МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**Український державний університет науки і технологій**

Кафедра Комп’ютерні інформаційні технології

«ДО ЗАХИСТУ»

Завідувач кафедри

 \_\_\_\_\_\_\_\_ Вадим ГОРЯЧКІН

 (підпис) (ПІБ)

2022р.\_\_\_\_\_\_\_ «\_\_\_\_\_»

ДИПЛОМНА РОБОТА

на здобуття ОС «бакалавр»

Галузь знань 12 Інформаційні технології

Спеціальність 121 Інженерія програмного забезпечення

Освітньо-професійна програма \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(повна назва)

Тема «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

Theme «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»

(theme in English)

Керівник дипломної роботи \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(посада) (підпис) (ПІБ)

Нормоконтролер \_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 (посада)       (підпис) (ПІБ)

Студент групи \_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

   (номер групи) (підпис) (ПІБ)

Student \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Family name)

Дніпро 2022

**Український державний університет науки і технологій**

Факультет “Комп’ютерні технології і системи”

Кафедра “Комп’ютерні інформаційні технології”

Спеціальність “Інженерія програмного забезпечення”

«ДО ЗАХИСТУ»

Завідувач кафедри

\_\_\_\_\_\_\_\_ Вадим ГОРЯЧКІН

 (підпис) (ПІБ)

2022р.\_\_\_\_\_\_\_ «\_\_\_\_\_»

ЗАВДАННЯ

до дипломної роботи на на здобуття ОС «бакалавр»

студента групи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(номер групи) (ПІБ)

1 Тема дипломної роботи: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ затверджена наказом по університету від «\_\_\_» \_\_\_\_\_\_\_\_ 201\_\_ р. № \_\_.

2 Термін подання студентом закінченої роботи\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3 Вихідні дані до дипломної роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4 Зміст пояснювальної записки (перелік питань до розробки) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5 Перелік демонстраційного матеріалу \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Реферат**

Пояснювальна записка складається з 8 розділів:

– вступ – в даному розділі описується сутність розробки, її актуальність. Складається з 1 сторінки;

– збір вимог до програмного забезпечення – у цьому розділі описуються аналоги програми та література по даній предметній області, а також проводиться опитування зацікавлених сторін для формування найбільш повних вимог до програмного забезпечення. Складається з N сторінок;

– зовнішнє і внутрішнє проектування – у цьому розділі проведений огляд вхідних і вихідних даних,формалізація задачі, розробка фізичного проекту, приводиться опис об’єктно-орієнтованого проектування, проектування інтерфейсу користувача, ескізи форм, аналіз проекту, проектування динаміки системи, вибір мови програмування. Складається з N сторінок;

– тестування та налагодження – включає в себе вибір стратегії тестування, опис тестів методами «чорного» та «білого» ящика. Також аналіз помилок їх вплив на систему та вирішення проблеми. Складається з 36 сторінок;

– охорона праці. Складається з N сторінок;

– висновки. Складається з 1 сторінки;

– список літератури – включає в себе бібліографічний список використаної літератури. Складає 2 сторінки;

– додатки – містить технічне завдання і робочий проект. Кількість таблиць: N штук. Кількість рисунків: N штук.

Ключові слова: економіка, макроекономіка, ефективна економічна модель, моделювання, налог, зарплата, економічна діяльність.

ЗМІСТ

[Перелік умовних познак, символів, скорочень і термінів 6](#_Toc103050481)

[Вступ 7](#_Toc103050482)

[1 Збір та аналіз вимог 8](#_Toc103050483)

[1.1 Опис аналогів 10](#_Toc103050484)

[2 Зовнішнє проектування 13](#_Toc103050485)

[2.1 Функціональне призначення 13](#_Toc103050486)

[2.2 Експлуатаційне призначення 13](#_Toc103050487)

[2.3 Функціональні вимоги 13](#_Toc103050488)

[2.4 Загальні вхідні дані 13](#_Toc103050489)

[2.5 Суб’єкти економічної діяльності 14](#_Toc103050490)

[2.5.1. Виробники сировини (ВС) 14](#_Toc103050491)

[2.5.2. Виробники засобів виробництва (ВЗВ) 14](#_Toc103050492)

[2.5.3. Виробники споживчих товарів (ВСТ) 15](#_Toc103050493)

[2.5.4. Торгівля та Послуги (ТП) 15](#_Toc103050494)

[2.5.5. Покупці споживчих товарів (ПСТ) 15](#_Toc103050495)

[2.5.6. Бюджет (Держава) (Б) 15](#_Toc103050496)

[2.5.7. Комерційні банки (КБ) 15](#_Toc103050497)

[2.5.8. Нацбанк (НБ) 15](#_Toc103050498)

[3 ВНУТРІШНЄ ПРОЕКТУВАННЯ 16](#_Toc103050499)

[3.1 Об’єктно-орієнтоване програмування 16](#_Toc103050500)

[3.2 Опис відповідальності класів 18](#_Toc103050501)

[3.3 Опис зв’язків класів 19](#_Toc103050502)

[3.4 Діаграма класів програми 19](#_Toc103050503)

[3.5 Опис специфікацій класів 19](#_Toc103050504)

[3.5.1 Опис специфікацій класу Program 19](#_Toc103050505)

[3.5.2 Опис специфікацій класу Form1 20](#_Toc103050506)

[3.5.3 Опис специфікацій класу Form2 20](#_Toc103050507)

[3.5.4 Опис специфікацій класу Form3 20](#_Toc103050508)

[3.5.5 Опис специфікацій класу Form4 20](#_Toc103050509)

[3.5.6 Опис специфікацій класу Form5 21](#_Toc103050510)

[3.5.7 Опис специфікацій класу Form6 21](#_Toc103050511)

[3.5.8 Опис специфікацій класу Form7 21](#_Toc103050512)

[3.5.9 Опис специфікацій класу Form8 21](#_Toc103050513)

[3.5.10 Опис специфікацій класу Form9 21](#_Toc103050514)

[3.5.11 Опис специфікацій класу InputData 22](#_Toc103050515)

[3.5.12 Опис специфікацій класу Сharacteristics 23](#_Toc103050516)

[3.5.13 Опис специфікацій класу Economy\_events 24](#_Toc103050517)

[3.6 Опис інтерфейсної частини класів 24](#_Toc103050518)

[3.6.1 Клас Program 24](#_Toc103050519)

[3.6.2 Клас Form1 24](#_Toc103050520)

[3.6.3 Клас Form2 27](#_Toc103050521)

[3.6.4 Клас Form3 28](#_Toc103050522)

[3.6.5 Клас Form4 28](#_Toc103050523)

[3.6.6 Клас Form5 29](#_Toc103050524)

[3.6.7 Клас Form6 29](#_Toc103050525)

[3.6.8 Клас Form7 30](#_Toc103050526)

[3.6.9 Клас Form8 30](#_Toc103050527)

[3.6.10 Клас Form9 31](#_Toc103050528)

[3.7 Проектування інтерфейсу користувача 31](#_Toc103050529)

[3.8 Аналіз проекту 33](#_Toc103050530)

[3.9 Вибір мови програмування 33](#_Toc103050531)

# Перелік умовних познак, символів, скорочень і термінів

# Вступ

Ця робота спрямована на програмне моделювання високоефективної макроекономічної системи, яка повністю співпадає з тим, що пропонує сучасна економічна теорія, оскільки вона функціонує на основі таких самих економічних принципів.

Мета роботи полягає у створенні додатку, яка виконує моделювання макроекономічної системи на основі дослідження фінансових потоків у макроекономіці між різними групами виробників благ і споживачів для кращого розуміння кінцевого результату їхньої діяльності.

Матеріали та методи. Людина є ключовою фігурою економіки та її кінцевим споживачем. Завдяки людині здійснюється науково-технічний прогрес та розвивається економіка. Людина – це не «трудовий ресурс», не «робоча сила» і не «гвинтик», а основна дійова особа економічного процесу, для задоволення потреб якого все, власне, і робиться. Тому економіка має бути підпорядкована інтересам людини, а не людина має бути підпорядкований економіці. Вся економіка заснована лише на людській праці. Завдяки праці створюються всі блага, необхідні для людського життя: будівлі, машини, обладнання, предмети споживання та багато іншого. Людина, як біологічний вид, для свого повноцінного існування, обов'язково має харчуватися, носити одяг, мати житло, а також користуватися іншими різноманітними матеріальними та культурними благами. Кожна людина, незалежно від того, в якій сфері працює і що виробляє врешті-решт отримує за свій працю заробітну плату, яку потім має обміняти на необхідні йому споживчі блага.

Практичне призначення розроблюваного ПЗ полягає в тому, щоб змоделювати макроекономічні системи на базі ринкової моделі.

# 1 Збір та аналіз вимог

Мета встановлення вимог полягає в тому, щоб дати розгорнуте визначення функціональних - а також не функціональних вимог, які учасники проекту очікують затвердити в системі, що реалізується та розгортається.

Методи збору вимог до програмного забезпечення При вивченні предметної області та збору вимог про майбутній об’єкт інформатизації використовується методологія системного аналізу. На цій стадії розробники повинні уточнити межі вивчення функціональних вимог до програми, вхідні та вихідні дані для функціонування програми та визначити коло користувачів майбутньої програми.

Метод бесіди – психологічний вербально-комунікативний метод, що полягає у веденні тематично спрямованого діалогу між дослідником і респондентом з метою отримання відомостей від останнього. Метод бесіди і консультацій найчастіше проводиться з керівниками підприємств і підрозділів або у формі ділової консультації з фахівцями з питань, що 40 мають глобальний характер і належать до визначення проблем і стратегій розвитку та управління підприємством.

Прототипування (prototyping) – це найбільш часто використовуваний сучасний метод виявлення вимог. Програмні прототипи конструюються для візуалізації системи або її частини для замовників з метою отримання їх відгуків. Прототип – це дуже ефективний спосіб виявлення вимог, які важко отримати від замовника за допомогою інших засобів.

Еволюційний прототип (evolutionary), який зберігається після виявлення вимог і використовується для створення кінцевого програмного продукту. Еволюційний прототип націлений на прискорення поставок товарів. Як правило, він концентрується на ясно викладених вимогах, тому першу версію продукту можна надати замовнику досить швидко (хоча її функціональні можливості, як правило, неповні)

При виконанні роботи були використані такі методи збору та аналізу вимог як метод діалогу та метод еволюційного прототипування. Комбінування методу діалогу та еволюційного прототипування дозволило швидко та зручно визначити вимоги та розробляти програму крок за кроком, поки усі вимоги не були покриті та погоджені із замовником.

Після збору та аналізу вимог, зібрана інформація була проаналізована і на її основі було розроблено бізнес-процес системи.

Бізнес процес для моделювання макроекономічної системи виконується наступним чином:

1. Користувач задає вхідні дані;

2. Відбувається моделювання системи із виводом результатів на екран у вигляді діаграм;

3. Запис усіх розрахунків до файлу.

Бізнес процес є завершеним, так як користувач отримав усю необхідну інформацію про змодельовану систему у вигляді діаграм та у виглядів чистих розрахунків.

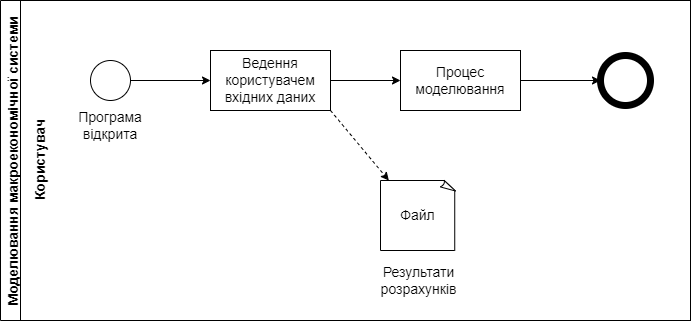


Рисунок 1. - Бізнес-процес моделювання макроекономічної системи

## 1.1 Опис аналогів

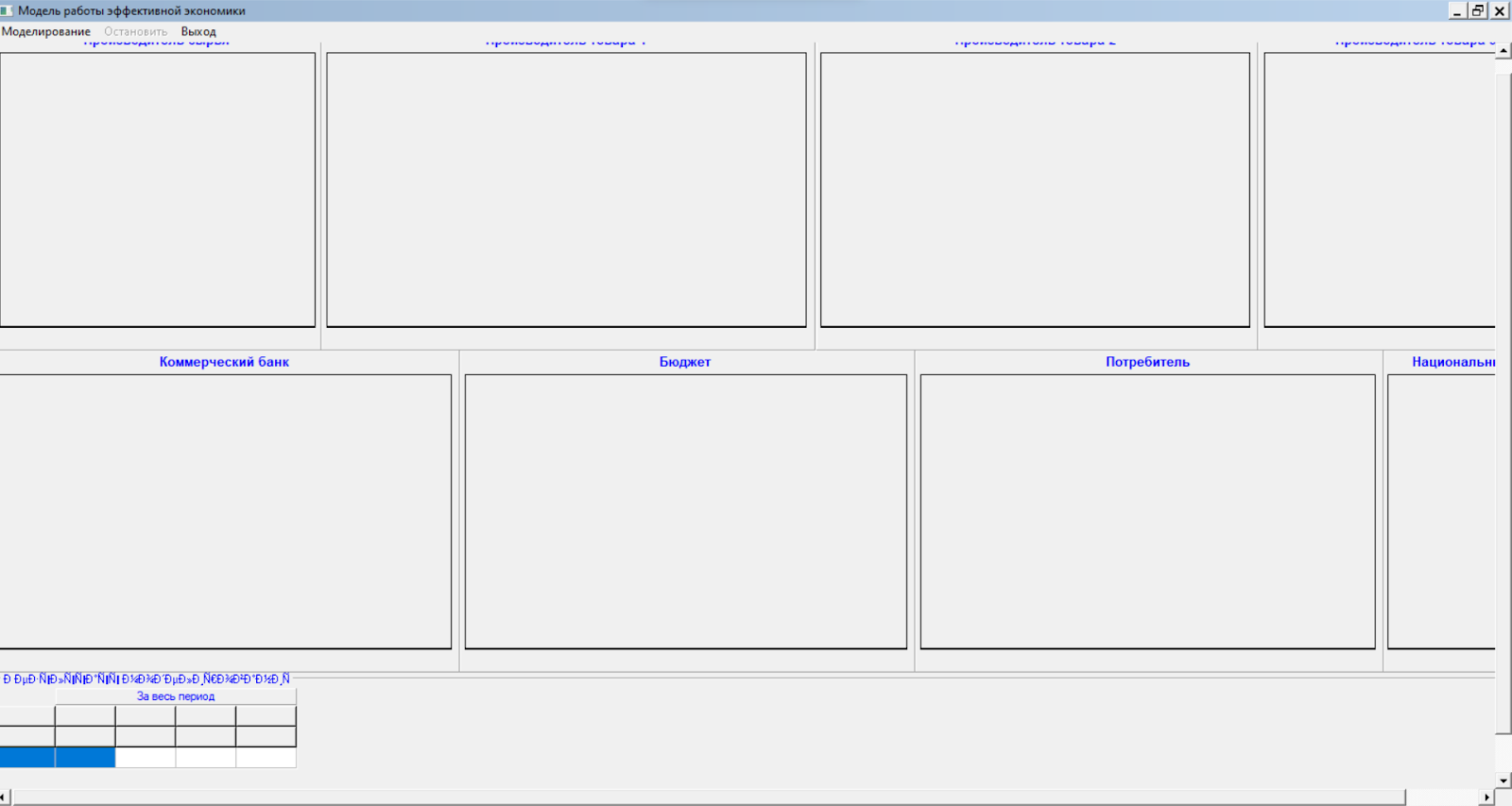
Існує тільки один аналог для моделювання макроекономічної моделі на базі безкризової економічної моделі та на базі класичної ринкової моделі - **SimpleModel**.

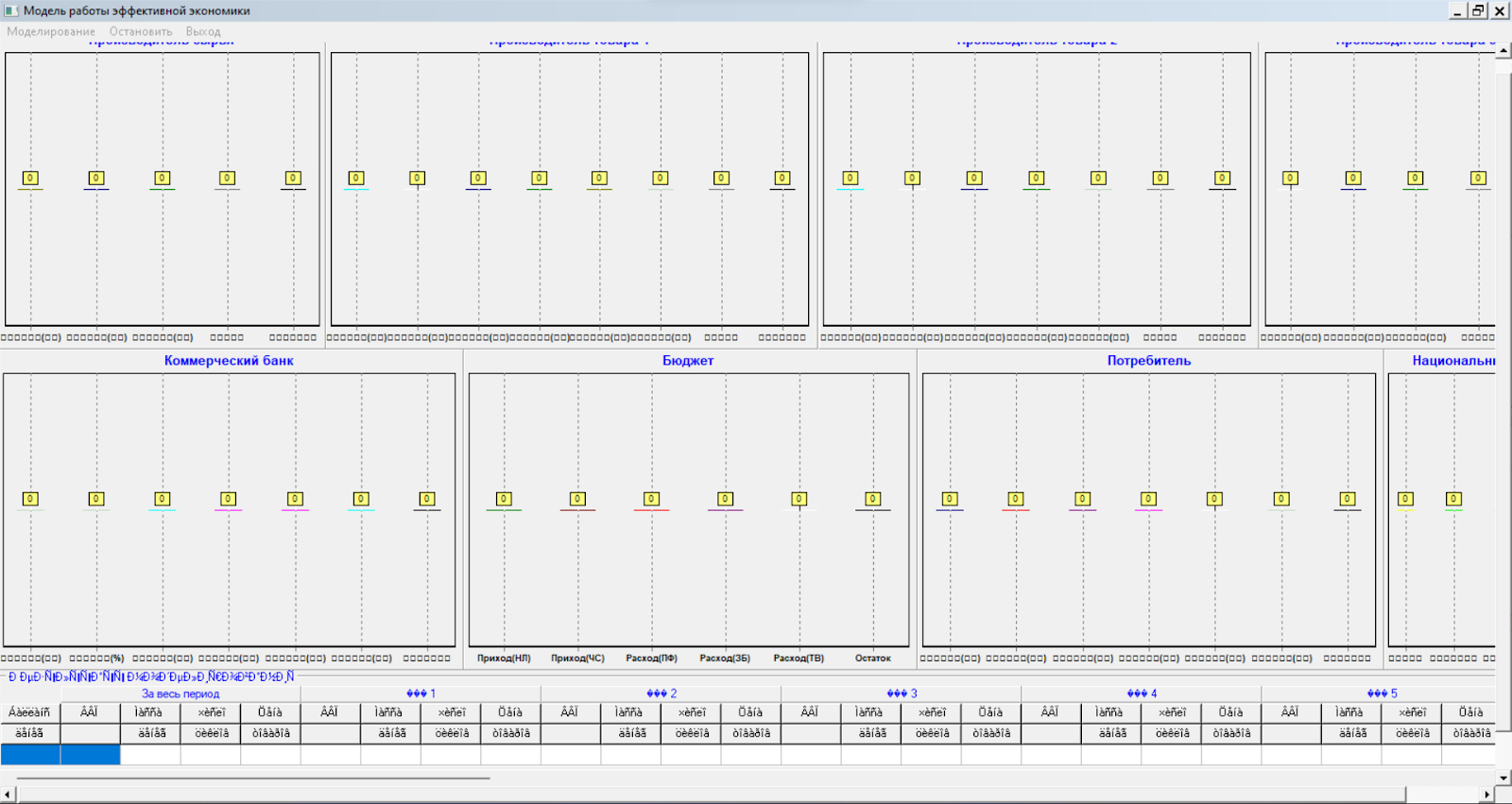
SimpleModel - перша версія додатку, яка була розроблена з метою моделювання макроекономічної системи.

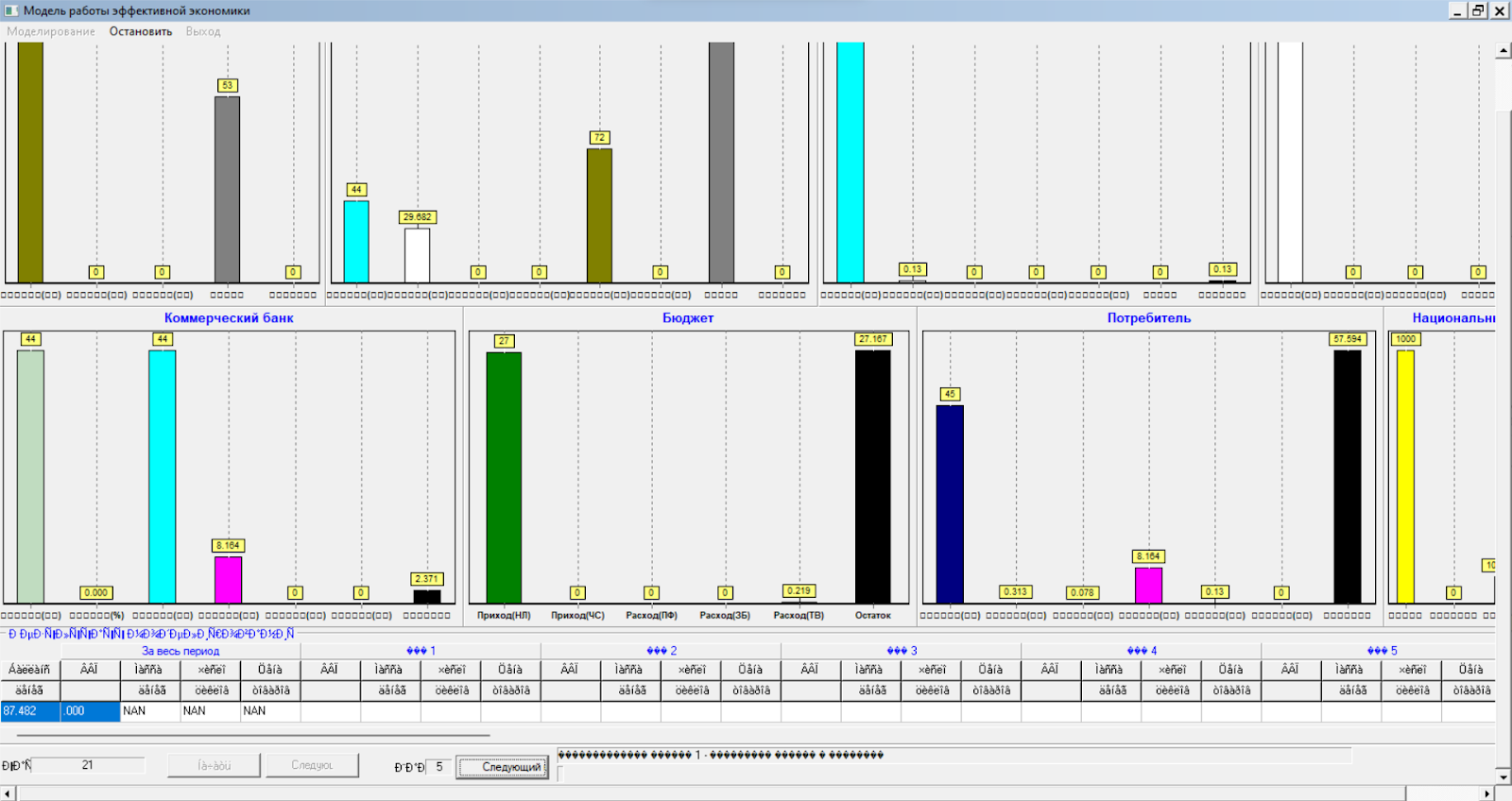
Головними перевагами цієї програми є те, що вона виконує свою головну функцію - виконує моделювання макроекономічної системи на базі ринкової моделі та на базі безкризової економічної моделі. Також вона має простий та зручний для користувача дизайн.

Недоліками програми є те, що всі вхідні дані програми вбудовані у програмний код, що не дає можливості змоделювати макроекономічну систему при різних даних. Також програма була розроблена з прив’язкою до старих систем Windows XP, Windows 7, що не дає гарантій на безвідказну та надійну роботу програми.

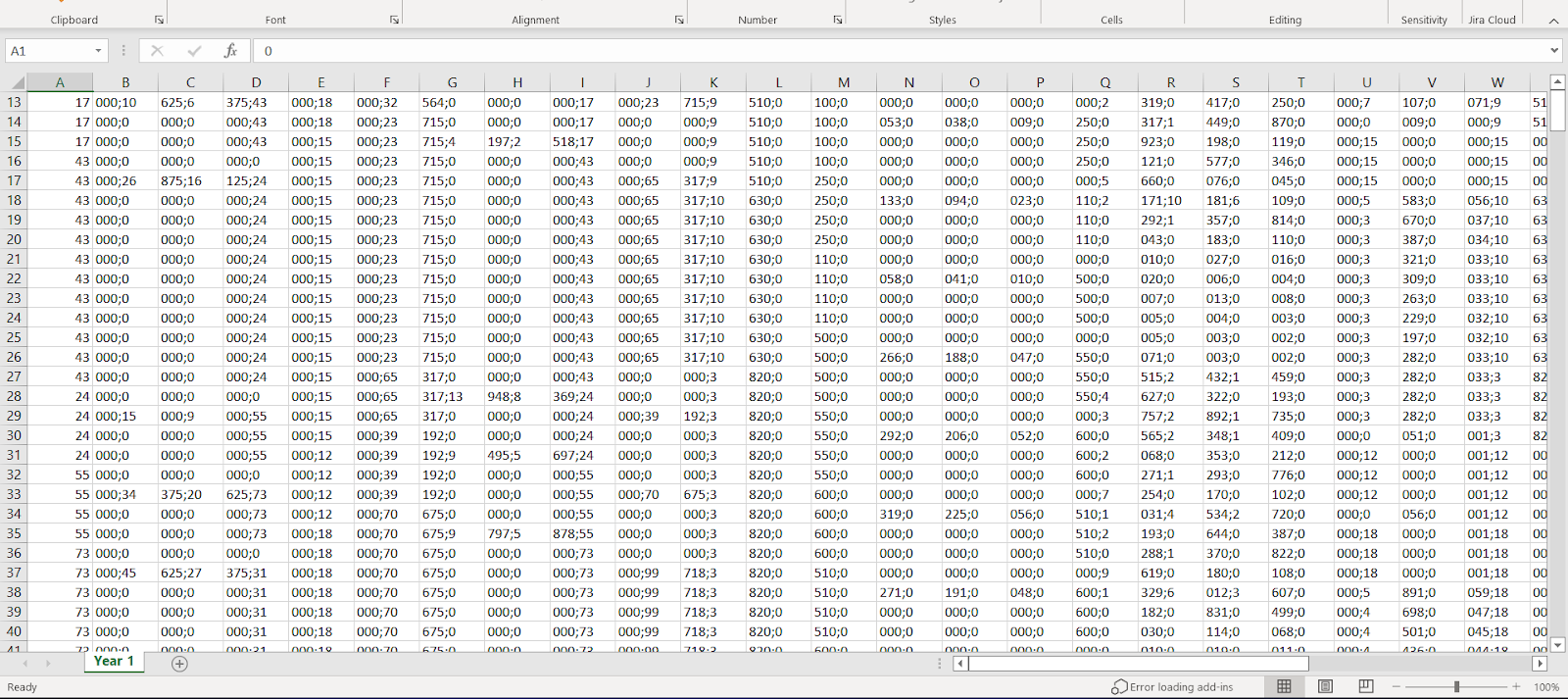
Приклади роботи програми:







Результатом моделювання є файл з розрахунками:



 Як видно із прикладів роботи програми, є певні недоліки у її роботі, такі як некоректне відображення більшості символів тексту, із за чого стає неможливим зрозуміти до чого відноситься та чи інша інформація, а також вихідний файл з розрахунками не має заголовків у колонках, що теж унеможливлює розуміння природу даних та до чого вони відносяться.

# 2 Зовнішнє проектування

## 2.1 Функціональне призначення

Програма моделює макроекономічну систему на основі базової ринкової моделі.

## 2.2 Експлуатаційне призначення

Дана програма допомагає користувачу краще зрозуміти механізм функціонування сучасної макроекономічної моделі.

## 2.3 Функціональні вимоги

Завданням даного ПЗ є демонстрація сучасної макроекономічної моделі.

Для спрощення розрахунків було прийнято агрегований спосіб моделювання. Кожен агрегований суб'єкт економічної діяльності (АСЕД) являє собою величезну кількість схожих об'єктів, але які виробляють абсолютно різну продукцію (різна ціна, різна тривалість технологічного циклу тощо).

Процес моделювання охоплює якийсь календарний проміжок часу, тому потрібен лічильник днів. Вважатимемо, що кожен день щось послідовно відбувається (взяття кредиту, покупка сировини, виготовлення продукції, продаж продукції, виплата заробітної плати, купівля сировини тощо). У даному моделюванні нас цікавить не кількість виробленого товару (що практично неможливо підрахувати), а умовну кількість грошей, яка обертається в макроекономічній системі та як воно впливає на макроекономічні показники.

У кожному АСЕД мають бути лічильники, які вважають прихід грошей і витрати грошей, а також показують скільки грошей знаходиться зараз і куди були спрямовані гроші.

Програма має щодня переходити від одного АСЭД до іншого та робити одну операцію згідно загальним вхідним даним.

## 2.4 Загальні вхідні дані

1. Задається відсоток грошей на сплату податку, %

2. Задається відсоток грошей, які спрямовуються на виплату заробітної плати, %

3. Задаються відсотки за кредитами та депозитами, %

4. Задається ставка Національного банку, %

5. Задається період повернення кредиту

6. Задається інтервал періоду моделювання на днях або вказується – 1 рік, 2 роки.

7. Задається період часу, через який виплачується

8. Зарплата та податок (або конкретна кількість днів, або можливий період). Може треба вказувати період, а якщо мова йде про конкретну кількість днів вказувати, наприклад, так 2-2 (два дні)

Після реалізації товару, наступного дня треба випадковим чином визначити кількість грошей, необхідні виготовлення наступного товару. Якщо після реалізації товару, грошей залишилося більше, то надлишок треба направити на виплату зарплати та податку. Якщо грошей недостатньо, треба необхідну суму дібрати рахунок кредиту.

## 2.5 Суб’єкти економічної діяльності

### 2.5.1. Виробники сировини (ВС)

Загальний лічильник фіксує 1-й день роботи.

Визначається початкова кількість грошей в умовних одиницях (задається оператором). Це може бути або конкретне число, або вказано можливий інтервал, з якого програма випадковим чином згідно нормальному закону розподілу сама має обрати якесь число (наприклад, 10-100). (мабуть другий варіант більш зручним). (наприклад, програма обрала 65) Це означає, що цю кількість грошей треба витратити на закупівлю необхідних товарів для випуску сировини.

Задається можливий період циклу технологічного процесу днями (наприклад, 1-20). Програма випадково вибирає тривалість технологічного циклу (наприклад, 15). Це означає що через 15 днів буде виготовлено товар.

Визначається відсоток торгової націнки в деяких межах (наприклад, 20-100). Наступного дня передбачається та сама процедура за тими ж вихідними даними.

Завдання вихідних даних по кожному АСЭД бачиться у вигляді форми, куди вводяться потрібні числа.

### 2.5.2. Виробники засобів виробництва (ВЗВ)

Визначається початкова кількість грошей в умовних одиницях (задається оператором). Це може бути або конкретне число, або вказано можливий інтервал, з якого програма випадковим чином згідно нормальному закону розподілу сама має обрати якесь число (наприклад, 20-50). (мабуть другий варіант більш зручним). (наприклад, програма вибрала 45) Це означає, що цю кількість грошей треба витратити на закупівлю необхідних товарів для випуску сировини

Задається можливий період циклу технологічного процесу днями (наприклад, 1-30). Програма випадково вибирає тривалість технологічного циклу (наприклад, 20). Це означає, що через 20 днів буде виготовлено товар.

Визначається відсоток торгової націнки в деяких межах(наприклад, 20-100).

### 2.5.3. Виробники споживчих товарів (ВСТ)

Визначається початкова кількість грошей в умовних одиницях (задається оператором). Це може бути або конкретне число, або вказано можливий інтервал, з якого програма випадковим чином згідно нормальному закону розподілу сама має обрати якесь число (наприклад, 20-50). (мабуть другий варіант більш зручним). (наприклад, програма обрала 30) Це означає, що цю кількість грошей треба витратити на закупівлю необхідних товарів для випуску сировини.

Задається можливий період реалізації товару днями (наприклад, 1-10). Програма випадково вибирає тривалість технологічного циклу (наприклад, 5). Це означає, що через 5 днів товар буде реалізовано.

Визначається відсоток торгової націнки в деяких межах (наприклад, 20-100)

### 2.5.4. Торгівля та Послуги (ТП)

Визначається початкова кількість грошей в умовних одиницях (задається оператором). Це може бути або конкретне число, або вказано можливий інтервал, з якого програма випадковим чином згідно нормальному закону розподілу сама має обрати якесь число (наприклад, 20-50). (мабуть другий варіант більш зручним). (наприклад, програма обрала 30) Це означає, що цю кількість грошей треба витратити на закупівлю необхідних товарів для випуску сировини.

Послуги надаються на ті суми, на які до них звертаються.

### 2.5.5. Покупці споживчих товарів (ПСТ)

Задаються відсотки витрати грошей на «Послуги» та «Торгівлю».

### 2.5.6. Бюджет (Держава) (Б)

Задаються відсотки витрати бюджетних грошей на «Послуги», «Торгівлю» та оплату бюджетників та соціально захищених верств населення.

Задається ймовірність настання Надзвичайної Ситуації (стихійне лихо, техногенна катастрофа та т.п.). Задається діапазон потреб у кредитах (емісії грошей).

### 2.5.7. Комерційні банки (КБ)

Задається кількість грошей в умовних одиницях.

### 2.5.8. Нацбанк (НБ)

Задається загальна кількість грошей в умовних одиницях.

# 3 ВНУТРІШНЄ ПРОЕКТУВАННЯ

## 3.1 Об’єктно-орієнтоване програмування

Об'єктно-орієнтоване програмування або ООП − це методологія програмування, заснована на представленні програми у вигляді сукупності об'єктів кожен з яких являється реалізацією визначеного типу, що використовує механізм пересилки повідомлень і класи організовані в ієрархію наслідування.

Центральний елемент ООП − абстракція. В об'єктно-орієнтованому програмуванні абстракція − це додання об'єкту характеристик, які відрізняють його від всіх інших об'єктів, чітко визначаючи його концептуальні межі. Основна ідея полягає в тому, щоб відокремити спосіб використання складових об'єктів даних від деталей їх реалізації у вигляді більш простих об'єктів, подібно до того, як функціональна абстракція розділяє спосіб використання функції і деталей її реалізації в термінах більш примітивних функцій, таким чином, дані обробляються функцією високого рівня за допомогою виклику функцій низького рівня.

Такий підхід є основою об'єктно-орієнтованого програмування. Це дозволяє працювати з об'єктами, а не вдаючись в особливості їх реалізації. У кожному конкретному випадку застосовується той чи інший підхід: інкапсуляції, поліморфізм або успадкування. Наприклад, при необхідності звернутися до прихованих даними об'єкта, слід скористатися інкапсуляцією, створивши, так звану, функцію доступу або властивість.

Інкапсуляція − це властивість мови програмування, що дозволяє користувачеві не замислюватися про складність реалізації використовуваного програмного компонента, 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.

Користувач може взаємодіяти з об'єктом тільки через цей інтерфейс. Користувач не може використовувати закриті дані і методи.

Інкапсуляція − це один з чотирьох найважливіших механізмів об'єктно-орієнтованого програмування (поряд з абстракцією, поліморфізмом і спадкуванням).

Приховування реалізації доцільно застосовувати в наступних випадках:

− гранична локалізація змін при необхідності таких змін;

− прогнозованість змін (які зміни в коді треба зробити для заданої зміни функціональності) і прогнозованість наслідків змін.

Спадкування − це один з чотирьох найважливіших механізмів об'єктно-орієнтованого програмування (поряд з інкапсуляцією, поліморфізмом і абстракцією), що дозволяє описати новий клас на основі вже існуючого (батьківського), при цьому властивості і функціональність батьківського класу запозичуються новим класом.

Іншими словами, клас-спадкоємець реалізує специфікацію вже існуючого класу. Це дозволяє звертатися к об'єктам класу-спадкоємця так само, як з об'єктами базового класу.

Клас, від якого відбулося спадкування, називається базовим або батьківським. Класи, які походять від базового, називаються нащадками, спадкоємцями або похідними класами.

Абстрактний клас − це клас, що містить хоча б один абстрактний метод, він описаний в програмі, має поля, методи і не може використовуватися для безпосереднього створення об'єкта. Тобто від абстрактного класу можна тільки наслідувати. Об'єкти створюються тільки на основі похідних класів, успадкованих від абстрактного.

При множинному спадкуванні у класі може бути більш одного предку. В цьому випадку клас успадковує методи всіх предків. Переваги такого підходу в більшій гнучкості.

Поліморфізм − це можливість об'єктів з однаковою специфікацією мати різну реалізацію.

Мова програмування підтримує поліморфізм, якщо класи з однаковою специфікацією можуть мати різну реалізацію − наприклад, реалізація класу може бути змінена в процесі успадкування.

Поліморфізм − це один з чотирьох найважливіших механізмів об'єктно-орієнтованого програмування (поряд з абстракцією, інкапсуляцією і спадкуванням).

Поліморфізм дозволяє писати більш абстрактні програми і підвищити коефіцієнт повторного використання коду. Загальні властивості об'єктів об'єднуються в систему, яку можуть називати по-різному − інтерфейс, клас. Спільність має зовнішнє і внутрішнє вираження:

− зовнішня спільність проявляється як однаковий набір методів з однаковими іменами та сигнатурами (ім'ям методів і типами аргументів і їх кількістю);

− внутрішня спільність − це однакова функціональність методів. Її можна описати інтуїтивно або висловити у вигляді строгих законів, правил, яким повинні підкорятися методи. Можливість приписувати різну функціональність одному методу (функції, операції) називається перевантаженням методу (перевантаженням функцій, перевантаженням операцій).

## 3.2 Опис відповідальності класів

Клас Program відповідає за запуск програми.

Клас Form1 відповідає за відображення інформації про моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

Клас Form2 відповідає за відображення інформації про загальні вхідні дані програми для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

Клас Form3 відповідає за відображення інформації про саму програму.

Клас Form4 відповідає за відображення подій, які відбуваються протягом процесу моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

Клас InputData відповідає за зберігання вхідних даних програми.

Клас Сharacteristics відповідає за зберігання характеристик макроекономічної системи держави на базі класичної ринкової моделі.

Клас Economy\_events відповідає за зберігання можливих подій під час моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

Клас Form5 відповідає за відображення інформації про вхідні дані виробників сировини програми для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

Клас Form6 відповідає за відображення інформації про вхідні дані виробників засобів виробництва програми для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

Клас Form7 відповідає за відображення інформації про вхідні дані виробників споживчих товарів програми для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

Клас Form8 відповідає за відображення інформації про вхідні дані торгівлі та послуг програми для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

Клас Form9 відповідає за відображення інформації про інші вхідні дані програми для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

## 3.3 Опис зв’язків класів

Описані вище класи мають наступні зв'язки.

Об'єкти сутності класу Program пов'язані з об'єктами класу Form1. Тип зв’язку – асоціація.

Об'єкти сутності класу Form1 пов'язані з об'єктами класів Form2, Form3, Form4, Form5, Form6, Form7, Form8, Form9, Сharacteristics. Тип зв’язку – асоціація.

Об'єкти сутності класу Form2, Form5, Form6, Form7, Form8, Form9 пов'язані з об'єктами класу InputData. Тип зв’язку – асоціація.

Об'єкти сутності класу Form4 пов'язані з об'єктами класу Economy\_events. Тип зв’язку – асоціація.

## 3.4 Діаграма класів програми

На основі вищеописаної інформації про об’єктно-орієнтовану модель програми була побудована діаграма класів. Діаграма класів представлена на рис. 3.1. Пояснення до цієї діаграми (специфікації класів) описані у наступному пункті розділу.

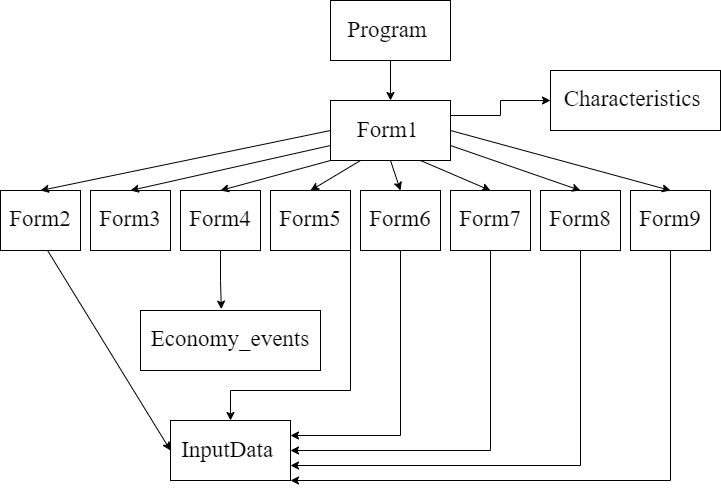


Рисунок 3.1 – Діаграма класів програми

## 3.5 Опис специфікацій класів

### 3.5.1 Опис специфікацій класу Program

Клас реалізовує наступні операції:

* запуск програми.

### 3.5.2 Опис специфікацій класу Form1

Клас реалізовує наступні операції:

* проведення розрахунків по моделюванню механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі та виведенні інформації на екран;
* моделюванню механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі у часі;
* перехід на форму введення загальних вхідних даних;
* перехід на форму введення вхідних даних виробників сировини;
* перехід на форму введення вхідних даних виробників засобів виробництва;
* перехід на форму введення вхідних даних виробників споживчих товарів;
* перехід на форму введення вхідних даних торгівлі та послуг;
* перехід на форму введення інших вхідних даних;
* перехід на форму перегляду вікна подій;
* перехід на форму перегляду детальної інформації про програму.

### 3.5.3 Опис специфікацій класу Form2

Клас реалізовує наступні операції:

* збереження інформації про загальні вхідні дані, введені користувачем програми;
* перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

### 3.5.4 Опис специфікацій класу Form3

Клас реалізовує наступні операції:

* відображення інформації про саму програму;
* перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

### 3.5.5 Опис специфікацій класу Form4

Клас реалізовує наступні операції:

* відображення інформації про події протягом моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі;
* оновлення інформації про події протягом моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі;
* перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

### 3.5.6 Опис специфікацій класу Form5

Клас реалізовує наступні операції:

* збереження інформації про вхідні дані виробників сировини, введені користувачем програми;
* перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

### 3.5.7 Опис специфікацій класу Form6

Клас реалізовує наступні операції:

* збереження інформації про вхідні дані виробників засобів виробництва, введені користувачем програми;
* перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

### 3.5.8 Опис специфікацій класу Form7

Клас реалізовує наступні операції:

* збереження інформації про вхідні дані виробників споживчих товарів, введені користувачем програми;
* перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

### 3.5.9 Опис специфікацій класу Form8

Клас реалізовує наступні операції:

* збереження інформації про вхідні дані торгівлі та послуг, введені користувачем програми;
* перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

### 3.5.10 Опис специфікацій класу Form9

Клас реалізовує наступні операції:

* збереження інформації про інші вхідні дані, введені користувачем програми;
* перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі.

### 3.5.11 Опис специфікацій класу InputData

Атрибути класу:

* змінна, що відповідає за швидкість покрокового моделювання;
* змінна, що відповідає за відсоток грошей на сплату податку;
* змінна, що відповідає за відсоток грошей на виплату заробітної плати;
* змінна, що відповідає за відсотки за кредитами та депозитами;
* змінна, що відповідає за ставку Національного банку;
* змінна, що відповідає за період часу повернення кредиту;
* змінна, що відповідає за інтервал періоду моделювання;
* змінна, що відповідає за первісну мінімальну кількість грошей виробників сировини;
* змінна, що відповідає за первісну максимальну кількість грошей виробників сировини;
* змінна, що відповідає за мінімальний період циклу технологічного процесу виробників сировини;
* змінна, що відповідає за максимальний період циклу технологічного процесу виробників сировини;
* змінна, що відповідає за мінімальний відсоток торгової націнки виробників сировини;
* змінна, що відповідає за максимальний відсоток торгової націнки виробників сировини;
* змінна, що відповідає за первісну мінімальну кількість грошей виробників засобів виробництва;
* змінна, що відповідає за первісну максимальну кількість грошей виробників засобів виробництва;
* змінна, що відповідає за мінімальний період циклу технологічного процесу виробників засобів виробництва;
* змінна, що відповідає за максимальний період циклу технологічного процесу виробників засобів виробництва;
* змінна, що відповідає за мінімальний відсоток торгової націнки виробників засобів виробництва;
* змінна, що відповідає за максимальний відсоток торгової націнки виробників засобів виробництва;
* змінна, що відповідає за первісну мінімальну кількість грошей виробників споживчих товарів;
* змінна, що відповідає за первісна максимальну кількість грошей виробників споживчих товарів;
* змінна, що відповідає за мінімальний період циклу технологічного процесу виробників споживчих товарів;
* змінна, що відповідає за максимальний період циклу технологічного процесу виробників споживчих товарів;
* змінна, що відповідає за мінімальний відсоток торгової націнки виробників споживчих товарів;
* змінна, що відповідає за максимальний відсоток торгової націнки виробників споживчих товарів;
* змінна, що відповідає за первісну мінімальну кількість грошей торгівлі та послуг;
* змінна, що відповідає за первісну максимальну кількість грошей торгівлі та послуг;
* змінна, що відповідає за відсоткове співвідношення купівлі товарів торгівлі та послуг;
* змінна, що відповідає за відсотки витрати грошей на «Послуги та торгівлю» ;
* змінна, що відповідає за відсотки витрачання бюджетних грошей на «Послуги та торгівлю» ;
* змінна, що відповідає за ймовірність настання Надзвичайної Ситуації;
* змінна, що відповідає за кількість грошей комерційного банку;
* змінна, що відповідає за загальну кількість грошей у Нацбанку.

### 3.5.12 Опис специфікацій класу Сharacteristics

Атрибути класу:

* змінна, що відповідає за прихід виробників сировини;
* змінна, що відповідає за витрати виробників сировини;
* змінна, що відповідає за сировина виробників сировини;
* змінна, що відповідає за остача виробників сировини;
* змінна, що відповідає за прихід виробників засобів виробництва;
* змінна, що відповідає за витрати виробників засобів виробництва;
* змінна, що відповідає за сировина виробників засобів виробництва;
* змінна, що відповідає за остача виробників засобів виробництва;
* змінна, що відповідає за прихід виробників споживчих товарів;
* змінна, що відповідає за витрати виробників споживчих товарів;
* змінна, що відповідає за сировина виробників споживчих товарів;
* змінна, що відповідає за остача виробників споживчих товарів;
* змінна, що відповідає за прихід торгівлі та послуг;
* змінна, що відповідає за витрати торгівлі та послуг;
* змінна, що відповідає за сировина торгівлі та послуг;
* змінна, що відповідає за остача торгівлі та послуг;
* змінна, що відповідає за прихід бюджету;
* змінна, що відповідає за витрати бюджету;
* змінна, що відповідає за остача бюджету;
* змінна, що відповідає за прихід комерційних банків;
* змінна, що відповідає за витрати комерційних банків;
* змінна, що відповідає за остача комерційних банків;
* змінна, що відповідає за прихід покупців споживчих товарів;
* змінна, що відповідає за витрати покупців споживчих товарів;
* змінна, що відповідає за остача покупців споживчих товарів;
* змінна, що відповідає за запас Нацбанку;
* змінна, що відповідає за витрати Нацбанку;
* змінна, що відповідає за емісія Нацбанку;
* змінна, що відповідає за остача Нацбанку.

### 3.5.13 Опис специфікацій класу Economy\_events

Атрибути класу:

* змінна, що відповідає за лічильник днів;
* змінна, що відповідає за підсумковий рядок подій;
* змінні, що відповідає за можливі підприємства;
* змінні, що відповідає за можливі дії.

## 3.6 Опис інтерфейсної частини класів

### 3.6.1 Клас Program

Члени класу описані у табл. 3.1.

Таблиця 3.1 – Члени класу Program

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| static void Main() | Функція, що запускає програму |  |  |

### 3.6.2 Клас Form1

Члени класу описані у табл. 3.2.

Таблиця 3.2 – Члени класу Form1

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| void Grafic\_Update() | Функція, що проводить розрахунки по моделюванню механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі та виводить інформацію на екран |  | Результати розрахунків на стовпчикових діаграмах |
| private void timer1\_Tick(object sender, EventArgs e) | Функція, що проводить розрахунки по моделюванню механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі із плином часу | Інформація про те, що таймер активний | Результати розрахунків на стовпчикових діаграмах |
| private void зупинитиПокроковеМоделюванняToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що зупиняє роботу таймера | Інформація про те, що кнопка була натиснута | Зупинка роботи таймера |
| private void розпочатиПокроковеМоделюванняToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що розпочинає роботу таймера | Інформація про те, що кнопка була натиснута | Початок роботи таймера |
| private void переглянутиToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує перехід на форму для перегляду вікна подій під час моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі | Інформація про те, що кнопка була натиснута | Перехід на форму для перегляду вікна подій |
| private void змоделювати1КрокToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що проводить розрахунки по моделюванню механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі та виводить інформацію на екран | Інформація про те, що кнопка була натиснута | Результати розрахунків на стовпчикових діаграмах |
| private void змінитиВхідніДаніToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує перехід на форму для введення загальних вхідних даних | Інформація про те, що кнопка була натиснута | Перехід на форму для введення загальних вхідних даних |
| private void вихідToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує завершення роботи програми | Інформація про те, що кнопка була натиснута | Завершення роботи програми |
| private void Form1\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |
| private void дузнатисяБільшеПроПрограмуToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує перехід на форму для перегляду детальнішої інформації про програму | Інформація про те, що кнопка була натиснута | Перехід на форму для перегляду детальнішої інформації про програму |
| private void виробникиСировиниToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує перехід на форму для введення вхідних даних виробників сировини | Інформація про те, що кнопка була натиснута | Перехід на форму для введення вхідних даних виробників сировини |
| private void виробникиЗасобівВиробництваToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує перехід на форму для введення вхідних даних виробників засобів виробництва | Інформація про те, що кнопка була натиснута | Перехід на форму для введення вхідних даних виробників засобів виробництва |
| private void виробникиСпоживчихТоварівToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує перехід на форму для введення вхідних даних виробників споживчих товарів | Інформація про те, що кнопка була натиснута | Перехід на форму для введення вхідних даних виробників споживчих товарів |
| private void торгівляТаПослугиToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує перехід на форму для введення вхідних даних торгівлі та послуг | Інформація про те, що кнопка була натиснута | Перехід на форму для введення вхідних даних торгівлі та послуг |
| private void іншіToolStripMenuItem\_Click(object sender, EventArgs e) | Функція, що реалізовує перехід на форму для введення інших вхідних даних | Інформація про те, що кнопка була натиснута | Перехід на форму для введення інших вхідних даних |

### 3.6.3 Клас Form2

Члени класу описані у табл. 3.3.

Таблиця 3.3 – Члени класу Form2

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| private void button1\_Click(object sender, EventArgs e) | Функція, що зберігає вхідні дані, введені користувачем та реалізовує перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі | Інформація про те, що кнопка була натиснута | Перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі |
| private void Form2\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |

### 3.6.4 Клас Form3

Члени класу описані у табл. 3.4.

Таблиця 3.4 – Члени класу Form3

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| private void Form3\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |

### 3.6.5 Клас Form4

Члени класу описані у табл. 3.5.

Таблиця 3.5 – Члени класу Form4

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| private void button1\_Click(object sender, EventArgs e) | Функція, що оновлює вміст поля, для виведення інформації про події під час моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі | Інформація про те, що кнопка була натиснута | Оновлення вмісту поля |
| private void Form2\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |

### 3.6.6 Клас Form5

Члени класу описані у табл. 3.6.

Таблиця 3.6 – Члени класу Form5

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| private void button1\_Click(object sender, EventArgs e) | Функція, що зберігає вхідні дані, введені користувачем та реалізовує перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі | Інформація про те, що кнопка була натиснута | Перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі |
| private void Form5\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |

### 3.6.7 Клас Form6

Члени класу описані у табл. 3.7.

Таблиця 3.7 – Члени класу Form6

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| private void button1\_Click(object sender, EventArgs e) | Функція, що зберігає вхідні дані, введені користувачем та реалізовує перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі | Інформація про те, що кнопка була натиснута | Перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі |
| private void Form6\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |

### 3.6.8 Клас Form7

Члени класу описані у табл. 3.8.

Таблиця 3.8 – Члени класу Form7

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| private void button1\_Click(object sender, EventArgs e) | Функція, що зберігає вхідні дані, введені користувачем та реалізовує перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі | Інформація про те, що кнопка була натиснута | Перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі |
| private void Form6\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |

### 3.6.9 Клас Form8

Члени класу описані у табл. 3.9.

Таблиця 3.9 – Члени класу Form8

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| private void button1\_Click(object sender, EventArgs e) | Функція, що зберігає вхідні дані, введені користувачем та реалізовує перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі | Інформація про те, що кнопка була натиснута | Перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі |
| private void Form6\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |

### 3.6.10 Клас Form9

Члени класу описані у табл. 3.10.

Таблиця 3.10 – Члени класу Form9

|  |  |  |  |
| --- | --- | --- | --- |
| Назва | Опис | Вхід | Вихід |
| private void button1\_Click(object sender, EventArgs e) | Функція, що зберігає вхідні дані, введені користувачем та реалізовує перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі | Інформація про те, що кнопка була натиснута | Перехід на форму для моделювання механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі |
| private void Form6\_Load(object sender, EventArgs e) | Функція, що реалізовує завантаження даних при переході на дану форму | Інформація про те, що користувач перейшов на дану форму | Завантаження даних при переході на дану форму |

## 3.7 Проектування інтерфейсу користувача

Інтерфейс − це сукупність інформаційної моделі предметної областей, засобів і способів взаємодії з інформаційною моделлю, а також компонентів які забезпечує формування інформаційної моделі в процесі роботи програмних систем.

Основою принципів проектування інтерфейсів користувача є людські можливості. Принципи проектування:

* природність (інтуїтивність). Користувача не слід змушувати істотно змінювати звичні способи вирішення завдань, коли робота з системою не викликає у користувача необхідного пошуку елементів інтерфейсу для поставленого завдання. Доцільно зберегти позначення та термінологію даної предметної області;
* узгодженість (несуперечність). Забезпечення наступності знань і навичок, коли в процесі роботи з системою користувачем використовувалися деякі прийоми роботи з деякою частиною системи, то в іншій частині прийоми повинні бути ідентичні. Погоджений інтерфейс є впізнаваним і передбачуваним, а також відповідним нормам;
* ненадмірність. Користувач повинен вводити тільки мінімальну інформацію для роботи або управління системою. Слід позбутися повторного вводу інформації;
* дружність (безпосередній доступ до системи допомоги). Коли в процесі роботи система забезпечує користувача повідомленнями про помилки та інформацією що система робить. Повідомлення про помилку має бути корисним і зрозумілим користувачеві. Ефективний інтерфейс повинен запобігати ситуації, коли користувач може помилитися, і запобігати наслідок можливих помилок;
* гнучкість (адаптивність). Коли інтерфейс системи може обслуговувати користувачів з різним рівнем підготовки. Передбачається можливість зміни структури діалогу або вхідних даних. Для недосвідчених користувачів інтерфейс може бути організований, як ієрархічна структура меню, а для досвідчених, як комбінації клавіш;
* простота інтерфейсу. Забезпечення легкості в його вивченні і використанні, крім того він повинен надавати доступ до всього переліку можливостей, передбачених додатком (що суперечить простоті);
* принцип зворотного зв'язку. Програма повинна мати відгук на будь-яку дію користувача.

До візуальних атрибутів відображення інформації відносяться:

* палітра кольорів;
* взаємне розташування і розміри об'єктів, що відображаються;
* засоби залучення уваги користувача.

Колір є одним з найважливіших візуальних атрибутів інтерфейсу. Оскільки колір має здатність залучати погляд, його слід використовувати для ідентифікації тих елементів інтерфейсу, на які користувачеві слід звернути увагу. Колір також має асоціативний аспект, у багатьох випадках об'єкти одного кольору сприймаються людьми як взаємопов'язані. Кольорова гамма може також надавати певний емоційний або психологічний вплив. Недарма, наприклад, поділяють кольори на «холодні» і «теплі».

У роботах, присвячених дослідженню психологічного впливу різних кольорів, наводяться такі дані:

* блакитний (заспокоює);
* червоний (хвилює і стомлює);
* зелений (налаштовує на добродушний і безініціативний лад);
* жовтий (веселий, оптимістичний, викликає легковажний настрій);
* помаранчевий (розкріпачує фантазію);
* фіолетовий (згубний для очей, колір заздрості, тривоги, незадоволеності);
* коричневий (пригнічує розумову активність);
* чорний (похмурий, сприяє виникненню головного болю, але знижує число помилок).

Таким чином, непродумане використання кольору може викликати у користувача негативний емоційний стан або навіть відволікати його від роботи, заважаючи зосередитися на виконуваному завданні.

## 3.8 Аналіз проекту

Аналізуючи розроблений проект, можна зробити висновок, що усі класи мають функціональну зв’язність.

## 3.9 Вибір мови програмування

Програма була написана на мові C#.

C# – це об'єктно-орієнтована, проста і водночас потужна мова програмування, яка дозволяє розробникам створювати багатофункціональні програми.

C# ставитися до мов компілюваного типу, тому він має всі переваги таких мов. Він поєднує найкращі ідеї сучасних мов програмування Java, C++, Visual Basic тощо. Через велику різноманітність синтаксичних конструкцій та можливості працювати з платформою .Net, C# дозволяє швидше, ніж будь-яку іншу мову, розробляти програмні рішення. Також він відрізняється надійністю. Великої надійності було досягнуто через роботу CLR машини, адже на відміну від інших компілятор CLR запускає розроблений додаток на віртуальному процесорі. Тому у разі виникнення будь-яких помилок це ніяк не вплине на роботу інших програм у системі, але це також означає, що для запуску програми потрібен додатковий час. Відповідно програми написані мовою програмування C# надійніші, але менш швидкі (ніж самі програми написані на C++).

# 4 ТЕСТУВАННЯ ТА НАЛАГОДЖЕННЯ ПРОГРАМИ

## 4.1 Стратегія тестування

Стратегія тестування — це план проведення робіт із тестування системи або її модуля, що враховує специфіку функціональності та залежності з іншими компонентами системи та платформи. Стратегія визначає типи тестів, які потрібно виконувати для даного функціоналу системи, включає опис необхідних підходів з точки зору цілей тестування та може задавати описи або вимоги до необхідних для проведення тестування інструментів та інфраструктури.

Стратегія відповідає на запитання:

1) Як, як тестування дасть відповідь, що цей функціонал працює?

2) Що потрібно зробити і чим користуватися з інструментальних засобів для досягнення цілей тестування?

3) Коли певний функціонал тестуватиметься і коли очікувати отримання результатів?

4) Як додаткове завдання, яке вирішується в процесі розуміння стратегії тестування, можна розглядати задачу мінімізації витрат на тестування.

Існує багато способів тестування програмного забезпечення. Для даної програми були обрані методи тестування «чорною скринькою» та «білою скринькою».

Тестування «чорною скринькою» – це функціональне та нефункціональне тестування без доступу до внутрішньої структури компонентів системи. Метод тестування «чорної скриньки» – процедура отримання та вибору тестових випадків на основі аналізу специфікації (функціональної або нефункціональної), компонентів або системи без посилання на їх внутрішній пристрій.

Переваги:

1) Тестування методом «чорної скриньки» дозволяє знайти помилки, які неможливо виявити методом «білої скриньки». Найпростіший приклад: розробник забув додати якусь функціональність. З погляду коду все працює ідеально, але з погляду специфікації це надкритичний баг.

2) «Чорна скринька» дозволяє швидко виявити помилки у функціональних специфікаціях (у них описані не лише вхідні значення, а й те, що ми маємо отримати). Якщо отриманий під час тестування результат відрізняється від заявленого специфікації, ми маємо привід спілкування з аналітиком уточнення кінцевого результату.

3) Тестувальнику не потрібна додаткова кваліфікація. Часто ми користуємося різними сервісами та програмами, не дуже в них розбираючись. Для того, щоб відкрити інстаграм і обробити свою фотографію, нам не потрібно знати спосіб реалізації фільтрів. Ми хочемо відкрити фотографію, вибрати фільтр та отримати гарну картинку на виході. Завдання тестувальника, який тестує цю функцію в інстаграм, – переконатися, що користувач отримає цю найкрасивішу картинку відповідно до вибраного фільтра. При цьому нам зовсім не обов'язково мати якусь спеціалізацію – потрібні лише телефон та інстаграм.

4) Тестування відбувається «з позиції користувача». Користувач завжди має рацію, він кінцевий споживач практично будь-якого ПЗ, а значить, йому має бути зручно, комфортно і зрозуміло.

5) Складати тест-кейси можна одразу після підготовки специфікації. Це значно скорочує час на тестування: до того моменту, як продукт готовий до тестування, тест-кейси вже розроблені і тестувальник може відразу приступати до перевірки.

Недоліки:

1) Основним недоліком методу «чорної скриньки» є можливість пропуску кордонів та переходів, які не очевидні зі специфікації, але є в реалізації коду (власне, це і змушує тестувальників використовувати метод «білої скриньки»). Згадується випадок, коли система отримувала котирування валют із біржі Forex та округляла до 3 знаків після коми. Система успішно пройшла тестування методом «чорної скриньки» (оскільки жодна валюта не виходила за відповідні кордони) і добре працювала доти, поки курс долара до біткоїну не вийшов за межі 1000 доларів. Тестування «білою скринькою» виявило б помилку: фахівець побачив би, що коефіцієнт конверсії валюти обмежений 3 знаками.

2) Можна протестувати лише невелику кількість можливих вступних (вхідних) значень; багато варіантів залишаються без перевірки.

3) Тести можуть бути надмірними, якщо розробник вже перевірив цю функціональність (наприклад, Unit-тестом).

4) За відсутності чіткої та повної специфікації проектувати тести та тест-сценарії виявляється важко.

Тестування «білою скринькою» - це тестування внутрішньої структури, дизайну та кодування програмного рішення. У цьому типі тестування код видно тестеру. Основна увага приділяється перевірці потоку вхідних та вихідних даних через додаток, покращення дизайну та зручності використання, посилення безпеки. Тестування білої скриньки також відоме як тестування Clear Box, тестування Open Box, структурне тестування, тестування прозорого боксу, тестування на основі коду та тестування Glass Box. Це зазвичай виконується розробниками.

Переваги:

1) Оптимізація коду шляхом пошуку прихованих помилок.

2) Тестові випадки «білої скриньки» можуть легко автоматизуватися.

3) Тестування є ретельнішим, оскільки зазвичай покриваються всі шляхи коду.

4) Тестування може розпочатися рано в SDLC, навіть якщо GUI недоступний.

Недоліки:

1) Тестування білої скриньки може бути досить складним та дорогим.

2) Розробники, які зазвичай виконують тести «білої скриньки», ненавидять це. Тестування білої скриньки розробниками не може детально призвести до виробничих помилок.

3) Тестування білої скриньки потребує професійних ресурсів із докладним розумінням програмування та реалізації.

4) Тестування білої скриньки забирає багато часу, більші програми для програмування вимагають часу для повного тестування.

## 4.2 Специфікація та текст модулів

### 4.2.1 Опис специфікацій та текст функції Grafic\_Update

Атрибути функції:

- змінна, що відповідає за відсоток грошей на сплату податку;

- змінна, що відповідає за відсоток грошей на виплату заробітної плати;

- змінна, що відповідає за відсотки за кредитами та депозитами;

- змінна, що відповідає за ставку Національного банку;

- змінна, що відповідає за період часу повернення кредиту;

- змінна, що відповідає за інтервал періоду моделювання;

- змінна, що відповідає за первісну мінімальну кількість грошей виробників сировини;

- змінна, що відповідає за первісну максимальну кількість грошей виробників сировини;

- змінна, що відповідає за мінімальний період циклу технологічного процесу виробників сировини;

- змінна, що відповідає за максимальний період циклу технологічного процесу виробників сировини;

- змінна, що відповідає за мінімальний відсоток торгової націнки виробників сировини;

- змінна, що відповідає за максимальний відсоток торгової націнки виробників сировини;

- змінна, що відповідає за первісну мінімальну кількість грошей виробників засобів виробництва;

- змінна, що відповідає за первісну максимальну кількість грошей виробників засобів виробництва;

- змінна, що відповідає за мінімальний період циклу технологічного процесу виробників засобів виробництва;

- змінна, що відповідає за максимальний період циклу технологічного процесу виробників засобів виробництва;

- змінна, що відповідає за мінімальний відсоток торгової націнки виробників засобів виробництва;

- змінна, що відповідає за максимальний відсоток торгової націнки виробників засобів виробництва;

- змінна, що відповідає за первісну мінімальну кількість грошей виробників споживчих товарів;

- змінна, що відповідає за первісна максимальну кількість грошей виробників споживчих товарів;

- змінна, що відповідає за мінімальний період циклу технологічного процесу виробників споживчих товарів;

- змінна, що відповідає за максимальний період циклу технологічного процесу виробників споживчих товарів;

- змінна, що відповідає за мінімальний відсоток торгової націнки виробників споживчих товарів;

- змінна, що відповідає за максимальний відсоток торгової націнки виробників споживчих товарів;

- змінна, що відповідає за первісну мінімальну кількість грошей торгівлі та послуг;

- змінна, що відповідає за первісну максимальну кількість грошей торгівлі та послуг;

- змінна, що відповідає за відсоткове співвідношення купівлі товарів торгівлі та послуг;

- змінна, що відповідає за відсотки витрати грошей на «Послуги та торгівлю»;

- змінна, що відповідає за відсотки витрачання бюджетних грошей на «Послуги та торгівлю»;

- змінна, що відповідає за ймовірність настання Надзвичайної Ситуації;

- змінна, що відповідає за кількість грошей комерційного банку;

- змінна, що відповідає за загальну кількість грошей у Нацбанку.

- змінна, що відповідає за прихід виробників сировини;

- змінна, що відповідає за витрати виробників сировини;

- змінна, що відповідає за сировина виробників сировини;

- змінна, що відповідає за остача виробників сировини;

- змінна, що відповідає за прихід виробників засобів виробництва;

- змінна, що відповідає за витрати виробників засобів виробництва;

- змінна, що відповідає за сировина виробників засобів виробництва;

- змінна, що відповідає за остача виробників засобів виробництва;

- змінна, що відповідає за прихід виробників споживчих товарів;

- змінна, що відповідає за витрати виробників споживчих товарів;

- змінна, що відповідає за сировина виробників споживчих товарів;

- змінна, що відповідає за остача виробників споживчих товарів;

- змінна, що відповідає за прихід торгівлі та послуг;

- змінна, що відповідає за витрати торгівлі та послуг;

- змінна, що відповідає за сировина торгівлі та послуг;

- змінна, що відповідає за остача торгівлі та послуг;

- змінна, що відповідає за прихід бюджету;

- змінна, що відповідає за витрати бюджету;

- змінна, що відповідає за остача бюджету;

- змінна, що відповідає за прихід комерційних банків;

- змінна, що відповідає за витрати комерційних банків;

- змінна, що відповідає за остача комерційних банків;

- змінна, що відповідає за прихід покупців споживчих товарів;

- змінна, що відповідає за витрати покупців споживчих товарів;

- змінна, що відповідає за остача покупців споживчих товарів;

- змінна, що відповідає за запас Нацбанку;

- змінна, що відповідає за витрати Нацбанку;

- змінна, що відповідає за емісія Нацбанку;

- змінна, що відповідає за остача Нацбанку;

- змінна, що відповідає за лічильник днів;

- змінна, що відповідає за підсумковий рядок подій;

- змінні, що відповідає за можливі підприємства;

- змінні, що відповідає за можливі дії.

Функція проводить розрахунки по моделюванню механізму функціонування макроекономічної системи держави на базі класичної ринкової моделі та виводить інформацію на екран.

Текст функції:

void Grafic\_Update()

{

Economy\_events.day++;

Random rand = new Random();

Economy\_events.result += "---------------- День " + Economy\_events.day + " ----------------\n";

if (Economy\_events.day == 1)//1

{

InputData.vs\_initial\_amount\_of\_money = rand.Next(InputData.vs\_initial\_amount\_of\_money\_min, InputData.vs\_initial\_amount\_of\_money\_max + 1);

InputData.vzv\_initial\_amount\_of\_money = rand.Next(InputData.vzv\_initial\_amount\_of\_money\_min, InputData.vzv\_initial\_amount\_of\_money\_max + 1);

InputData.vst\_initial\_amount\_of\_money = rand.Next(InputData.vst\_initial\_amount\_of\_money\_min, InputData.vst\_initial\_amount\_of\_money\_max + 1);

InputData.ttp\_initial\_amount\_of\_money = rand.Next(InputData.ttp\_initial\_amount\_of\_money\_min, InputData.ttp\_initial\_amount\_of\_money\_max + 1);

InputData.vs\_process\_cycle\_time = rand.Next(InputData.vs\_process\_cycle\_time\_min, InputData.vs\_process\_cycle\_time\_max + 1);

InputData.vs\_markup\_percentage = rand.Next(InputData.vs\_markup\_percentage\_min, InputData.vs\_markup\_percentage\_max + 1);

InputData.vzv\_process\_cycle\_time = rand.Next(InputData.vzv\_process\_cycle\_time\_min, InputData.vzv\_process\_cycle\_time\_max + 1);

InputData.vzv\_markup\_percentage = rand.Next(InputData.vzv\_markup\_percentage\_min, InputData.vzv\_markup\_percentage\_max + 1);

InputData.vst\_process\_cycle\_time = rand.Next(InputData.vst\_process\_cycle\_time\_min, InputData.vst\_process\_cycle\_time\_max + 1);

InputData.vst\_markup\_percentage = rand.Next(InputData.vst\_markup\_percentage\_min, InputData.vst\_markup\_percentage\_max + 1);

Сharacteristics.material\_remainder = InputData.vs\_initial\_amount\_of\_money;

Сharacteristics.manufacturer\_remainder = InputData.vzv\_initial\_amount\_of\_money;

Сharacteristics.consumer\_remainder = InputData.vst\_initial\_amount\_of\_money;

Сharacteristics.trade\_remainder = InputData.ttp\_initial\_amount\_of\_money;

Сharacteristics.commercial\_remainder = InputData.commercial\_bank\_amount;

Сharacteristics.bank\_stock = InputData.national\_Bank\_total\_amount;

Сharacteristics.bank\_remainder = InputData.national\_Bank\_total\_amount;

}

Сharacteristics.material\_coming = 0;

Сharacteristics.manufacturer\_coming = 0;

Сharacteristics.consumer\_coming = 0;

Сharacteristics.trade\_coming = 0;

Сharacteristics.budget\_coming = 0;

Сharacteristics.commercial\_coming = 0;

Сharacteristics.buyers\_coming = 0;

Сharacteristics.material\_consumption = 0;

Сharacteristics.manufacturer\_consumption = 0;

Сharacteristics.consumer\_consumption = 0;

Сharacteristics.trade\_consumption = 0;

Сharacteristics.budget\_consumption = 0;

Сharacteristics.commercial\_consumption = 0;

Сharacteristics.buyers\_consumption = 0;

Сharacteristics.bank\_consumption = 0;

if (Сharacteristics.material\_raw > 0)//2

{

Сharacteristics.material\_coming = Сharacteristics.material\_raw \* InputData.vs\_markup\_percentage / 100;//заработал деньги за товар

Сharacteristics.manufacturer\_remainder -= Сharacteristics.material\_raw \* InputData.vs\_markup\_percentage / 300;//отнимаем деньги у купивших товар

Сharacteristics.consumer\_remainder -= Сharacteristics.material\_raw \* InputData.vs\_markup\_percentage / 300;

Сharacteristics.trade\_remainder -= Сharacteristics.material\_raw \* InputData.vs\_markup\_percentage / 300;

Сharacteristics.manufacturer\_consumption += Сharacteristics.material\_raw \* InputData.vs\_markup\_percentage / 300;//делаем расход купившим товар

Сharacteristics.consumer\_consumption += Сharacteristics.material\_raw \* InputData.vs\_markup\_percentage / 300;

Сharacteristics.trade\_consumption += Сharacteristics.material\_raw \* InputData.vs\_markup\_percentage / 300;

Сharacteristics.material\_raw = 0;//продал товар, его не стало

Economy\_events.result += Economy\_events.material + Economy\_events.ev10;

Economy\_events.result += Economy\_events.manufacturer + Economy\_events.ev8;

Economy\_events.result += Economy\_events.consumer + Economy\_events.ev8;

Economy\_events.result += Economy\_events.trade + Economy\_events.ev8;

}

else

{

if (material\_counter == InputData.vs\_process\_cycle\_time)

{

Сharacteristics.material\_consumption += Сharacteristics.material\_remainder / 10;//потратил деньги на производство сырья

Сharacteristics.material\_remainder -= Сharacteristics.material\_remainder / 10;

Сharacteristics.material\_raw += (Сharacteristics.material\_remainder + Сharacteristics.material\_remainder / 10);//произвели товар(думаю)

material\_counter = 0;

Economy\_events.result += Economy\_events.material + Economy\_events.ev6;

}

else

material\_counter++;

}

if (Сharacteristics.manufacturer\_raw > 0)//2

{

Сharacteristics.manufacturer\_coming += Сharacteristics.manufacturer\_raw \* InputData.vzv\_markup\_percentage / 100;//заработал деньги за товар

Сharacteristics.material\_remainder -= Сharacteristics.manufacturer\_raw \* InputData.vzv\_markup\_percentage / 300;

Сharacteristics.consumer\_remainder -= Сharacteristics.manufacturer\_raw \* InputData.vzv\_markup\_percentage / 300;

Сharacteristics.trade\_remainder -= Сharacteristics.manufacturer\_raw \* InputData.vzv\_markup\_percentage / 300;

Сharacteristics.material\_consumption += Сharacteristics.manufacturer\_raw \* InputData.vzv\_markup\_percentage / 300;

Сharacteristics.consumer\_consumption += Сharacteristics.manufacturer\_raw \* InputData.vzv\_markup\_percentage / 300;

Сharacteristics.trade\_consumption += Сharacteristics.manufacturer\_raw \* InputData.vzv\_markup\_percentage / 300;

Сharacteristics.manufacturer\_raw = 0;//продал товар, его не стало

Economy\_events.result += Economy\_events.manufacturer + Economy\_events.ev10;

Economy\_events.result += Economy\_events.material + Economy\_events.ev8;

Economy\_events.result += Economy\_events.consumer + Economy\_events.ev8;

Economy\_events.result += Economy\_events.trade + Economy\_events.ev8;

}

else

{

if (manufacturer\_counter == InputData.vzv\_process\_cycle\_time)

{

Сharacteristics.manufacturer\_consumption += Сharacteristics.manufacturer\_remainder / 10;//потратил деньги на производство сырья

Сharacteristics.manufacturer\_remainder -= Сharacteristics.manufacturer\_remainder / 10;

Сharacteristics.manufacturer\_raw += (Сharacteristics.manufacturer\_remainder + Сharacteristics.manufacturer\_remainder / 10);//произвели товар(думаю)

Сharacteristics.manufacturer\_coming = 0;

Economy\_events.result += Economy\_events.manufacturer + Economy\_events.ev6;

}

else

manufacturer\_counter++;

}

if (Сharacteristics.consumer\_raw > 0)//2

{

Сharacteristics.consumer\_coming += Сharacteristics.consumer\_raw \* InputData.vst\_markup\_percentage / 100;//заработал деньги за товар

Сharacteristics.trade\_remainder -= Сharacteristics.consumer\_raw \* InputData.vst\_markup\_percentage / 100;

Сharacteristics.trade\_consumption += Сharacteristics.consumer\_raw \* InputData.vst\_markup\_percentage / 100;

Сharacteristics.consumer\_raw = 0;//продал товар, его не стало

Economy\_events.result += Economy\_events.consumer + Economy\_events.ev10;

Economy\_events.result += Economy\_events.trade + Economy\_events.ev8;

}

else

{

if (consumer\_counter == InputData.vst\_process\_cycle\_time)

{

Сharacteristics.consumer\_consumption += Сharacteristics.consumer\_remainder / 10;//потратил деньги на производство сырья

Сharacteristics.consumer\_remainder -= Сharacteristics.consumer\_remainder / 10;

Сharacteristics.consumer\_raw += (Сharacteristics.consumer\_remainder + Сharacteristics.consumer\_remainder / 10);//произвели товар(думаю)

consumer\_counter = 0;

Economy\_events.result += Economy\_events.consumer + Economy\_events.ev6;

}

else

consumer\_counter++;

}

if (Сharacteristics.trade\_raw > 0)//2

{

Сharacteristics.trade\_coming += Сharacteristics.trade\_raw / 10;//заработал деньги за товар

Сharacteristics.material\_consumption += Сharacteristics.trade\_raw / 70;

Сharacteristics.consumer\_consumption += Сharacteristics.trade\_raw / 70;

Сharacteristics.manufacturer\_consumption += Сharacteristics.trade\_raw / 70;

Сharacteristics.budget\_consumption += Сharacteristics.trade\_raw / 70;

Сharacteristics.commercial\_consumption += Сharacteristics.trade\_raw / 70;

Сharacteristics.buyers\_consumption += Сharacteristics.buyers\_remainder \* InputData.money\_spending\_percentage / 100;

Сharacteristics.bank\_consumption += Сharacteristics.trade\_raw / 70;

Сharacteristics.material\_remainder -= Сharacteristics.trade\_raw / 70;

Сharacteristics.consumer\_remainder -= Сharacteristics.trade\_raw / 70;

Сharacteristics.manufacturer\_remainder -= Сharacteristics.trade\_raw / 70;

Сharacteristics.budget\_remainder -= Сharacteristics.trade\_raw / 70;

Сharacteristics.commercial\_remainder -= Сharacteristics.trade\_raw / 70;

Сharacteristics.buyers\_remainder -= Сharacteristics.buyers\_remainder \* InputData.money\_spending\_percentage / 100;

Сharacteristics.bank\_remainder -= Сharacteristics.trade\_raw / 70;

Сharacteristics.trade\_raw = 0;//продал товар, его не стало

Economy\_events.result += Economy\_events.trade + Economy\_events.ev10;

Economy\_events.result += Economy\_events.material + Economy\_events.ev8;

Economy\_events.result += Economy\_events.consumer + Economy\_events.ev8;

Economy\_events.result += Economy\_events.manufacturer + Economy\_events.ev8;

Economy\_events.result += Economy\_events.budget + Economy\_events.ev8;

Economy\_events.result += Economy\_events.commercial + Economy\_events.ev8;

Economy\_events.result += Economy\_events.buyers + Economy\_events.ev8;

Economy\_events.result += Economy\_events.bank + Economy\_events.ev8;

}

else

{

if (trade\_counter == 1)

{

Сharacteristics.trade\_consumption += Сharacteristics.trade\_remainder / 10;//потратил деньги на производство сырья

Сharacteristics.trade\_remainder -= Сharacteristics.trade\_remainder / 10;

Сharacteristics.trade\_raw += (Сharacteristics.trade\_remainder + Сharacteristics.trade\_remainder / 10);//произвели товар(думаю)

trade\_counter = 0;

Economy\_events.result += Economy\_events.trade + Economy\_events.ev6;

}

else

trade\_counter++;

}

if (Сharacteristics.material\_coming > 0)//3

{

Сharacteristics.material\_remainder += Сharacteristics.material\_coming -

(Сharacteristics.material\_coming \* InputData.tax\_money\_percentage / 100

+ Сharacteristics.material\_coming \* InputData.pay\_wages\_percentage / 100);//заплатил налоги и зп

Сharacteristics.budget\_coming += Сharacteristics.material\_coming - Сharacteristics.material\_coming \* InputData.tax\_money\_percentage / 100;

Сharacteristics.budget\_remainder += Сharacteristics.material\_coming - Сharacteristics.material\_coming \* InputData.tax\_money\_percentage / 100;

Economy\_events.result += Economy\_events.material + Economy\_events.ev1;

Economy\_events.result += Economy\_events.material + Economy\_events.ev2;

}

if (Сharacteristics.manufacturer\_coming > 0)//3 5

{

Сharacteristics.manufacturer\_remainder += Сharacteristics.manufacturer\_coming -

(Сharacteristics.manufacturer\_coming \* InputData.tax\_money\_percentage / 100

+ Сharacteristics.manufacturer\_coming \* InputData.pay\_wages\_percentage / 100);//заплатил налоги и зп

Сharacteristics.budget\_coming += Сharacteristics.manufacturer\_coming - Сharacteristics.manufacturer\_coming \* InputData.tax\_money\_percentage / 100;

Сharacteristics.budget\_remainder += Сharacteristics.manufacturer\_coming - Сharacteristics.manufacturer\_coming \* InputData.tax\_money\_percentage / 100;

Economy\_events.result += Economy\_events.manufacturer + Economy\_events.ev1;

Economy\_events.result += Economy\_events.manufacturer + Economy\_events.ev2;

}

if (Сharacteristics.consumer\_coming > 0)//3

{

Сharacteristics.consumer\_remainder += Сharacteristics.consumer\_coming -

(Сharacteristics.consumer\_coming \* InputData.tax\_money\_percentage / 100

+ Сharacteristics.consumer\_coming \* InputData.pay\_wages\_percentage / 100);//заплатил налоги и зп

Сharacteristics.budget\_coming += Сharacteristics.consumer\_coming - Сharacteristics.consumer\_coming \* InputData.tax\_money\_percentage / 100;

Сharacteristics.budget\_remainder += Сharacteristics.consumer\_coming - Сharacteristics.consumer\_coming \* InputData.tax\_money\_percentage / 100;

Economy\_events.result += Economy\_events.consumer + Economy\_events.ev1;

Economy\_events.result += Economy\_events.consumer + Economy\_events.ev2;

}

if (Сharacteristics.trade\_coming > 0)//3

{

Сharacteristics.trade\_remainder += Сharacteristics.trade\_coming -

(Сharacteristics.trade\_coming \* InputData.tax\_money\_percentage / 100

+ Сharacteristics.trade\_coming \* InputData.pay\_wages\_percentage / 100);//заплатил налоги и зп

Сharacteristics.budget\_coming += Сharacteristics.trade\_coming - Сharacteristics.trade\_coming \* InputData.tax\_money\_percentage / 100;

Сharacteristics.budget\_remainder += Сharacteristics.trade\_coming - Сharacteristics.trade\_coming \* InputData.tax\_money\_percentage / 100;

Economy\_events.result += Economy\_events.trade + Economy\_events.ev1;

Economy\_events.result += Economy\_events.trade + Economy\_events.ev2;

}

if (Сharacteristics.material\_remainder < (InputData.vs\_initial\_amount\_of\_money / 10))//кредит у комбанка

{

Сharacteristics.material\_remainder += InputData.commercial\_bank\_amount \* InputData.interest\_on\_loans / 70;

Сharacteristics.commercial\_remainder -= Сharacteristics.commercial\_remainder / 10;

Сharacteristics.commercial\_consumption += Сharacteristics.commercial\_remainder / 10;

Сharacteristics.material\_coming += InputData.commercial\_bank\_amount \* InputData.interest\_on\_loans / 70;

Economy\_events.result += Economy\_events.material + Economy\_events.ev3;

}

if (Сharacteristics.manufacturer\_remainder < (InputData.vzv\_initial\_amount\_of\_money / 10))//кредит у комбанка

{

Сharacteristics.manufacturer\_remainder += InputData.commercial\_bank\_amount \* InputData.interest\_on\_loans / 70;

Сharacteristics.commercial\_remainder -= Сharacteristics.commercial\_remainder / 10;

Сharacteristics.commercial\_consumption += Сharacteristics.commercial\_remainder / 10;

Сharacteristics.manufacturer\_coming += InputData.commercial\_bank\_amount \* InputData.interest\_on\_loans / 70;

Economy\_events.result += Economy\_events.manufacturer + Economy\_events.ev3;

}

if (Сharacteristics.consumer\_remainder < (InputData.vst\_initial\_amount\_of\_money / 10))//кредит у комбанка

{

Сharacteristics.consumer\_remainder += InputData.commercial\_bank\_amount \* InputData.interest\_on\_loans / 70;

Сharacteristics.commercial\_remainder -= Сharacteristics.commercial\_remainder / 10;

Сharacteristics.commercial\_consumption += Сharacteristics.commercial\_remainder / 10;

Сharacteristics.consumer\_coming += InputData.commercial\_bank\_amount \* InputData.interest\_on\_loans / 70;

Economy\_events.result += Economy\_events.consumer + Economy\_events.ev3;

}

if (Сharacteristics.trade\_remainder < (InputData.ttp\_initial\_amount\_of\_money / 10))//кредит у комбанка

{

Сharacteristics.trade\_remainder += InputData.commercial\_bank\_amount \* InputData.interest\_on\_loans / 70;

Сharacteristics.commercial\_remainder -= Сharacteristics.commercial\_remainder / 10;

Сharacteristics.commercial\_consumption += Сharacteristics.commercial\_remainder / 10;

Сharacteristics.trade\_coming += InputData.commercial\_bank\_amount \* InputData.interest\_on\_loans / 70;

Economy\_events.result += Economy\_events.trade + Economy\_events.ev3;

}

if (Сharacteristics.commercial\_remainder < (InputData.commercial\_bank\_amount / 10))//кредит у нацбанка

{

Сharacteristics.commercial\_remainder += InputData.national\_Bank\_total\_amount \* InputData.national\_Bank\_rate / 70;

Сharacteristics.bank\_remainder -= Сharacteristics.bank\_remainder / 7;

Сharacteristics.bank\_consumption += Сharacteristics.bank\_remainder / 7;

Сharacteristics.commercial\_coming += InputData.national\_Bank\_total\_amount \* InputData.national\_Bank\_rate / 70;

Economy\_events.result += Economy\_events.commercial + Economy\_events.ev3;

}

if (Сharacteristics.budget\_remainder > 300)

{

Сharacteristics.buyers\_coming += Сharacteristics.budget\_remainder / 2;

Сharacteristics.buyers\_remainder += Сharacteristics.budget\_remainder / 2;

Сharacteristics.budget\_consumption += Сharacteristics.budget\_remainder;

Сharacteristics.budget\_remainder = 0;

Economy\_events.result += Economy\_events.budget + Economy\_events.ev7;

Economy\_events.result += Economy\_events.budget + Economy\_events.ev9;

}

if (rand.Next(1, 100) < InputData.likelihood\_of\_an\_emergency)

{

Сharacteristics.budget\_consumption += Сharacteristics.budget\_remainder / 2;

Сharacteristics.budget\_remainder -= Сharacteristics.budget\_remainder / 2;

Economy\_events.result += Economy\_events.budget + Economy\_events.ev5;

}

if (Сharacteristics.bank\_remainder < (InputData.national\_Bank\_total\_amount / 10))//закончились деньги в нацбанке

{

Сharacteristics.bank\_stock += 100;

Сharacteristics.bank\_emission += 100;

Сharacteristics.bank\_remainder += 100;

Economy\_events.result += Economy\_events.bank + Economy\_events.ev4;

}

chart1.Series["Data"].Points.DataBindXY(new string[] { "Прихід", "Витрати", "Сировина", "Остача" }, new int[] { Сharacteristics.material\_coming, Сharacteristics.material\_consumption, Сharacteristics.material\_raw, Сharacteristics.material\_remainder });

//chart1.Series["Data"].Points.DataBindXY(new string[] { "Прихід" }, new int[] { Сharacteristics.material\_coming });

//chart1.Series["Data1"].Points.DataBindXY(new string[] { "Витрати" }, new int[] { Сharacteristics.material\_consumption });

//chart1.Series["Data2"].Points.DataBindXY(new string[] { "Сировина" }, new int[] { Сharacteristics.material\_raw });

//chart1.Series["Data3"].Points.DataBindXY(new string[] { "Остача" }, new int[] { Сharacteristics.material\_remainder });

chart2.Series["Data"].Points.DataBindXY(new string[] { "Прихід", "Витрати", "Сировина", "Остача" }, new int[] { Сharacteristics.manufacturer\_coming, Сharacteristics.manufacturer\_consumption, Сharacteristics.manufacturer\_raw, Сharacteristics.manufacturer\_remainder });

chart3.Series["Data"].Points.DataBindXY(new string[] { "Прихід", "Витрати", "Сировина", "Остача" }, new int[] { Сharacteristics.consumer\_coming, Сharacteristics.consumer\_consumption, Сharacteristics.consumer\_raw, Сharacteristics.consumer\_remainder });

chart4.Series["Data"].Points.DataBindXY(new string[] { "Прихід", "Витрати", "Сировина", "Остача" }, new int[] { Сharacteristics.trade\_coming, Сharacteristics.trade\_consumption, Сharacteristics.trade\_raw, Сharacteristics.trade\_remainder });

chart6.Series["Data"].Points.DataBindXY(new string[] { "Прихід", "Витрати", "Остача" }, new int[] { Сharacteristics.budget\_coming, Сharacteristics.budget\_consumption, Сharacteristics.budget\_remainder });

chart7.Series["Data"].Points.DataBindXY(new string[] { "Прихід", "Витрати", "Остача" }, new int[] { Сharacteristics.commercial\_coming, Сharacteristics.commercial\_consumption, Сharacteristics.commercial\_remainder });

chart5.Series["Data"].Points.DataBindXY(new string[] { "Прихід", "Витрати", "Остача" }, new int[] { Сharacteristics.buyers\_coming, Сharacteristics.buyers\_consumption, Сharacteristics.buyers\_remainder });

chart8.Series["Data"].Points.DataBindXY(new string[] { "Запас", "Витрати", "Ємісія", "Остача" }, new int[] { Сharacteristics.bank\_stock, Сharacteristics.bank\_consumption, Сharacteristics.bank\_emission, Сharacteristics.bank\_remainder });

if (Сharacteristics.bank\_emission == 1000)//кризис

{

Economy\_events.result += "---------------- Кінець програми ----------------\n";

timer1.Stop();

MessageBox.Show("У країні настала криза!", "Увага", MessageBoxButtons.OK);

}

}

## 4.3 Тестування методом «білого ящика»

Блок-схема функції Grafic\_Update зображена на рис. 4.1.

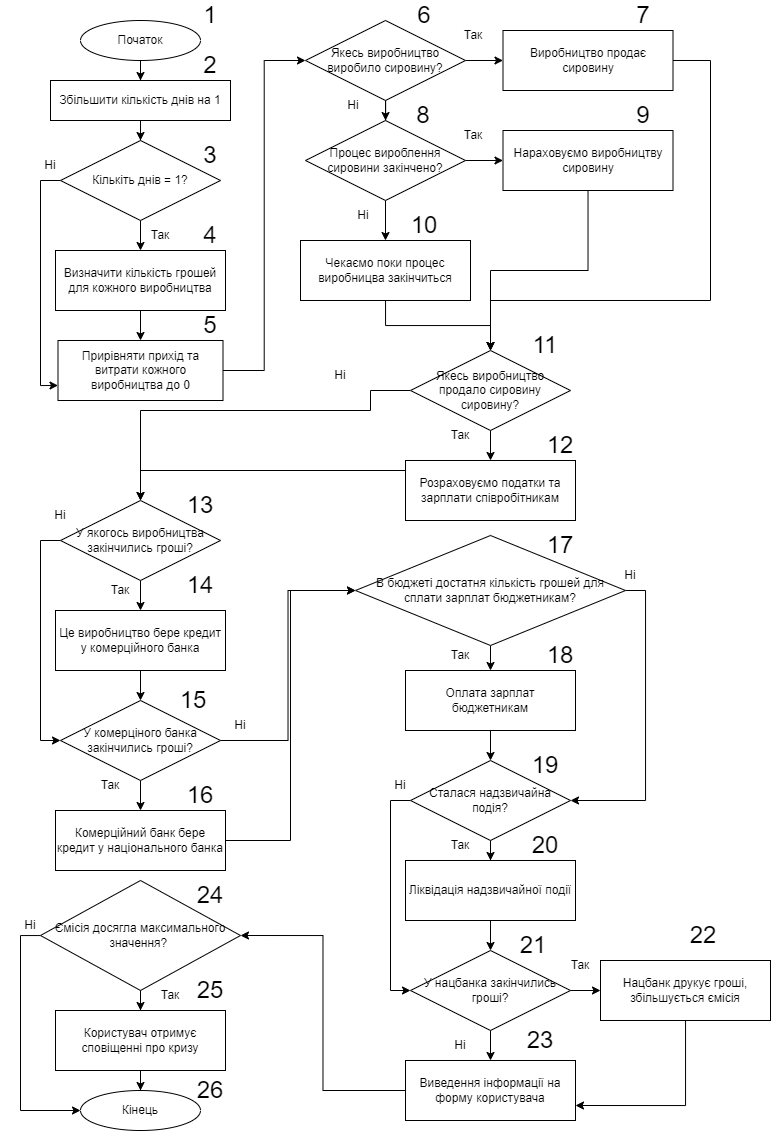


Рисунок 4.1 – Блок-схема функції Grafic\_Update

Тести функції Grafic\_Update описані у табл. 4.1.

Вхідні та вихідні дані представлені у такому порядку:

1. Кількість днів;

2. Остача кожного виробництва (виробника сировини, виробника засобів виробництва, виробника споживчих товарів, торгівлі та послуг відповідно);

3. Кількість сировини кожного виробництва (виробника сировини, виробника засобів виробництва, виробника споживчих товарів, торгівлі та послуг відповідно);

4. Кількість днів до кінця виготовлення сировини кожного виробництва (виробника сировини, виробника засобів виробництва, виробника споживчих товарів, торгівлі та послуг відповідно);

5. Остача комерційного банку;

6. Остача бюджету;

7. Остача нацбанку;

8. Емісія нацбанку.

Таблиця 4.1 – Тести функції Grafic\_Update

|  |  |  |
| --- | --- | --- |
| Номер тесту | Вхідні значення | Вихідні значення |
| 1 | 1. 0  2. 0 0 0 0  3. 0 0 0 0  4. 3 5 2 1  5. 0  6. 0  7. 0 | 1. 0  2. 0 0 0 0  3. 0 0 0 0  4. 3 5 2 1  5. 0  6. 0  7. 0 |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |