добутку подій *A*, *B* та *C*:

· $P(A * B * C) = P(A) * P(B) * P(C)$

· $P(A * B * C) = P(A) * P(B_A) * P(C_{AB})$

· $P(A * B * C) = P(A) + P(B) + P(C)$

· $P(A * B * C) = A * B * C$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Ймовірність добутку подій *A*, *B* та *C* дорівнює добутку ймовірностей цих подій».

Крок 7. Отже, формула має такий вигляд:

$P(A*B*C) = P(A)*P(B)*P(C)$

Тому, ймовірність події *X* обчислюється так:

$P(A*B*C) = \square * \square * \square = \square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Впишіть ймовірності з умови задачі у комірки та розрахуйте результат виразу».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(A*B*C) = 0,5*0,7*0,4 = 0,14$».

Розглянемо алгоритм розв’язку задачі про пошук ймовірності комбінуванням теорем суми та добутку подій. Він складається з 16 кроків.

Далі в описі алгоритму в тестових запитаннях лише перша відповідь правильна. В тренажері відповіді перемішуються.

Умова задачі: Три студенти одночасно складають іспит. Ймовірність складання іспиту на «відмінно» першим студентом – 0,5; другим – 0,7; третім – 0,4. Знайти ймовірність того, що двоє із цих студентів складуть іспит на «відмінно».

Крок 1. Нехай:

Подія *X* – два студенти складуть іспит на «відмінно»

Подія *A* – перший студент склав іспит на «відмінно»

Подія *B* – другий студент склав іспит на «відмінно»

Подія *C* – третій студент склав іспит на «відмінно»

Подія $\overline{A}$ – перший студент не склав іспит на «відмінно»

Подія $\overline{B}$ – другий студент не склав іспит на «відмінно»

Подія $\overline{C}$ – третій студент не склав іспит на «відмінно»

Введіть значення ймовірностей згідно умови і введених позначень:

$P(A) = \square$, $P(B) = \square$, $P(C) = \square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Ймовірності можна записати так: $P(A) = 0,5$, $P(B) = 0,7$, $P(C) = 0,4$».

Крок 2. Введемо такі події:

Подія D_1 – другий та третій студенти склали іспит на «відмінно», перший – ні

Подія D_2 – перший та третій студенти склали іспит на «відмінно», другий – ні

Подія D_3 – перший та другий студенти склали іспит на «відмінно», третій – ні

Для того щоб знайти ймовірність непояви події С згадаємо наступну інформацію.

Випадкові події називаються протилежними, якщо:

- несумісні і утворюють повну групу
- незалежні і утворюють повну групу

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «За означенням випадкові події називаються протилежними, якщо несумісні і утворюють повну групу».

Крок 3. Подія $\bar{C}$ протилежна до події С.

Отже, згідно з означенням, ймовірність події $\bar{C}$ можна виразити наступним чином:

- $P(\bar{C}) = 1 - P(C)$
- $P(\bar{C}) = P(C) - 1$
- $P(\bar{C}) = 1 + P(C)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Ймовірність події $\bar{C}$ можна виразити як різницю 1 та ймовірності події С, тобто $P(\bar{C}) = 1 - P(C)$».

Крок 4. Аналогічно подія $\overline{A}$ протилежна до події A, $\overline{B}$ протилежна до події B. Їхні ймовірності можна виразити як:

$P(\overline{A}) = \square$, $P(\overline{B}) = \square$, $P(\overline{C}) = \square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Ймовірності можна записати так: $P(\overline{A}) = 0,5$, $P(\overline{B}) = 0,3$, $P(\overline{C}) = 0,4$».

Крок 5. Згадаємо деякі теоретичні відомості.

Добутком двох подій *A* та *B* називають подію, що полягає у появі:

- в одночасній появі цих подій.
- обох цих подій.
- події *A* або події *B*.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Добутком двох подій *A* та *B* називають подію, що полягає у появі обох цих подій одночасно».

Крок 6. В заданій задачі події *A*, *B* та *C*:

- незалежні
- залежні

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Події *A*, *B* та *C* незалежні, тому що ймовірність скласти іспит кожного студента не залежить від того склали інші студенти іспит чи ні».

Крок 7. Отже, оскільки події *A*, *B* та *C* незалежні, а $\overline{C}$ протилежна до *C*, то подію D_3 можна виразити таким чином:

- $D_3 = A * B * \overline{C}$
- $D_3 = A * B * (1 - C)$
- $D_3 = A * B_A * \overline{C}_{AB}$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Подію D_3 можна виразити як добуток незалежних подій *A*, *B* та $\overline{C}$».

Крок 8. Аналогічно попередньому кроку, можна виразити події D_1 та D_2:

$D_1 = \bar{A} * B * C$,

$D_2 = A * \bar{B} * C$.

А формула ймовірності добутку подій *A*, *B* та $\bar{C}$:

- $P(A * B * \bar{C}) = P(A) * P(B) * (1 - P(C))$
- $P(A * B * \bar{C}) = P(A) * P(B_A) * (1 - P(C_{AB}))$
- $P(A * B * \bar{C}) = P(A) + P(B) + P(\bar{C})$
- $P(A * B * \bar{C}) = A * B * (1 - C)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Ймовірність добутку подій *A*, *B* та $\bar{C}$ дорівнює добутку ймовірностей цих подій, де $P(\bar{C}) = 1 - P(C)$».

Крок 9. Аналогічно попередньому кроку знаходимо ймовірність події D_1 та D_2:

$P(D_1) = P(\bar{A} * B * C) = (1 - P(A)) * P(B) * P(C)$,
$P(D_2) = P(A * \bar{B} * C) = P(A) * (1 - P(B)) * P(C)$.

Використаємо знайдені формули та обчислимо ймовірності.

Ймовірність події D_1:

$P(D_1) = \square * \square * \square = \square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Обчисліть та впишіть ймовірності у комірки та розрахуйте результат виразу».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(D_1) = 0,5 * 0,7 * 0,4 = 0,14$».

Крок 10. Ймовірність події D_2:

$P(D_2) = \square * \square * \square = \square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Обчисліть та впишіть ймовірності у комірки та розрахуйте результат виразу».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(D_2) = 0,5 * 0,3 * 0,4 = 0,06$».

Крок 11. Ймовірність події D_3:

$P(D_3) = \square * \square * \square = \square$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Обчисліть та впишіть ймовірності у комірки та розрахуйте результат виразу».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(D_3) = 0,5 * 0,7 * 0,6 = 0,21$».

Крок 12. Для того щоб виразити та обчислити подію *X* пригадаємо наступне.

Випадкові події *A* та *B* називаються сумісними, якщо:

- поява однієї події не виключає появи іншої.
- поява однієї події виключає появу іншої.
- поява однієї події залежить від появи іншої.
- поява однієї події відбувається лише одночасно з появою іншої.

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Випадкові події *A* та *B* називаються сумісними, якщо поява однієї події не виключає появи іншої».

Крок 13. В заданій задачі події D_1, D_2 та D_3:

- несумісні
- сумісні

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Події D_1, D_2 та D_3 несумісні, так як вони виключають одна одну».

Крок 14 . Отже, оскільки події D_1, D_2 та D_3 несумісні, то подію *X* можна виразити через події D_1, D_2 та D_3 таким чином:

- $X = D_1 + D_2 + D_3$
- $X = D_1 * D_2 * D_3$
- $X = D_1 + D_2 + D_3 - (D_1 * D_2 * D_3)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Подію X можна виразити як суму подій D_1, D_2 та D_3».

Крок 15. Тому ймовірність суми подій D_1, D_2 та D_3:

- $P(D_1 + D_2 + D_3) = P(D_1) + P(D_2) + P(D_3)$
- $P(D_1 + D_2 + D_3) = P(D_1) * P(D_2) * P(D_3)$
- $P(D_1 + D_2 + D_3) = P(D_1) + P(D_2) + P(D_3) - P(D_1 + D_2 + D_3)$
- $P(D_1 + D_2 + D_3) = P(D_1 + D_2 + D_3) - P(D_1 * D_2 * D_3)$

Якщо у цьому кроці користувач вибере неправильну відповідь, то з'явиться вікно із текстом «Ймовірність суми подій D_1, D_2 та D_3 дорівнює сумі ймовірностей цих подій».

Крок 16. Отже, формула має такий вигляд:

$$P(D_1 + D_2 + D_3) = P(D_1) + P(D_2) + P(D_3)$$

Тому, ймовірність події *X* обчислюється так:

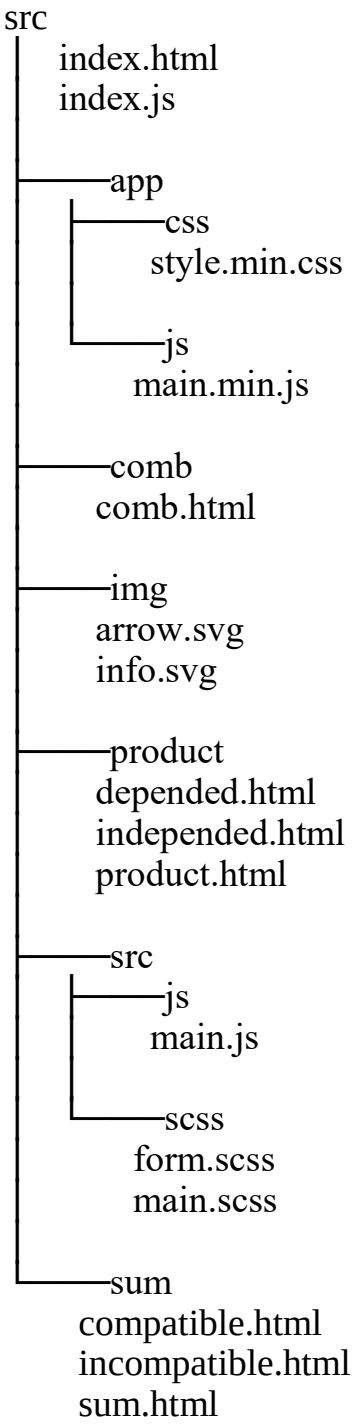
$$P(X) = P(D_1 + D_2 + D_3) = \square + \square + \square = \square$$

Якщо у цьому кроці користувач впише неправильну відповідь, то з'явиться вікно із текстом «Впишіть знайдені ймовірності у комірки та розрахуйте результат виразу».

Якщо у цьому кроці користувач впише неправильну відповідь вдруге, то з'явиться вікно із текстом «Вираз може мати такий вигляд: $P(X) = 0,14 + 0,06 + 0,21 = 0,41$».

ДОДАТОК Б

Файлова структура проекту



ДОДАТОК В
gulpfile.js

```
var gulp = require('gulp');
var sass = require('gulp-sass');
var watch = require('gulp-watch');
var autoprefixer = require('gulp-autoprefixer');
var concat = require('gulp-concat');
var rename = require("gulp-rename");
var uglify = require('gulp-uglify');
var minify = require('gulp-minify-css');
gulp.task('scss', function(done) {
	gulp.src(["app/src/scss/main.scss","app/src/scss/*.scss"])
	.pipe(concat("style.scss"))
	.pipe(sass().on('error', sass.logError))
	.pipe(autoprefixer({
		overrideBrowserslist: ['last 2 versions'],
		cascade: false
	}))
	.pipe(rename({suffix: ".min"}))
	.pipe(minify())
	.pipe(gulp.dest("app/app/css"));
	done();
});
gulp.task('minjs', function(done) {
	gulp.src("app/src/js/*.js")
	.pipe(rename({suffix: ".min"}))
	.pipe(uglify())
	.pipe(gulp.dest("app/app/js"));
	done();
})
gulp.task('watch', function(){
	gulp.watch("app/src/scss/*.*", gulp.series('scss'));
});
gulp.task('default', gulp.series('watch'));
```