Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Горгальова Дмитра Євгенійовича*
(ПІБ)

академічної групи *122-18ск-1*
(шифр)

спеціальності *122 Комп'ютерні науки*
(код і назва спеціальності)

освітньої програми *Комп'ютерні науки*
(назва освітньої програми)

на тему: *Розробка інформаційно-навчального додатку для вивчення структур даних на прикладі «червоно-чорного дерева»*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*доц. Гуліна І.Г.*			
розділів:				
спеціальний	*доц. Гуліна І.Г.*			
економічний	*доц. Касьяненко Л.В.*			
Рецензент	*доц. Шедловський І.А.*			
Нормоконтролер	*доц. Гуліна І.Г.*			

Дніпро
2021

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ І.М. Удовик
(підпис) (прізвище, ініціали)

« » _____ 2021 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *122-18ск-1* *Горгальова Дмитра Євгенійовича*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи *Розробка інформаційно-навчального додатку для вивчення структур даних на прикладі «червоно-чорного дерева»*

затверджена наказом ректора НТУ «ДП» від « » 2021 р. №

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів виробничої практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі.*** ***Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***13.05.2021 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***27.05.2021 р.***

Завдання видав _____ *доц. Гуліна І.Г.*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Горгальов Д.Є.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *11.01.2021 р.*

Термін подання кваліфікаційної роботи до ЕК: *14.06.2021 р.*

РЕФЕРАТ

Пояснювальна записка: 93 с., 18 рис., 3 дод., 26 джерел.

Об'єкт розробки: інформаційно-навчальний додаток для вивчення структур даних на прикладі «червоно-чорного дерева».

Мета кваліфікаційної роботи: підвищення ефективності вивчення структур даних на прикладі «червоно-чорного дерева».

У вступі розглядається аналіз та сучасний стан проблеми, конкретизується мета кваліфікаційної роботи та галузь її застосування, наведено обґрунтування актуальності теми та уточнюється постановка завдання.

У першому розділі проведено аналіз предметної галузі, визначено актуальність завдання та призначення розробки, розроблена постановка завдання, задані вимоги до програмної реалізації, технологій та програмних засобів.

У другому розділі виконано аналіз існуючих рішень, обрано вибір платформи для розробки, виконано проектування і розробка програми, наведено опис алгоритму і структури функціонування підсистеми, визначені вхідні і вихідні дані, наведені характеристики складу параметрів технічних засобів, описаний виклик та завантаження застосунку, описана робота програми.

В економічному розділі визначено трудомісткість розробленої інформаційної підсистеми, проведений підрахунок вартості роботи по створенню системи та розраховано час на її створення.

Практичне значення полягає у створенні інформаційного додатку для вивчення структур даних на прикладі «червоно-чорного дерева». Написані тексти з головної теорії і тести для закріплення знань користувача.

Актуальність програмного продукту визначається великим попитом на подібні роботи.

Список ключових слів: СТРУКТУРА ДАНИХ, РАЦІОНАЛІЗАЦІЯ, ОРГАНІЗАЦІЯ ДАНИХ, ЧАСОВА СКЛАДНІСТЬ, МОДЕЛЮВАННЯ, НАВЧАЛЬНИЙ ДОДАТОК, ІНФОРМАЦІЙНА СИСТЕМА.

ABSTRACT

Explanatory note: 93 p., 19 figs., 3 apps., 26 sources.

Object of development: information-educational application for studying data structures on the example of "red-black tree".

The purpose of the diploma project: to increase the efficiency of studying data structures on the example of "red-black tree".

The introduction considers the analysis and current state of the problem, specifies the purpose of the qualification work and its scope, provides a justification for the relevance of the topic and clarifies the problem.

In the first section the analysis of the subject area is carried out, the urgency of the task and purpose of development are defined, the statement of the task is developed, requirements to software realization, technologies and software are set.

The second section analyzes the existing solutions, selects the platform for development, performs design and development of the program, describes the algorithm and structure of the subsystem, determines the input and output data, provides characteristics of the parameters of hardware, describes the call and application load, describes the program.

In the economic section, the complexity of the developed information subsystem is determined, the cost of work on the creation of the system is calculated and the time for its creation is calculated.

The practical significance lies in the creation of an information application for studying data structures on the example of the "red-black tree". Written texts on the main theory and tests to consolidate user knowledge.

The relevance of the software product is determined by the high demand for such work.

Keywords: DATA STRUCTURE, RATIONALIZATION, DATA ORGANIZATION, TIME COMPLEXITY, MODELING, LEARNING APPENDIX, INFORMATION SYSTEM.

ЗМІСТ

СПИСОК УМОВНИХ ПОЗНАЧЕНЬ

ЕОМ - електронно-обчислювальна машина;

ІС - інформаційна система;

ПЗ - програмне забезпечення;

ОС - операційна система;

ІТ - інформаційні технології.

ВСТУП

Кількість інформації, яку ми виробляємо та яку компаніям доводиться опрацьовувати, без сумніву, величезна і зростає щодня. Щоб використовувати потужності обчислювальних машин якомога раціональніше, комп'ютерні науковці винаходять та використовують алгоритми й структури даних. Структури даних – це спосіб організації даних. Зі структурами даних також пов'язані алгоритми, які виконуються з елементами при вставці чи видаленні.

Структури даних активно використовуються в мовах програмування та операційних системах, де кожна мілісекунда має значення. Червоно-чорні дерева використовуються у ядрі операційної системи «Linux» у планувальнику виконання задач («Completely Fair Scheduler»). У мові програмування «Java 8 за допомогою червоно-чорних дерев реалізована» структура даних «HashMap». Прикладів безліч й існують вони для кожної структури даних.

Метою даної роботи є познайомити користувача зі структурами даних на прикладі червоно-чорного дерева, продемонструвати його роботу за рахунок розробки та використання відповідного інформаційно-навчального додатку.

Для вивчення структур даних необхідно мати хорошу уяву і багато прикладів, тому допомогти зрозуміти тему має інтерактивна візуалізація вищезгаданої структури даних, на якій користувач побачить розташування елементів у дереві та зможе додати/видалити/шукати елементи. Також користувач матиме змогу побачити числові значення часу, витраченого на виконання операцій, порівняти його з часом аналогічної операції для структури даних бінарне дерево пошуку і стандартного масиву, вбудованого в мову C#. Закріпити власні знання можна виконавши випадково згенерований тест з 5 питань.

Отже, для досягнення поставленої мети в роботі ставляться наступний перелік задач:

– зробити модель червоно-чорного дерева, з якою зможе взаємодіяти користувач;

– зробити алгоритм та структуру додатку;

– написати програмний код для реалізації червоно-чорного дерева;

– створити невеликий легкий для розуміння інформаційно-навчаоьний додаток зі структур даних;

– реалізувати інтерфейс тестування для самоперевірки;

– виконати тестування додатку.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Загальні відомості з предметної галузі

Структура даних – це спосіб організації даних у комп’ютерах. Для певних задач існують оптимальні структури даних, які дозволяють значно зменшити час і ресурси, витрачені на роботу з даними. Найвідомішою структурою даних, яка є у кожній мові програмування, є масив – елементи, доступ до яких відбувається за індексом необхідного елементу. Основними операції з будь-якою структурою даних є пошук, вставка і видалення.

Зазвичай структури даних реалізуються за допомогою вказівників – типу даних, який у собі містить значення елемента і посилання на інші вказівники.

Прикладом структури даних є стек (рис.1.1.) (англ. stack — стек), елементи в якому містяться за принципом «last in first out», а операції виконуються лише з останнім доданим (найвищим) елементом стека. Стек можна уявити як книги, що лежать одна на одній. Зрозуміло, що всі можливі операції – взяти останню покладену книгу (pop) або покласти ще одну книгу на гору книжок (push). Стеки використовуються, наприклад, у компіляторах, для перевірки наявності у кожної дужки закриваючої дужки.

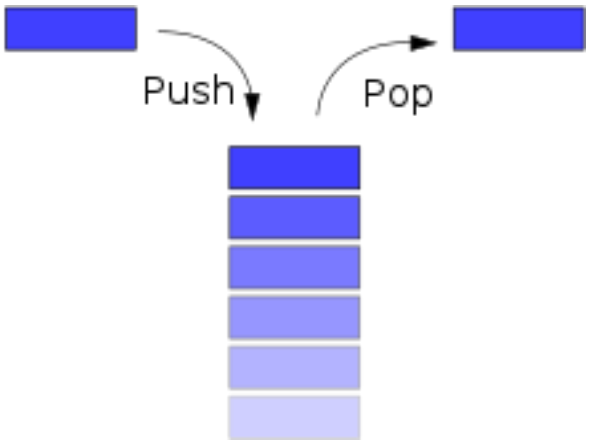

Рис. 1.1. Стек

Основною характеристикою структури даних є середня і найгірша часова складність у записі великого О. Вона означає, що, якщо середня часова складність – це O(n) (n – кількість елементів), то графік залежності часу операції від кількості елементів являє собою графік лінійної залежності (рис. 1.2.). Якщо часова складність O(1), то час не змінюється залежно від кількості елементів. Якщо $O(\log_2 n)$, то графік – логарифмічна залежність і так далі. Очевидно, що зі збільшенням кількості елементів, час для пошуку елементів у структурі даних з часовою складністю пошуку $O(\log_2 n)$ буде значно менше, ніж якби часова складність була б O(n).

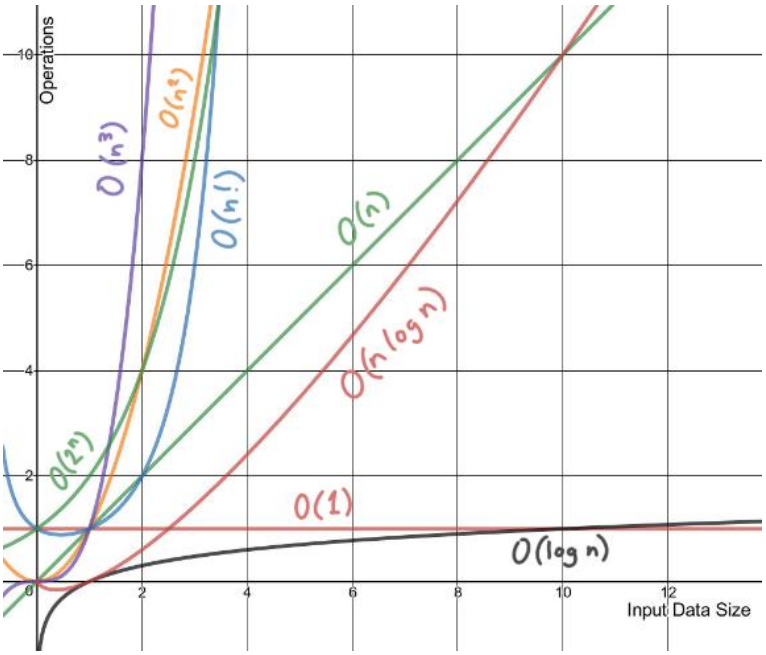

Рис. 1.2. Графік різних часових складностей.

Двійкове (бінарне) дерево пошуку (англ. binary search tree) - структура даних у вигляді дерева з теорії графів, тобто зв’язного графу без циклів. Кожна вершина має максимум дві дочірні вершини, до того ж, для кожної вершини значення лівої дитини вершини менше, за значення батьківської вершини, а значення правої більше значення батьківської вершини (рис. 1.3.).

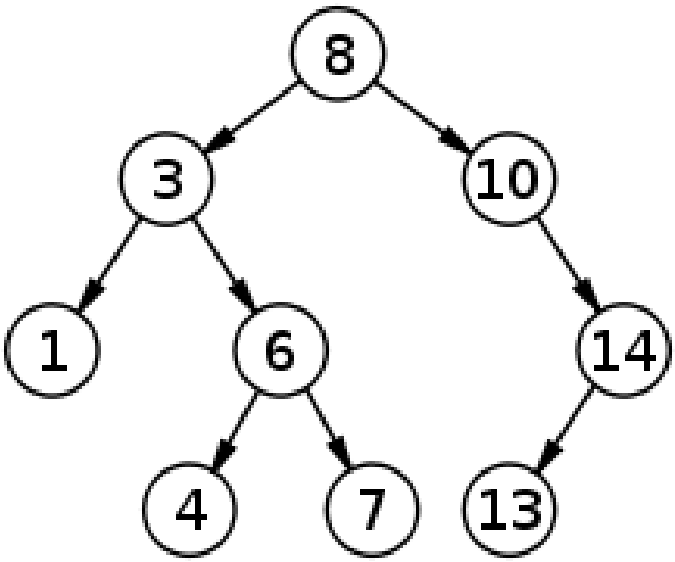

Рис. 1.3. Приклад двікового дерева. Коренева вершина має значення 8

Для такої структури даних, для пошуку, вставки і видалення часові складності однакові: середня - $O(\log_2 n)$, найгірша $O(n)$.

Алгоритм пошуку такий: порівнюємо значення вершини з шуканим значенням, якщо значення збігаються, ми знайшли те, що шукали, якщо значення більше/менше, то ми рекурсивно викликаємо функцію пошуку для правої/лівої дочірньої вершини відповідно. Оскільки для вставки ми шукаємо елемент, який стане батьком новому елементу, а для видалення ми шукаємо елемент який необхідно видалити, їхня часова складність збігається з часовою складністю операції пошуку.

Різниця між найгіршою і середньою складністю виникає через те, що у деяких випадках висота піддерев може сильно відрізнятися. Наприклад, якщо коренева вершина має порівняно мале значення, і всі наступні елементи більші за нього, то праве піддерево значно більше за ліве, і ми фактично отримаємо масив (рис. 1.4.).

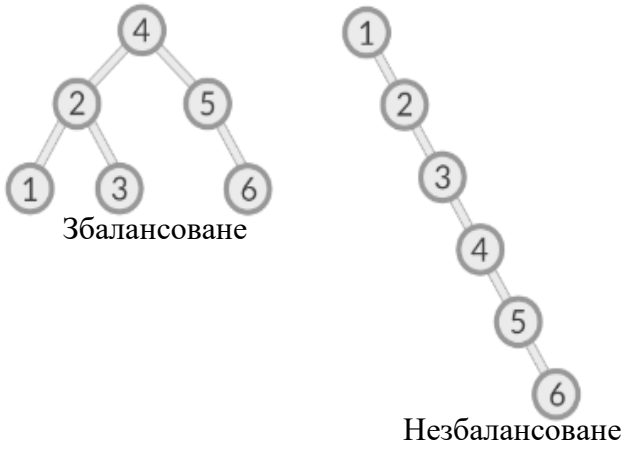

Рис. 1.4. Збалансоване/незбалансоване дерево

Це є найгіршим випадком, оскільки пошук зводиться до порівнянь значення вершини з шуканим значення і переходу до наступної вершини, що аналогічне пошуку значення у звичайному масиві. Натомість якщо дерево більш збалансоване, щоразу, коли значення не збігається зі значенням вершини, ми відкидаємо половину з варіантів, які залишилися. На дереві з шістьома елементами суттєвої різниці не помітити, проте, коли елементів велика кількість, бінарне дерево пошуку може майже повністю втратити свої переваги над масивом.

Наочним прикладом переваги бінарних дерев пошуку над масивами є процес пошуку слова у словнику на 1000 сторінок: можна почати з 500, якщо перша літера першого слова на цій сторінці далі за першу літеру шуканого слова, перегортаємо на 250. Тобто уявімо, що на сторінці 500 існує посилання на слова з 250 і 750 сторінки, на 250 – на слова з 125 і 375 і так далі. Так працює пошук у бінарному дереві пошуку. А можна почати з 1 сторінки і перевіряти першу літеру першого слова на кожній сторінці. Так працює пошук у масиві.

Червоно-чорне дерево. Бінарні дерева пошуку і справді гарантують повільне зростання часу виконання операції зі зростанням кількості елементів (часову складність O(log2n)), проте це можливо лише у найкращому випадку, імовірність якого зменшується зі збільшенням кількості елементів. Для того,

щоб дерево залишалося збалансованим, а середня часова складність дорівнювала найгіршій часовій складності, були розроблені самозбалансовані двійкові дерева пошуку, одним з яких є структура даних червоно-чорне дерево.

RB-Властивості. Для балансування мають бути збереженими такі властивості (далі – RB-властивості):

– кожна вершина або червона, або чорна;

– корінь дерева – чорний;

– кожний лист (нульова кінцева вершина, також зветься NIL) чорний;

– якщо вершина червона, обидві її дочірні вершини чорні (червона вершина не може мати червоних дочірніх вершин);

– усі прості шляхи від будь-якої вершини до листів проходять через однакову кількість чорних вершин.

Такі властивості надають червоно-чорному дереву додаткового обмеження: найдовший шлях з кореня до будь-якого листа перевищує найкоротший шлях не більше ніж вдвічі. «Чорна висота», кількість чорних елементів на шляху від кореня до листа, залишається сталою, тобто дерево відносно збалансоване, що гарантує однакову найгіршу і середню часову складність.

Для опису операцій збереження RB-властивостей, будуть використані терміни «(ліва/права) дитина», «брат», «батько», «дядько» і «дідусь» елемента ікс (рис.1.5).

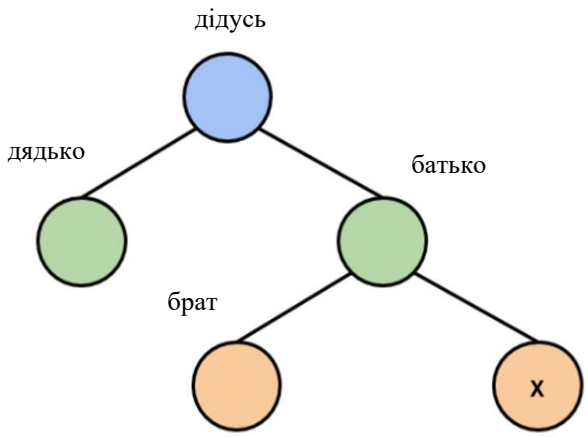

Рис. 1.5. "Сім'я" вершини x

Алгоритм вставки зі збереженням RB-властивостей:

– Знайти вершину батька (методом порівнювання значення нової вершини і рекурсії відносно правої/лівої дитини).

– «Перефарбувати» нову вершину у червоний колір.

– Якщо нова вершина – корінь дерева, змінюємо колір на чорний.

– Якщо батько і дядько нової вершини червоного кольору, змінюємо колір батька і дядька на чорний, колір дідуся на червоний. Повторюємо процес збереження RB-властивостей для дідуся.

– Якщо батько червоного кольору, а дядько чорного кольору або є нульовою вершиною, існує чотири варіанти:

- нова вершина – ліва дитина свого батька, батько – ліва дитина свого батька (Left Left Case, рис. 1.6.);

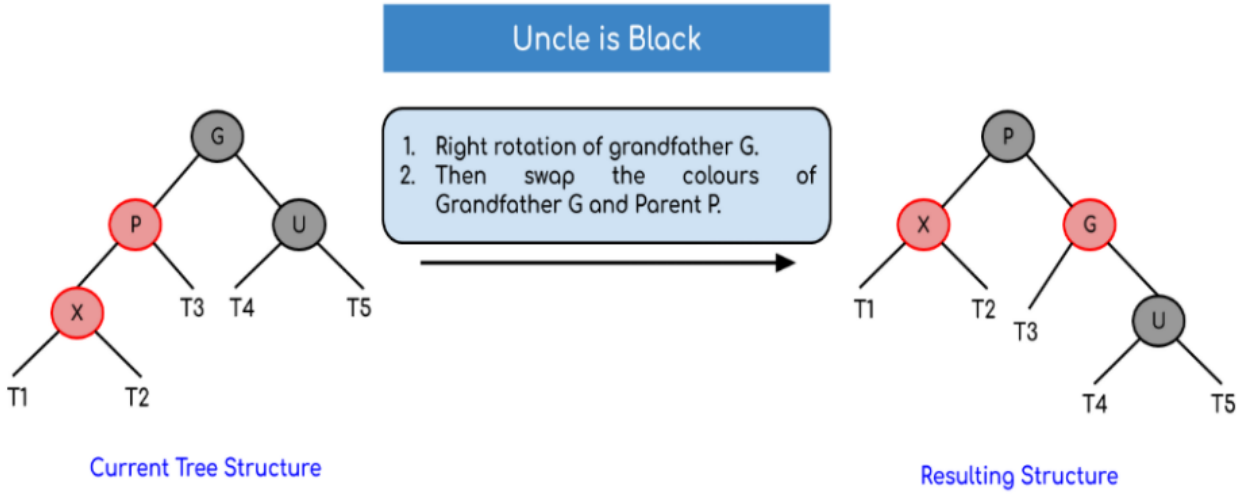

Рис. 1.6. Left Left Case

- нова вершина – права дитина свого батька, батько – ліва дитина свого батька (Left Right Case, рис. 1.7.);

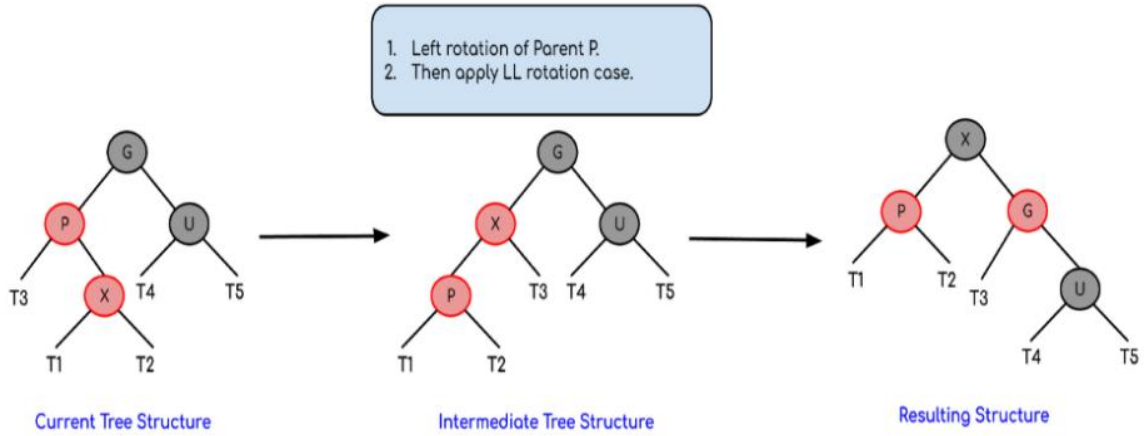

Рис. 1.7. Left Right Case

- нова вершина – права дитина свого батька, батько – права дитина свого батька (Right Right Case), випадок симетричний випадку на рис. 1.6.;
- нова вершина – ліва дитина свого батька, батько – права дитина свого батька (Right Left Case), випадок симетричний випадку на рис. 1.7.

– Для балансування дерева використовувався «поворот» бінарного дерева (рис.1.8.). Ця операція є основною для ребалансування дерев після вставки елементів.

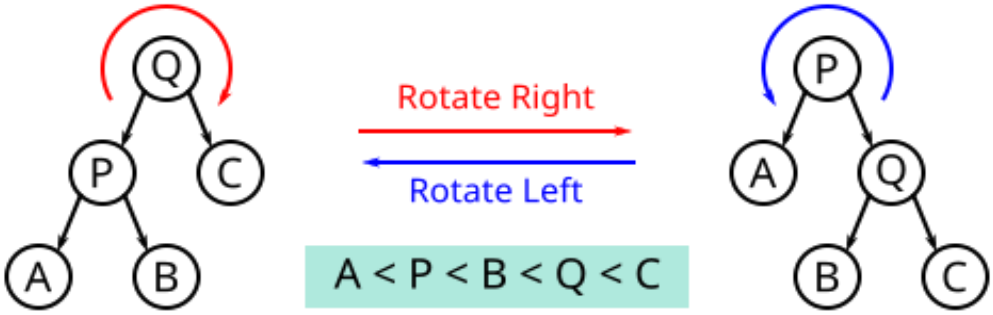

Рис. 1.8. «Поворот» бінарного дерева.

Алгоритм видалення зі збереження RB-властивостей. Якщо для збереження властивостей після вставки ми в основному перевіряли колір батька й дядька, то у випадку з видаленням, ми перевірятимемо вершину-брата, для

зручності назвемо її s. Вершину, яку необхідно видалити, назвемо v, а її вершину-дитину – u. Тоді, алгоритм такий:

За значенням шукаємо вершину, назвемо її v, яку треба видалити.

Якщо ця вершина немає дітей – просто видаляємо. У цьому випадку RB-властивості не можуть бути порушеними.

Якщо ця вершина має одну дитину – видаляємо цю вершину, у батька видаленої вершини змінюємо посилання на видалену вершину на посилання на дитину видаленої вершини. Виконуємо операції для збереження властивостей.

Якщо дитина має дві дитини, замінюємо її значення на значення найменшої (найлівішої) вершини з правого піддерева, видаляємо найменшу вершину правого піддерева (викликаємо цей же алгоритм видалення, тільки вже для неї).

Якщо v або u червона – видаляємо вершину v, ставимо на її місце вершину u, фарбуємо u в чорний колір. Чорна висота не змінюється.

Якщо і u, і v чорні:

Якщо s чорна і має бодай одну червону дитину, робляться повороти, налогічні тим, що були при вставці:

Left Left Case;

Left Right Case;

Right Right Case (рис.1.9.);

Right Left Case (рис.1.10.).

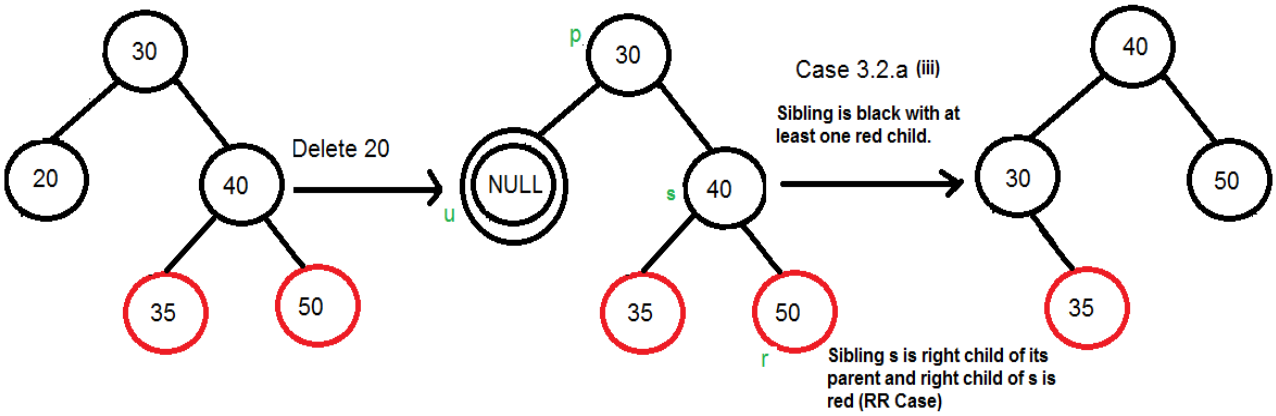

Рис. 1.9. Right Right Case

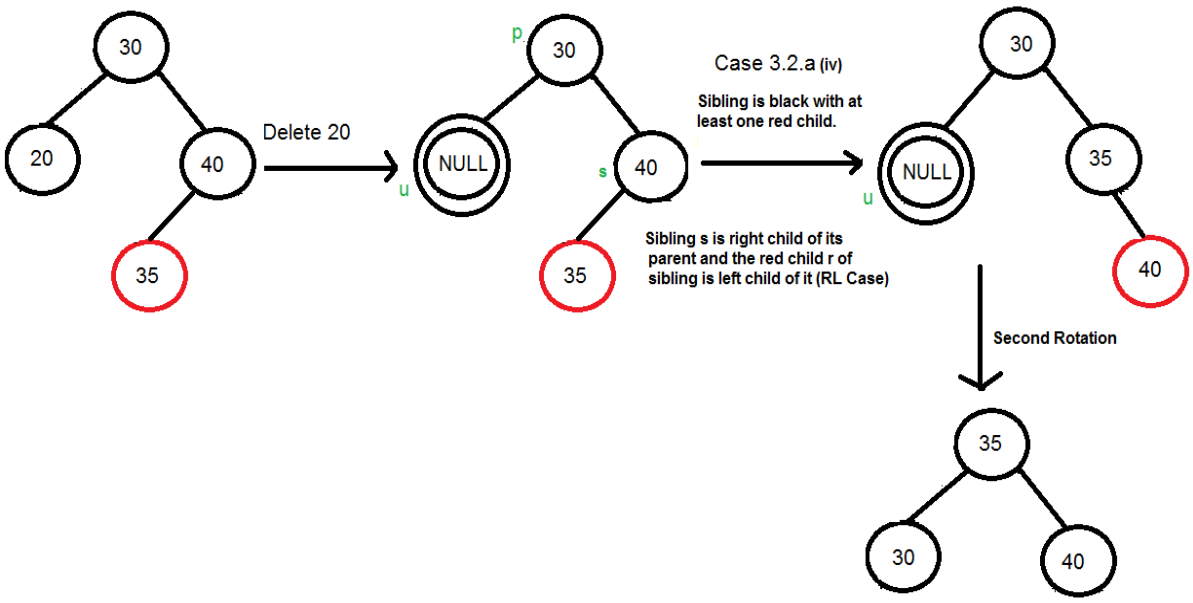

Рис. 1.10. Right Left Case

Якщо s чорна і має дві чорні дитини, надаємо u статусу double black, якщо батько u червоний – змінюємо його на чорний, якщо колір батька чорний, передаємо йому статус double black, рекурсією доходимо до червоного батьківського елемента і змінюємо його колір на чорний, таким чином позбувшись вершини зі статусом double black і зберігши чорну висоту дерева.

Якщо s червона, рухаємо її «нагору» виконавши поворот дерева. Отримуємо випадок b, описаний вище і робимо відповідні операції (рис.1.11.)

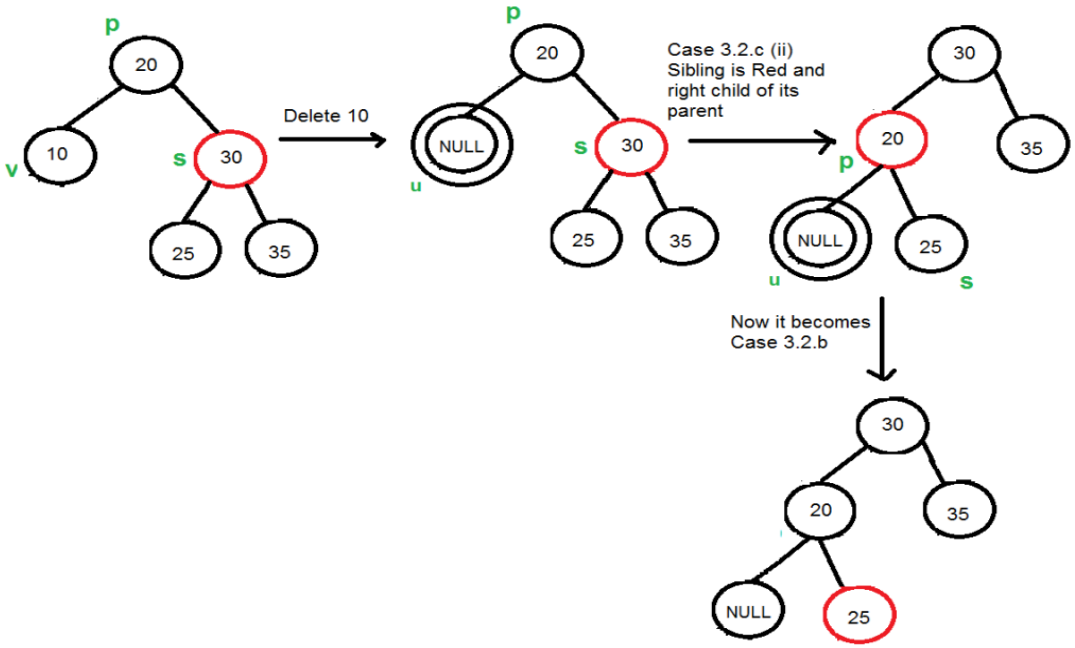

Рис. 1.11. Результат виконання операції перетворення дерева

1.2. Призначення розробки та галузь застосування

Іінформаційна система, що розробляється, в першу чергу призначена для підвищення ефективності вивчення структур даних на прикладі червоно-чорного дерева. Пояснити основні операції з червоно-чорним деревом: побудова, вставка, видалення, пошук, балансування. Порівняти з іншими структурами даних при розв’язку задач пошуку. Для закріплення отриманих знань, створення тесту за даною темою.

1.3. Підстава для розробки

Відповідно до освітньої програми, згідно навчального плану та графіків навчального процесу, в кінці навчання студент виконує кваліфікаційну роботу. Тема роботи узгоджується з керівником проекту, випускаючою кафедрою, та затверджується наказом ректора.

Отже, підставами для розробки кваліфікаційної роботи є:

- освітня програма спеціальності 122 «Комп’ютерні науки»;
- навчальний план та графік навчальногопроцесу;
- наказ ректора Національного технічного університету «Дніпровська політехніка» № 317-с від 07.06 .2021 р;
- завдання на кваліфікаційну роботу на тему «Розробка інформаційно-навчального додатку для вивчення структур даних на прикладі «червоно-чорного дерева»».

1.4. Постановка завдання

Отже, для досягнення поставленої мети в роботі ставляться наступний перелік задач:

– зробити модель червоно-чорного дерева, з якою зможе взаємодіяти користувач;

– зробити алгоритм та структуру додатку;

– написати програмний код для реалізації червоно-чорного дерева;

– створити невеликий легкий для розуміння інформаційно-навчаоьний додаток зі структур даних;

– реалізувати інтерфейс тестування для самоперевірки;

– виконати тестування додатку.

1.5. Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

До додатку, що розобляється в роботі, за мету ставляться наступні вимоги до функціональних характеристик:

- зручний інтуітвно-зрозумілий інтерфейс;
- невимогливість до складу програмно-апаратного комплексу.

1.5.2. Вимоги до інформаційної безпеки

Для уникнення некоректної роботи програми необхідно реалізувати:

– контроль вхідних даних;

– обробку виняткових ситуацій;

– виведення повідомлень про помилки;

– можливість повторного введення даних;

– можливість безперервної роботи протягом не менше 120 годин (5 діб);

– платформну незалежність;

– вірогідність виникнення не більше 2 логічних помилок на 1000 операторів за 1 рік експлуатації.

При роботі з програмою достатньо встановлених в системі вимог до інформаційної безпеки. Особливі вимоги до інформаційної безпеки не встановлюються.

Виключень з боку антивірусних програм та програм захисту не потребує.

Програма не потребує прав адміністратора чи спеціальної кваліфікації.

1.5.3. Вимоги до складу та параметрів технічних засобів

Апаратні вимоги:

- Двоядерний процесор частотою 2 GHz або вище.
- 4096 MB RAM.
- 100 MB вільного місця на жорсткому диску.
- Відеокарта — 1024 Mb.

1.5.4. Вимоги до інформаційної та програмної сумісності

Вимоги до інформаційних структур, вихідним кодам, мов програмування та програмним засобам, використаним для розробки та функціонування розробленої системи не встановлюються за винятком того що програма повинна бути невимогливою до програмно-апаратного комплексу та встановлено наступні:

- Версія ОС Windows: 7, 8 та новіші.
- NET.Framework 4.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Функціональне призначення системи

Іінформаційна система, що розробляється, в першу чергу призначена для підвищення ефективності вивчення структур даних на прикладі червоно-чорного дерева. Пояснити основні операції з червоно-чорним деревом: побудова, вставка, видалення, пошук, балансування. Порівняти з іншими структурами даних при розв’язку задач пошуку. Для закріплення отриманих знань, створенний тест за даною темою.

2.2. Опис застосованих математичних методів

Оскільки особливості предметної галузіі задачі, що вирішується в данній кваліфікаційній роботі не потребують застосування додаткових математичних методів, то при розробці застосунку для роботи з системою математичні методи не використовувалися. Методи, що було використано, наведено в п.1.1.

2.3. Опис використаних технологій та мов програмування

Програму створено в середовищі програмування Microsoft Visual Studio 2017 мовою C#. Було використано технології об'єктно-оріентованого програмування та компонентного програмування.

Об'єктно-оріентоване програмування (ООП) - головним елементом алгоритму є класи - нові типи даних, що розширюють можливості мови. Використання класів дозволяє сховати від програміста більшість чорнової роботи, тому що змінні й підпрограми ховаються в класах і викликаються невидимо для програміста. Це дозволило писати великі й складні програми, тому що в процедурних мовах з ростом програми різко збільшувалося число

помилок. Проте, усередині класи пишуться, як і звичайні процедурні програми.

C# відноситься до сім'ї мов із C-подібним синтаксисом, з них її синтаксис найбільш близький до C++ і Java. Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (у тому числі операторів явного і неявного приведення типу), делегати, атрибути, події, властивості, узагальнені типи і методи, ітератори, анонімні функції з підтримкою замикань, винятки, коментарі у форматі XML.

У принципі, явного лідера немає, тому при виборі мови потрібно враховувати те, що важливіше - швидкість розробки або швидкість роботи готового додатку. Можна, звичайно, зробити і багатомовний проект, але поєднувати такі мови, як Java, C++ і C# - технічно не найпростіше завдання. Хоча в деяких програмах воно і вирішене, варто взяти до уваги, що такий підхід збільшує витрати на розробку.

Ще один важливий фактор - кількість підтримуваних платформ. Для Java і C# це не особливо критично, а ось на C++ весь платформно-залежний код потрібно буде писати окремо для кожної ОС. Відповідно, чим менше підтримуваних ОС, тим менший код.

C# – об’єктно-орієнтована мова програмування. Розроблена в 1998-2001 роках групою інженерів під керівництвом Андерса Хейлсберга в компанії Microsoft як мова розробки додатків для платформи Microsoft .NET Framework і згодом був стандартизований як ECMA-334 та ISO/IEC 23270.

Перейнявши багато чого від своїх попередників – мов C++, Java, Delphi і Smalltalk – C#, спираючись на практику їх використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад, C# не підтримує множинне успадкування класів (на відміну від C++).

C# розроблявся як мова програмування прикладного рівня для CLR і, як такий, який залежить, насамперед, від можливостей самої CLR. Це стосується, перш за все, системи типів C#, яка відображає BCL. Присутність або відсутність тих або інших виразних особливостей мови диктується тим, чи

може конкретна мовна бути трансльована у відповідні конструкції CLR. Так, з розвитком CLR від версії 1.1 до 2.0 значно збагатився і сам C#; подібної взаємодії слід чекати і надалі. (Проте ця закономірність була порушена з виходом C# 3.0, що представляє собою розширення мови, не спираються на розширення платформи .NET.) CLR надає C#, як і всім іншим .NET-орієнтованим мовам, багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, збірка сміття не реалізована в самому C#, а виробляється CLR для програм, написаних на C# точно так само, як це робиться для програм на VB.NET, J# і інше.

2.4. Опис структури системи та алгоритмів її функціонування

Для реалізації червоно-чорного дерева необхідно створити два класи: клас вказівника (RBNode) і клас самого дерева, який містить посилання на корінь дерева і в якому прописані алгоритми, необхідні для збереження RB-властивостей.

Клас RBNode. Цей клас є контейнером для даних, які знаходяться у дереві. До нього є кілька технічних умов:

- Посилання (поле типу RBNode) на ліву дитину.
- Посилання на праву дитину.
- Посилання на батька.
- Поле кольору (можна було б використовувати поле типу bool, проте у C# воно займає байт інформації, а не біт, тому використовується поле типу byte, яке займає байт, але дає змогу надати вершині статус double-black, червоний колір відповідає значенню 0, чорний – значенню 1).
- Методи-конструктори.

Таким чином, знаючи одну вершину, ми можемо отримати дані будь-якої вершини дерева. Наприклад, вершина-дядько для вершини x буде записана як x.parent.parent.left або x.parent.parent.right. Тобто ми посилаємося на дитину (.left/.right) вершини-діда для x (x.parent.parent).

Для цього написано такий код:

```
public class RBNode
{
public byte color; // 0 - red, 1 - black
public IComparable data; //щоб можна було задавати значення null
public RBNode left;
public RBNode right;
public RBNode parent;
public RBNode()
{
 this.color = 0;
this.left = null;
this.right = null;
}
public RBNode(IComparable newData)
{
this.data = newData;
this.left = null;
this.right = null;
}
public RBNode(IComparable newData, RBNode Left, RBNode Right)
{
this.data = newData;
this.left = Left;
this.right = Right;
}
}
```

Клас RBTree. Цей клас має містити посилання на кореневу вершину, посилання на вершину-лист, посилання на яку означатиме, що далі елементів

уже немає, а також методи для вставки, пошуку, видалення і відновлення RB-властивостей після них.

Будь-який набір елементів починається зі вставки. У моєму коді метод вставки, аналогічний вставці у звичайне двійкове дерево пошуку, і метод відновлення RB-властивостей – два різних методи, для більшої наочності різниці між цими двома структурами даних. Оскільки ця частина коду доволі велика, переглянути метод вставки можна на додатку А. Далі йде метод, який зберігає RB-властивості, який знаходиться в додатку А.

Для зручності, метод повороту двійкового дерева реалізований окремо. Метод повороту двійкового дерева (рисунок 2.1., 2.2.):

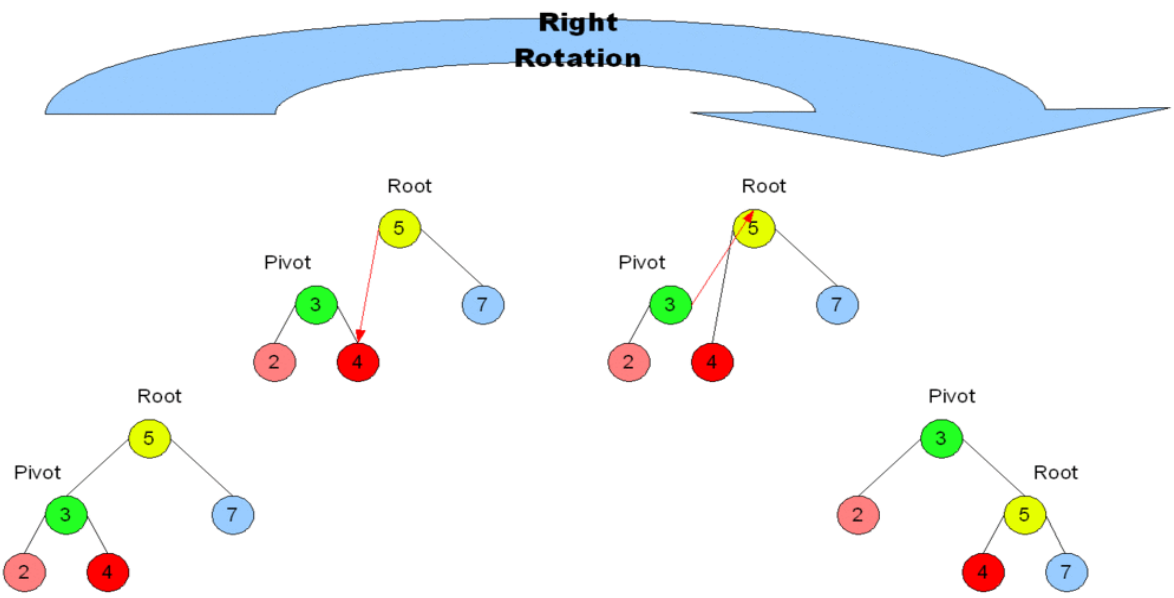

Рис. 2.1. Правий поворот дерева

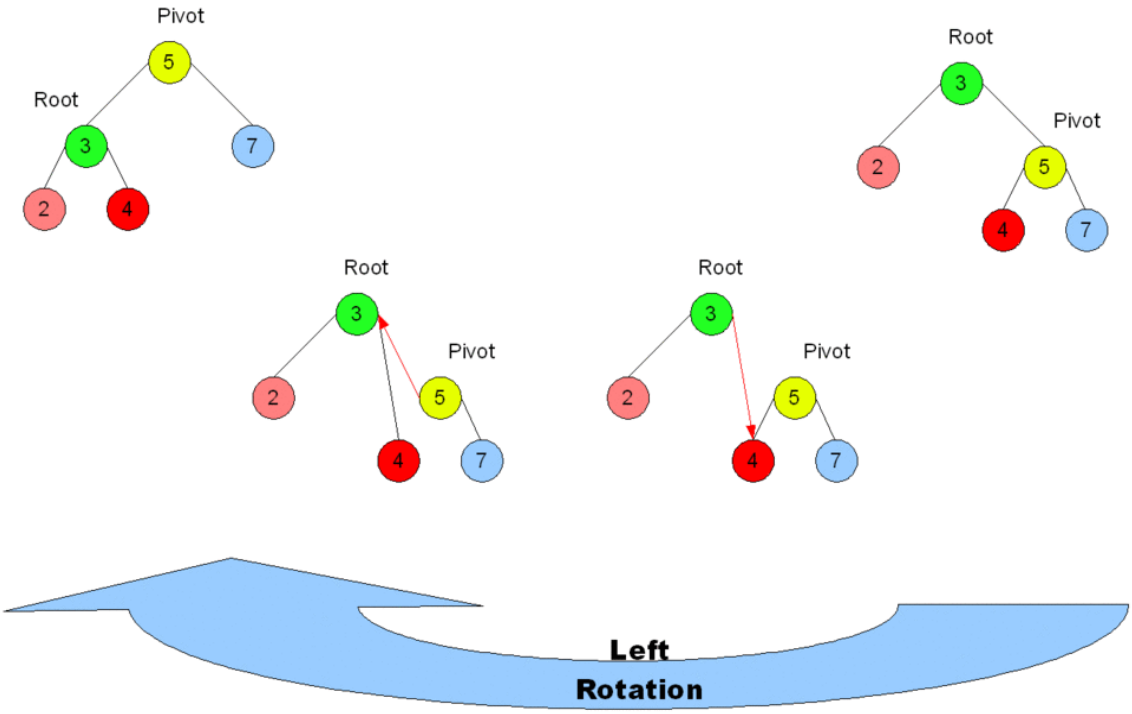

Рис.2.2. Лівий поворот дерева

```
enum Direction
{
   right,
   left
}
private void Rotate(RBNode pivot, Direction dr)
{
   RBNode rotRoot = pivot.parent;
   if (dr == Direction.right)
   {
      rotRoot.left = pivot.right;
      pivot.right.parent = rotRoot;
      pivot.right = rotRoot;
      pivot.parent = rotRoot.parent;
      rotRoot.parent = pivot;
```

```
        if (rotRoot == root)
        {
            root = pivot;
        }
//оновлюємо посилання з батьківської вершини на pivot
        else if (pivot.parent.right == rotRoot)
        {
            pivot.parent.right = pivot;
        }
        else
        {
            pivot.parent.left = pivot;
        }
    }
     else
     {
        rotRoot.right = pivot.left;
        pivot.left.parent = rotRoot;
        pivot.left = rotRoot;
        pivot.parent = rotRoot.parent;
        rotRoot.parent = pivot;
        if (rotRoot == root)
        {
            root = pivot;
        }
        else if (pivot.parent.right == rotRoot)
        {
            pivot.parent.right = pivot;
        }
        else
```

```
        {
            pivot.parent.left = pivot;
        }
    }
```

Метод видалення (див. додаток А.), він дещо важчий за вставку, проте не вимагає жодних нових вмінь програмування.

Після видалення, необхідно обов’язково викликати метод збереження RB-властивостей після видалення.

Зазвичай, у методі для виведення всіх елементів дерева немає потреби, але оскільки дана програма є навчальною і необхідно зробити модель дерева, треба розробити метод виводу дерева. Для цього є спеціальний термін – обхід дерева (англ. tree traversal) – і спеціальні рекурсивні алгоритми. Розглянемо реалізацію серединного пошуку вглиб (depth-first in-order binary tree search). При такому обході дерева, елементи виводяться у зростаючому порядку.

```
public void Display()
        {
            this.Display(root);
        }
private void Display(RBNode temp)
        {
            if (temp != freshNode)
            {
                Display(temp.left);
                Console.WriteLine(temp.data.ToString());
                Display(temp.right);
            }
        }
```

Якщо мета – вивести червоно-чорне дерево на pictureBox, задача дещо ускладнюється, проте алгоритм не змінюється. Просто тепер ми щоразу, як переходимо до наступного «рівня» вершин, додаємо до координати кола, яке

позначає вершину, width і height. Width = 2 * радіус кола, height – задана величина, яка позначає відстань між «рівнями».

```
static void DisplayTree(RBTree tree, PictureBox pb)
{
   g.Clear(Color.White);
   width = 0;
   DisplayTree(tree.root, 0);
}
static void DisplayTree(RBNode temp, int height)
{
   if(temp.data == null)
   {
      return;
   }
   height += l;
   DisplayTree(temp.left, height);

   if(temp.color == 0)
   {
      g.FillEllipse(new SolidBrush(Color.DarkRed), width, height, 2 * r, 2 * r);
    }
   else
   {
      g.FillEllipse(new SolidBrush(Color.Black), width, height, 2 * r, 2 * r);
   }
   g.DrawString(temp.data.ToString(), new Font(FontFamily.GenericSansSerif,
 (int)(r * 0.75)), new SolidBrush(Color.White), width + Convert.ToInt32(0.4 * r), hei
ght + Convert.ToInt32(0.4 * r));
   if(temp.parent == null)
   {
```

```
            g.DrawString("Корінь", new Font(FontFamily.GenericSansSerif, (int)(r *
 0.75)), new SolidBrush(Color.Black), width - r / 2, height - r - 5);
        }
        width += 2 * r;

        DisplayTree(temp.right, height);
    }
```

2.5. Обгрунтування та організація вхідних та вихідних даних програми

Вхідними даними є обрані та задані користувачем початкові дані та обраний режим роботи додатку.

Вихідними є дані побудованого дерева.

2.6. Опис роботи розробленої системи

2.6.1. Використані технічні засоби

Наведені вище технічні характеристики є рекомендованими, тобто при наявності технічних засобів не нижче зазначених, розроблений програмний вироб буде функціонувати відповідно до вимог щодо надійності, швидкості обробки даних і безпеки.

Для роботи системи можуть використовуватись різні типи комп’ютерів чи ноутбуків під управлінням ОС Windows та різних типів і характеристик.

Програма розроблена для робочих станцій під управлінням ОС Windows написана мовою програмуванням C# у середовищі Visual Studio 2017.

Згідно з довідкою розробника середовища, фірмою Microsoft, розміщеною на офіційному сайті для використання Visual Studio 2017 встановлено наступні системні вимоги Visual Studio 2017:

Середа Visual Studio 2017 підтримується на наступних операційних системах:

– Windows 7 з Service Pack 1;
– Windows 8.1;
– Windows 10.

Мінімальні вимоги до обладнання:

– Процесор з частотою не нижче 1,8 ГГц. Рекомендується використовувати як мінімум двоядерний процесор;
– 2 ГБ оперативної пам'яті, рекомендується 4 ГБ;
– вільного місця на жорсткому диску від 1 ГБ до 40 ГБ, залежно від встановлених компонентів;
– відеоадаптер з мінімальним дозволом 1280 на 720 пікселів.

Система не є вимогливою до складу інших технічних засобів.

Програма була розроблена на ноутбуці Asus X552MD з такими технічними характеристиками: екран 15.6" (1366x768) HD LED / Intel Celeron N2840 (2.16 ГГц) / RAM 4 ГБ / HDD 500 ГБ / nVidia GeForce 820M, 1 ГБ / DVD Super Multi / Wi-Fi / Bluetooth 4.0 / веб-камера.

2.6.2. Використані програмні засоби

Програму створено в середовищі програмування Microsoft Visual Studio 2019 мовою C#.

C# - сучасний об'єктно-орієнтований і тіпобезопасний мову програмування. C # дозволяє розробникам створювати безліч типів безпечних і надійних програм, які потребують екосистемі .NET. C# відноситься до широко відомому сімейству мов C, і здасться добре знайомим кожному, хто працював з C, C++, Java або JavaScript. Тут представлений огляд основних компонентів мови C# 8 і більше ранніх версій. Якщо ви хочете вивчити мову за допомогою інтерактивних прикладів, рекомендуємо попрацювати з вступними посібниками по C#.

C# - це об'єктно-і компонентно-орієнтована мова програмування. C# надає мовні конструкції для безпосередньої підтримки такої концепції роботи. Завдяки цьому C# підходить для створення і застосування програмних компонентів. З моменту створення мову C# збагатився функціями для підтримки нових робочих навантажень і сучасними рекомендаціями по розробці ПЗ.

Ось лише кілька функцій мови C#, які дозволяють створювати надійні та стійкі додатки. “Прибирання сміття” автоматично звільняє пам'ять, зайняту недоступними невикористовуваними об'єктами. Типи, що допускають значення null, забезпечують захист від змінних, які не посилаються на виділені об'єкти. Обробка винятків надає структурований і розширюваний підхід до виявлення помилок і відновлення після них. Лямбда-вирази підтримують прийоми функціонального програмування. Синтаксис LINQ створює загальний шаблон для роботи з даними з будь-якого джерела. Підтримка мов для асинхронних операцій надає синтаксис для створення розподілених систем. У C# діє єдина система типів. Всі типи C#, включаючи типи-примітиви, такі як int і double, успадковують від одного кореневого типу object. Всі типи використовують загальний набір операцій, а значення будь-якого типу можна зберігати, передавати і обробляти схожим чином. Більш того, C# підтримує як визначаються користувачами посилальні типи, так і типи значень. C # дозволяє динамічно виділяти об'єкти та зберігати спрощені структури в стеці. C # підтримує універсальні методи і типи, що забезпечують підвищену безпеку типів і продуктивність. C# надає ітератори, які дозволяють розробникам класів колекцій визначати призначені для користувача варіанти поведінки для клієнтського коду.

У C# особлива увага приділяється управлінню версіями для забезпечення сумісності програм і бібліотек при їх зміні. Питання управління версіями істотно вплинули на такі аспекти розробки C#, як роздільні модифікатори virtual і override, правила вирішення перевантаження методів і підтримка явного оголошення членів інтерфейсу.

Visual Studio 2019 забезпечує безпрецедентну продуктивність для будь-якого пристрою, додатки або платформи. Використовуйте Visual Studio 2019 для розробки додатків для Android, iOS, Windows, Linux, а також веб-додатків і хмарних додатків. Швидко пишіть код, виконуйте налагодження і діагностику з легкістю, часто тестируйте і впевнено створюйте випуски рішень. Visual Studio можна розширити і налаштувати, створивши власні розширення. Система управління версіями в цьому випуску робить розробку гнучкої, а спільну роботу - ефективною.

Для пошуку інформації використано Google Chrome.

2.6.3. Виклик та завантаження програми

Спосіб виклику програми з відповідного носія даних та умови його завантаження є стандартними для запуску виконуючих файлів при роботі в ОС Windows. Додаткових чи специфічних вимог щодо запуску програми не встановлено, програма не потребує спеціального завантаження та налаштування і може бути скопійована на інші носії інформації.

Для роботи потрібен ПК чи наутбук з ОС Windows. Щоб установити програму на комп’ютер, треба скопіювати папку «Program» та відкрити файл .exe.

2.6.4. Опис інтерфейсу користувача

Стартовою сторінкою є титульна сторінка, на якій відображено основні дані про роботу, аналогічні даним з титульної сторінки документації. При кліку по темі роботи відкривається головне меню з трьома кнопками: «Теорія і код», «Моделювання» і «Перевір себе», при кліку по яких здійснюється перехід на відповідні форми (рис. 2.3.).

Рис. 2.3. Головна форма додатку

На формі «Теорія і код» (рис. 2.4) користувач може прочитати невелику статтю на обрану в treeView тему, а також побачити головні частини коду реалізації червоно-чорного дерева.

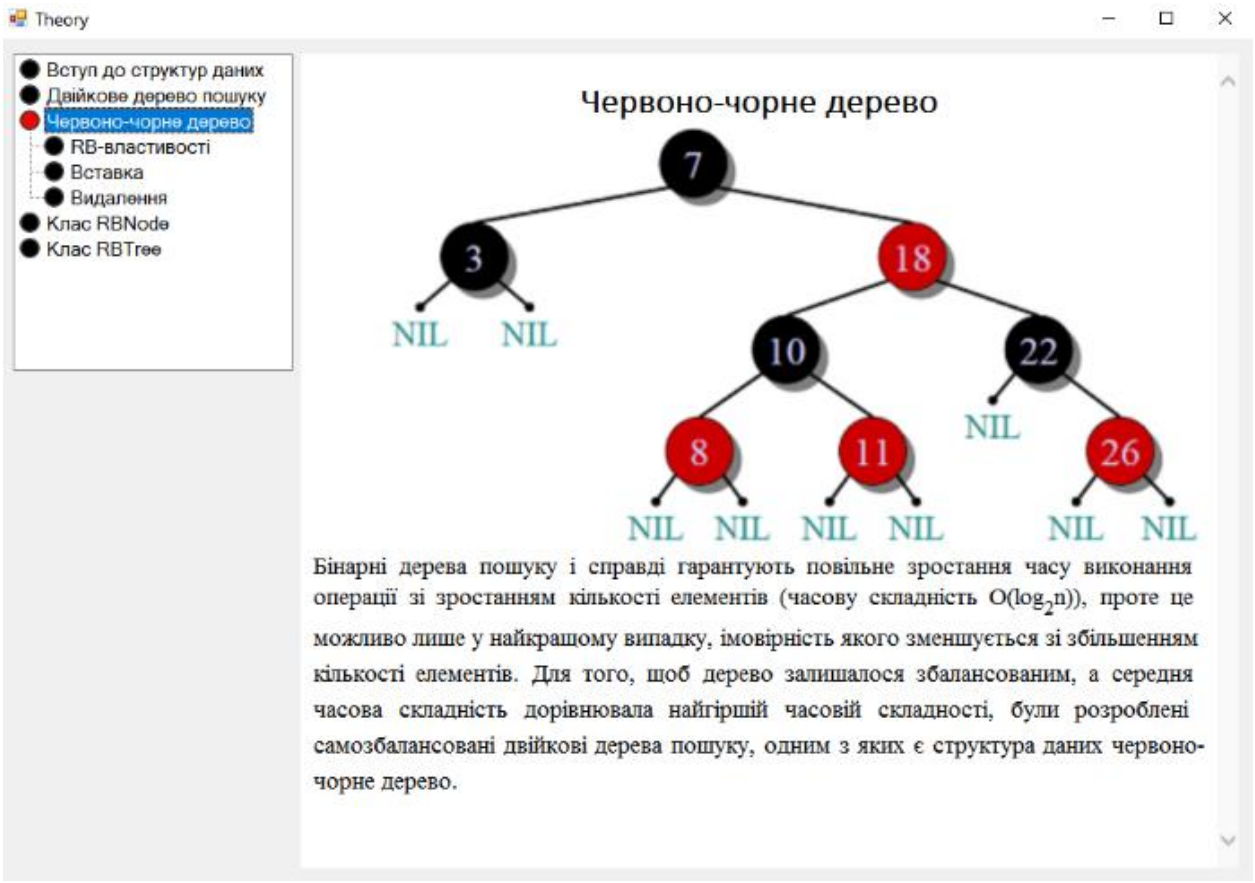

Рис.2.4. Форма додатку з теорією

На формі «Моделювання» користувач може попрацювати з червоно-чорним деревом (рис. 2.5.).

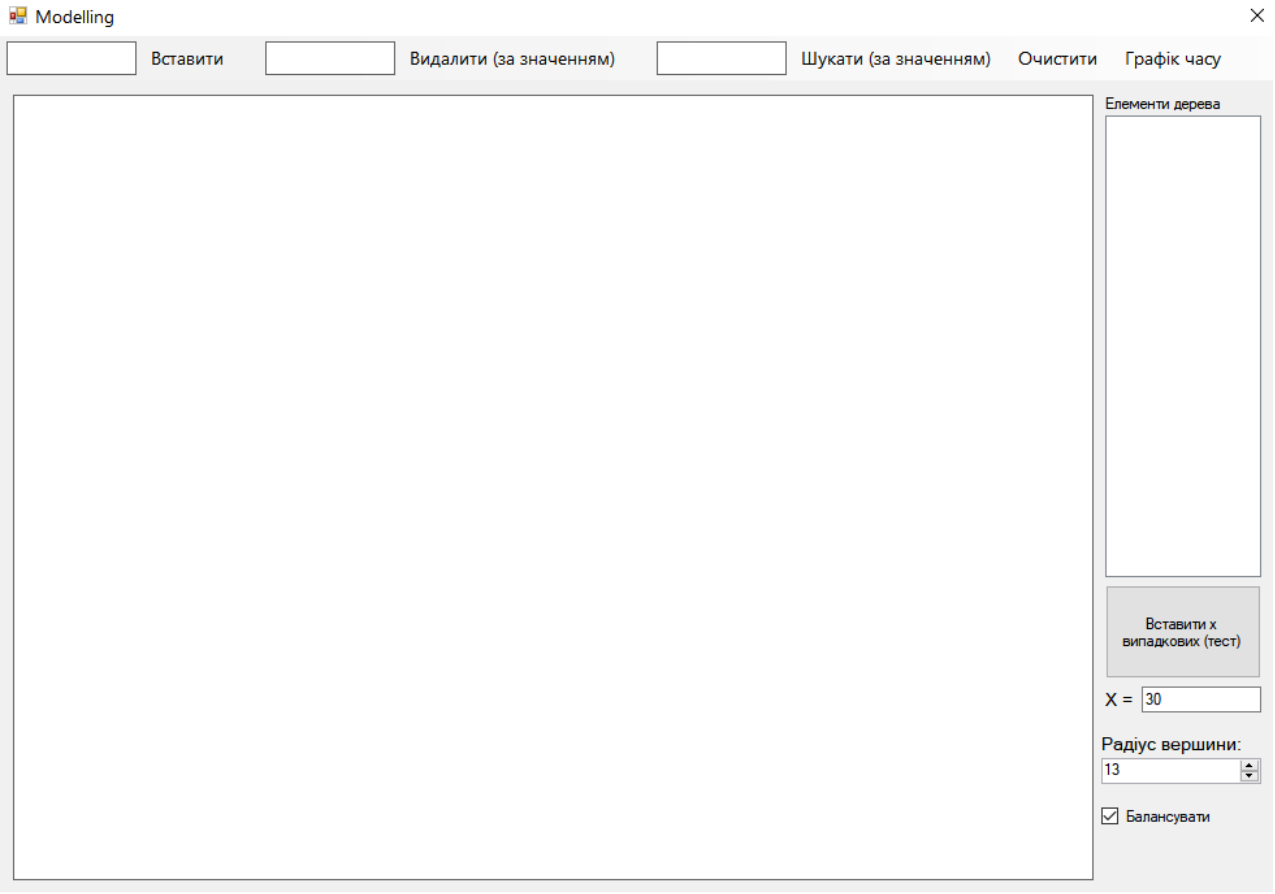

Рис. 2.5. Форма «Моделювання»

Для цього є кнопки «Вставити», «Видалити», «Шукати» з відповідними полями для значень вершини і кнопка Очистити у верхньому меню (menuStrip). Ще існує кнопка «Вставити x випадкових», яка вставляє у дерево x випадкових значень. Значення x можна ввести у текстове поле нижче. Відображення дерев, як і виведення елементів у listBox, відбувається автоматично. Користувач також може зняти галочку з чекбоксу «Балансувати» і побачити дерево, в яке вводилися аналогічні дані, але не проводилися методи балансування (рис. 2.6.). Детальніша інструкція наявна у самому додатку.

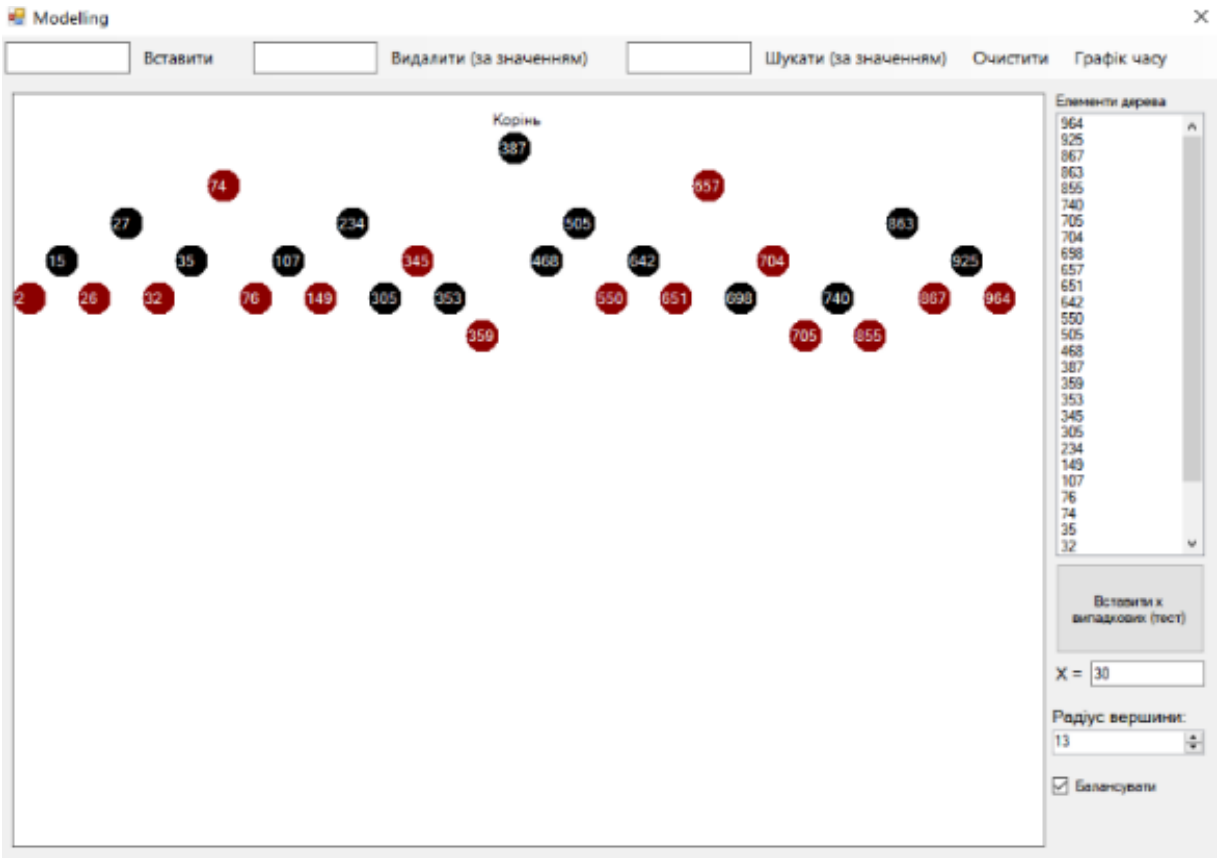

Рис. 2.6. Моделювання

На формі «Перевір себе» користувач проходить тест і отримує оцінку. П’ять питань обираються навмання зі списку питань (рис. 2.7.).

Рис.2.7. Форма «Перевір себе»

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Початкові дані:

1. передбачуване число операторів програми – 1300;
2. коефіцієнт складності програми – 1,6;
3. коефіцієнт корекції програми в ході її розробки – 0,05;
4. годинна заробітна плата програміста– 60грн/год;
5. коефіцієнтзбільшеннявитратпрацівнаслідокнедостатньогоопи сузадачі – 1,2;
6. коефіцієнткваліфікаціїпрограміста, обумовленийвід стажу роботи з даноїспеціальності – 1,2;
7. вартість машино-години ЕОМ –13грн/год.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д, \text{ людино-годин,} \quad (3.1)$$

де t_o- витрати праці на підготовку й опис поставленої задачі (приймається 50 людино-годин);

t_u - витрати праці на дослідження алгоритму рішення задачі;

t_a- витрати праці на розробку блок-схеми алгоритму;

t_n-витрати праці на програмування по готовій блок-схемі;

$t_{отл}$-витрати праці на налагодження програми на ЕОМ;

$t_д$ - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у програмногму забезпеченні, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \cdot C \cdot (1 + p),$$

де q - передбачуване число операторів (1300);

C - коефіцієнт складності програми (1,6);

p - коефіцієнт корекції програми в ході її розробки (0,05).

Звідси умовне число операторів в програмі:

$$Q = 1{,}6 \cdot 1300 \cdot (1 + 0{,}05) = 2184$$

Витрати праці на вивчення опису задачі t_u визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q \cdot B}{(75..85) \cdot k}\ \text{, людино-годин,}$$

де B - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі;

k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності.При стажі роботи від 3 до 5 років він складає 1,2.

Приймемо збільшення витрат праці внаслідок недостатнього опису завдання не більше 50% (B = 1,2). З урахуванням коефіцієнта кваліфікації k = 1,2, отримуємо витрати праці на вивчення опису завдання:

$$t_u = (2184 \cdot 1{,}2) / (75 \cdot 1{,}2) = 29{,}12 \text{ людино-годин}$$

Витрати праці на розробку алгоритму рішення задачі визначаються за формулою:

$$t_a = \frac{Q}{(20\ldots25)\cdot k}\ \text{, людино-годин,}\quad (3.2)$$

де Q – умовне число операторів програми;

k – коефіцієнт кваліфікації програміста.

Підставивши відповідні значення в формулу (3.2), отримаємо:

$$t_a = 2184 / (20 \cdot 1{,}2) = 91 \text{людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20\ldots25)\cdot k}\ \text{, людино-годин.}$$

$$t_n = 2184 / (20 \cdot 1{,}2) = 91 \text{ людино-годин.}$$

Витрати праці на налагодження програми на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4..5)\cdot k}\ \text{, людино-годин.}$$

$$t_{отл} = 2184 / (5 \cdot 1{,}2) = 364 \text{ чел.-ч.}$$

- за умови комплексного налагодження завдання:

$$t^k_{отл} = 1{,}5 \cdot t_{отл}\ \text{, людино-годин.}$$

$$t^{k}_{отл} = 1,5 \cdot 364 = 546 \text{ людино-годин.}$$

Витрати праці на підготовку документації визначаються за формулою:

$$t_{д} = t_{др} + t_{до} \text{ , людино-годин,}$$

де $t_{др}$-трудомісткість підготовки матеріалів і рукопису:

$$t_{др} = \frac{Q}{(15..20) \cdot k} \text{ , людино-годин,}$$

$t_{до}$ - трудомісткість редагування, печатки й оформлення документації:

$$t_{до} = 0,75 \cdot t_{др} \text{ , людино-годин.}$$

Підставляючи відповідні значення, отримаємо:

$$t_{др} = 2184 / (18 \cdot 1,2) = 101,11 \text{ людино-годин.}$$

$$t_{до} = 0,75 \cdot 101,11 = 75,83 \text{ людино-годин.}$$

$$t_{д} = 101,11 + 75,83 = 176,94 \text{ людино-годин.}$$

Повертаючись до формули (3.1), отримаємо повну оцінку трудомісткості розробки програмного забезпечення:

$$t = 50 + 29,12 + 91 + 91 + 364 + 176,94 = 802,06 \text{ людино-годин.}$$

3.2. Розрахунок витрат на створення програми

Витрати на створення ПЗ $K_{ПО}$ включають витрати на заробітну плату виконавця програми $З_{ЗП}$ і витрат машинного часу, необхідного на налагодження програми на ЕОМ:

$$K_{ПО} = З_{ЗП} + З_{МВ}, \text{ грн.}$$

Заробітна плата виконавців визначається за формулою:

$$З_{ЗП} = t \cdot C_{ПР}, \text{ грн,}$$

де: t - загальна трудомісткість, людино-годин;

$C_{ПР}$ - середня годинна заробітна плата програміста, грн/година

З урахуванням того, що середня годинна зарплата програміста становить 60 грн / год, отримуємо:

$$З_{ЗП} = 802{,}06 \cdot 60 = 48123{,}87 \text{ грн.}$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ, визначається за формулою:

$$З_{мв} = t_{отл} \cdot C_{мч}, \text{ грн,} \quad (3.3)$$

де $t_{отл}$ - трудомісткість налагодження програми на ЕОМ, год;

$C_{мч}$ - вартість машино-години ЕОМ, грн/год (13 грн/год).

Підставивши в формулу (3.3) відповідні значення, визначимо вартість необхідного для налагодження машинного часу:

$$З_{мв} = 364 \cdot 13 = 4732 \text{ грн.}$$

Звідси витрати на створення програмного продукту:

$$К_{ПО} = 48123{,}87 + 4732 = 52855{,}87 \text{ грн.}$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p}, \text{ міс.}$$

де B_k- число виконавців (дорівнює 1);

F_p - місячний фонд робочого часу (при 40 годинному робочому тижні F_p=176 годин).

Звідси витрати на створення програмного продукту:

$$T = 802{,}06 / 1 \cdot 176 \approx 4{,}5 \text{ міс.}$$

Висновок: в роботі розроблена програма з метою підвищення ефективності вивчення структур даних. Вартість даного програмного забезпечення становить 52 тис. грн. і не вимагає додаткових витрат як при розробці програми. Очікуваний час розробки становить 4,5 місяці. Цей термін пов'язаний зі значним числом операторів, і включає час на дослідження і розробку алгоритму вирішення поставленого завдання, програмування по готовому алгоритму, налагодження програми і підготовку документації.

ВИСНОВКИ

У ході виконання роботи було вивчено основи структур даних. Детально розглянута структура даних червоно-чорне дерево. На основі здобутих знань створено теоретичну довідку зі структур даних, яка стане в пригоді людям, які вивчають структури даних.

Для практичної реалізації червоно-чорного дерева було розроблено два класи: клас вершини дерева і клас самого дерева. Поглиблено навички в об’єктно-орієнтованому програмуванні. Створена реалізація буде використовуватися при виконанні проєктів, в яких передбачена робота з великим об’ємом даних.

Порівняно ефективність червоно-чорного дерева зі звичайним масивом.

Для підкреслення навчальної складової роботи, було написано кількадесят питань. З них випадково формується тест з п’яти питань. Цей тест користувач може пройти і отримати оцінку, тим самим закріпивши знання.

Також користувач має можливість подивитися на код реалізації червоно-чорних дерев, тим самим покращити власні навички ООП та використання структур даних на практиці.

Усі поставлені в роботі задачі було виконано.

Програму створено в середовищі програмування Microsoft Visual Studio 2019 мовою C#. Для пошуку інформації використано Google Chrome.

Під час написання програми були використані методи об’єктно-орієнтованого підходу.

Також у кваліфікаційній роботі було визначено трудомісткість розробленого програмного продукту 802 люд-год, проведений підрахунок вартості роботи по створенню програми 52855грн. та розраховано час на його створення 4,5 міс.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Robert Sedgewick. Algorithms (4th Edition). — Addison-Wesley Professional, 2011. — Chapter 3.

2. Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. Introduction to Algorithms (3rd Edition). — The MIT Press, 2009.

3. https://www.geeksforgeeks.org/ (цикл статей й ілюстрацій про червоно-чорне дерево)

4. https://en.wikipedia.org/wiki/Red%E2%80%93black_tree

5. https://docs.microsoft.com/en-us/dotnet/csharp/

6. Иванов Б.Н. Дискретная математика. Алгоритмы и программы. Полный курс. − М.: Физматлит, 2007. – 408 с.

7. Кривий С.Л. Курс дискретної математики. Навчальний посібник.– К:Наукова думка,2007. – 432 с.

8. Кушнерьов О. Про деякі застосування теорії графів [Текст] / О. Кушнерьов // Фізико-математична освіта : збірник наукових праць. – Суми : Вид-во фізико-математичного факультету СумДПУ імені А.С. Макаренка, 2015. – № 1 (7). – С. 50–56.

9. Матвієнко М. П. Дискретна математика Навч. посібник.– К.: «ВидавництвоЛіра-К», 2013. – 324 с.

10. Нікольський Ю. В., Пасічник В., Щербина Ю. М. Дискретна математика: Підручник. – Львів: «Магнолія – 2006», 2009. – 432 с.

11. Новиков Ф.А. Дискретная математика для программистов: Учебник для вузов. − 2-е изд. − СПб.: Питер, 2007. − 364 с.

12. Трохимчук Р.М. Теорія графів. Навчальний посібник для студентів факультету кібернетики / Трохимчук Р.М. К.: РВЦ «Київський університет», 1998. – 43 с.

13. Агуров Павел. C#. Сборник рецептов / Павел Агуров. - М.: "БХВ-Петербург", 2012. - 432 с.

14. Албахари Джозеф. C# 3.0. Справочник / Джозеф Албахари , Бен Албахари. - М.: БХВ-Петербург, 2013. - 944 с.

15. Альфред В. Ахо. Компиляторы. Принципы, технологии и инструментарий / Альфред В. Ахо и др. - М.: Вильямс, 2015. - 266 с.

16. Бишоп Дж. C# в кратком изложении / Дж. Бишоп, Н. Хорспул. - М.: Бином. Лаборатория знаний, 2013. - 472 с.

17. Вагнер Билл. C# Эффективное программирование / Билл Вагнер. - М.: ЛОРИ, 2013. - 320 с.

18. Зиборов Виктор. Visual C# 2010 на примерах / Виктор Зиборов. - М.: "БХВ-Петербург", 2011. - 432 с.

19. Ишкова Э. А. Самоучитель C#. Начала программирования / Э.А. Ишкова. - М.: Наука и техника, 2013. - 496 с.

20. Лотка Рокфорд. C# и CSLA .NET Framework. Разработка бизнес-объектов / Рокфорд Лотка. - М.: Вильямс, 2010. - 816 с.

21. Мак-Дональд Мэтью. Silverlight 5 с примерами на C# для профессионалов / Мэтью Мак-Дональд. - М.: Вильямс, 2013. - 848 с.

22. Подбельский В. В. Язык C#. Базовый курс / В.В. Подбельский. - М.: Финансы и статистика, Инфра-М, 2011. - 384 с.

23. Рихтер Джеффри. CLR via C#. Программирование на платформе Microsoft .NET Framework 4.0 на языке C# / Рихтер. - М.: Питер, 2013. - 928 с.

24. Троелсен Эндрю. Язык программирования C# 5.0 и платформа .NET 4.5 / Эндрю Троелсен. - М.: Вильямс, 2015. - 486 с.

25. Методичні вказівки з виконання економічного розділу в дипломних проектах студентів спеціальності «Комп'ютерні системи» / О.Г. Вагонова, О.Б. Нікітіна, Н.Н. Романюк; М-во освіти і науки України, ДВНЗ «Нац. гірн. ун-т». – Д.: НГУ, 2013. – 11 с.

26. Методичні рекомендації до виконання кваліфікаційних робіт бакалаврів напряму підготовки 6.050101 «Комп'ютерні науки / І.М. Удовик, Л.М. Коротенко, О.С. Шевцова. Нац. гірн. ун-т. – Д : НТУ «Дніпровська політехніка» . - 2018. – 65 с.

ДОДАТОК А

КОД ПРОГРАМИ

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace RedBlackTrees
{
   static class Program
   {
      /// <summary>
      /// The main entry point for the application.
      /// </summary>
      [STAThread]
      static void Main()
      {
         Application.EnableVisualStyles();
         Application.SetCompatibleTextRenderingDefault(false);
         Application.Run(new Title());
      }
   }
}


using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace RedBlackTrees
{
   public class RBNode
   {
      public byte color; // 0 - red, 1 - black      p.s. виявилось, що bool займає один байт, а не один біт, тому я вирішив використовувати числове значення
      public IComparable data; //щоб можна було задавати значення null
      public RBNode left;
      public RBNode right;
      public RBNode parent;
      public RBNode()
```

```
        {
            this.color = 0;
            this.left = null;
            this.right = null;
        }
        public RBNode(IComparable newData)
        {
            this.data = newData;
            this.left = null;
            this.right = null;
        }
        public RBNode(IComparable newData, RBNode Left, RBNode Right)
        {
            this.data = newData;
            this.left = Left;
            this.right = Right;
        }
    }
}

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace RedBlackTrees
{
    class RBTree
    {
        private RBNode freshNode = new RBNode(null); //leaf NIL node
        public RBNode root;
        public bool isFixing;
        public int maxDepth = 0;
        public int nodesCount = 0;
        public RBTree()
        {
            freshNode.color = 1;
            root = freshNode;
            isFixing = true;
        }
        public RBTree(bool b)
        {
            freshNode.color = 1;
            root = freshNode;
```

```
        isFixing = b;
    }

    public void Insert(IComparable input)
    {
        nodesCount++; //загальна кількість елементів у дереві збільшується на 1
        RBNode currentNode = root; //початок пошуку місця новій вершині починається з кореневої вершини
        RBNode newNode = new RBNode(input, freshNode, freshNode); //створення НОВОГО вказівника і задання йому інформації і посилань на дочірні вказівники. Фарбуємо у червоний
        newNode.color = 0;
        int currentDepth = 1;

        while (currentNode != freshNode) //рекурсивно шукаємо вершину батька для нової вершини. Вершина-батько має бути такою, щоб її права дитина за значенням була більша, а ліва – менша.
        {
            newNode.parent = currentNode;
            //якщо поточна вершина (currentNode), яка є кандидатом на вершину-батька для новоприбулої
            //вершини, за значенням більша, за нову (нова має значення input), то ми переходимо на рівень
            //нижче, поточною вершиною стає ліва (менша) дитина поточної вершини.
            if (input.CompareTo(currentNode.data) < 0)
            {
                currentNode = currentNode.left;
            }
            else
            {
                currentNode = currentNode.right;
            }
            currentDepth++;
        }
        //до цього моменту, ми вже встановили вершину, яка стане батьком нової вершини (newNode стає
        //на місце currentNode, яка вже дорівнює freshNode, тобто є листям, елементом без значення)

        //якщо дерево ще не містить елементів, тобто корінь = лист, новий елемент стає коренем.
        if (root == freshNode)
        {
            root = newNode;
```

```
        }
        else //якщо ні, то налаштовуємо посилання з батька нової вершини на нову вершину.
        {
            if (input.CompareTo(newNode.parent.data) < 0)   //новий елемент став на місце листя (тобто на місце дитини останнього елемента, який не був null)
            {
                newNode.parent.left = newNode;
            }
            else
            {
                newNode.parent.right = newNode;
            }
        }

        if(currentDepth > maxDepth)
        {
            maxDepth = currentDepth;
        }
        //вставка, аналогічна вставці у звичайне двійкове дерево пошуку, закнічена. Наступний рядок викликає метод збереження RB-властивостей, якщо вибраний режим червоно-чорного дерева, а не звичайного двійкового
        if (isFixing == true)
        {
            InsertFix(newNode);             //викликаємо метод для збереження RB-властивостей, якщо не балансування не вимкнене.
        }
    }

    private void InsertFix(RBNode x)
    {
        //якщо новий вказівник - коренева вершина, фарбуємо у чорний, бо це RB-властивість
        if (x == root)
        {
            x.color = 1;
        }
        else if (x.parent.parent != null && x.parent.color == 0)          //перевірка, чи має нова вершина батька і дідуся, щоб не видавало Unhandled Exception
        {
            //за RB-властивостями, червоний батько не може мати червоних дітей
            if (x.parent.parent.left.color == 0 && x.parent.parent.right.color == 0) //перший випадок: дядько і батько нового елементу червоні
            {
```

```
            x.parent.parent.left.color = 1;                    //змінюємо колір
батька і дядька на чорний, дідуся на червоний
            x.parent.parent.right.color = 1;
            x.parent.parent.color = 0;
            x = x.parent.parent;
            InsertFix(x); //рекурсивно повторюємо, щоб не було двох червоних
вершин поспіль
         }
         else
         {
            //другий випадок: червоний батько, чорний або відсутній дядько
(чотири підвипадки)
            //i) Left Left Case(p is left child of g and x is left child of p)
            //ii) Left Right Case(p is left child of g and x is right child of p)
            //iii) Right Right Case(Mirror of case i)
            //iv) Right Left Case(Mirror of case ii)
            if (x.parent == x.parent.parent.left)
            {
               if (x == x.parent.left)
               {
                  x.parent.color = 1;
                  x.parent.parent.color = 0;
                  Rotate(x.parent, Direction.right);
               }
               else
               {
                  x.color = 1;
                  x.parent.parent.color = 0;
                  Rotate(x, Direction.left);
                  Rotate(x, Direction.right);
               }
            }
            else
            {
               if (x == x.parent.right)
               {
                  x.parent.color = 1;
                  x.parent.parent.color = 0;
                  Rotate(x.parent, Direction.left);
               }
               else
               {
                  x.color = 1;
                  x.parent.parent.color = 0;
                  Rotate(x, Direction.right);
```

```
                    Rotate(x, Direction.left);
                }
            }
        }
    }
    root.color = 1; //чорна коренева вершина - RB-властивість
}

enum Direction
{
    right,
    left
}
private void Rotate(RBNode pivot, Direction dr)                    //поворот
бінарного дерева
{
    RBNode rotRoot = pivot.parent;
    if (dr == Direction.right)
    {
        rotRoot.left = pivot.right;
        pivot.right.parent = rotRoot;
        pivot.right = rotRoot;
        pivot.parent = rotRoot.parent;
        rotRoot.parent = pivot;

        if (rotRoot == root)
        {
            root = pivot;
        }
        else if (pivot.parent.right == rotRoot)
        {
            pivot.parent.right = pivot;
        }
        else
        {
            pivot.parent.left = pivot;
        }
    }
    else
    {
        rotRoot.right = pivot.left;
        pivot.left.parent = rotRoot;
        pivot.left = rotRoot;
        pivot.parent = rotRoot.parent;
        rotRoot.parent = pivot;
```

```
        if (rotRoot == root)
        {
            root = pivot;
        }
        else if (pivot.parent.right == rotRoot)          //оновлюємо посилання
        {
            pivot.parent.right = pivot;
        }
        else
        {
            pivot.parent.left = pivot;
        }
    }
}


public RBNode Search(IComparable nodeData)
{
    RBNode currentNode = root;
    while (currentNode.data.CompareTo(nodeData) != 0)
    {
        if (currentNode == freshNode)
        {
            break;
        }
        else
        {
            if (nodeData.CompareTo(currentNode.data) < 0)
            {
                currentNode = currentNode.left;
            }
            else
            {
                currentNode = currentNode.right;
            }
        }
    }

    if (currentNode == freshNode)
    {
        return freshNode;
        //ЕЛЕМЕНТ НЕ НАЛЕЖИТЬ ДЕРЕВУ
    }
    else
```

```
        {
            return currentNode;
        }
    }

    public void Delete(IComparable nodeData)
    {
        RBNode deleteNode = this.Search(nodeData);
        if (deleteNode == freshNode || deleteNode == root)
        {
            throw new Exception("Вершина не належить дереву або є кореневою,
тому не може бути видалена.");
        }
        else if (deleteNode.left == freshNode && deleteNode.right == freshNode)
//обраний елемент немає дітей
        {
            if (deleteNode.data.CompareTo(deleteNode.parent.data) < 0)
            {

                deleteNode.parent.left = freshNode;
            }
            else
            {

                deleteNode.parent.right = freshNode;
            }
            deleteNode = null;
        }
        else if (deleteNode.left != freshNode && deleteNode.right == freshNode ||
deleteNode.left == freshNode && deleteNode.right != freshNode) //обраний
елемент має лише одну дитину
        {
            if (deleteNode.data.CompareTo(deleteNode.parent.data) < 0)
            {
                if (deleteNode.right != freshNode)
                {
                    deleteNode.parent.left = deleteNode.right;
                    deleteNode.right.parent = deleteNode.parent;
                }
                else
                {
                    deleteNode.parent.left = deleteNode.left;
                    deleteNode.left.parent = deleteNode.parent;
                }
            }
```

```
            else
            {
               if (deleteNode.right != freshNode)
               {
                  deleteNode.parent.right = deleteNode.right;
                  deleteNode.right.parent = deleteNode.parent;
               }
               else
               {
                  deleteNode.parent.right = deleteNode.left;
                  deleteNode.left.parent = deleteNode.parent;
               }
            }
            if (isFixing == true)
            {
               DeleteFix(deleteNode); //якщо вершина немає дітей, її видаленням
неможливо порушити РБ-властивості, якщо має дві дитини, то її видалення
зводиться до видалення вершини з однією дитиною
            }
            deleteNode = null;
         }
         else //вершина має двох дітей
         {
            RBNode currentNode = deleteNode.right;
            while (currentNode.left != freshNode)          //шукаємо найменшу
вершину правого піддерева (правило видалення елементів з бінарних дерев
пошуку) (in-order predecessor node)
            {
               currentNode = currentNode.left;
            }
            IComparable t = currentNode.data;
            this.Delete(currentNode.data);               //заміщуємо видалену
інформацію
            deleteNode.data = t;
         }
      }


      private void DeleteFix(RBNode v)
      {
         RBNode u; //дитина видаленої вершини
         RBNode s; //sibling of u
         RBNode r; //red child of s
         if (v.left != freshNode)
         {
```

```
    u = v.left;
}
else
{
    u = v.right;
}

if (u == u.parent.left)
{
    s = u.parent.right;
}
else
{
    s = u.parent.left;
}

if (s.right.color == 0 && s.left.color == 0)
{
    r = s;
}
else if (s.left.color == 0)
{
    r = s.left;
}
else if (s.right.color == 0)
{
    r = s.right;
}
else
{
    r = null;
}


if (u.color == 0 || v.color == 0)
{
    u.color = 1;
}
else if (u.color == v.color && v.color == 1)
{
    u.color = 2; //double black color code
}

if (u.color == 2 && u != root)
{
```

```
        if (s.color == 1 && (s.left.color == 0 || s.right.color == 0))  //Чотири випадки - Left Left, Left right, Right Right, Right Left
        {
            if (s == s.parent.left)
            {
                if (r == s.left || r == s)
                {
                    s.left.color = 1;
                    Rotate(s, Direction.right);
                }
                else if (r == s.right)
                {
                    Rotate(r, Direction.left);
                    Rotate(r, Direction.right);
                }
            }
            else
            {
                if (r == s.left || r == s)
                {
                    s.color = 0;
                    r.color = 1;
                    Rotate(r, Direction.right);
                    Rotate(r, Direction.left);
                }
                else if (r == s.right)
                {
                    s.right.color = 1;
                    Rotate(s, Direction.left);
                }
            }
        }
        else if (s.color == 1 && s.left.color == 1 && s.right.color == 1)
        {
            s.color = 0;
            if (s.parent.color == 0)
            {
                s.parent.color = 1;
            }
            else
            {
                s.parent.color = 2;
                DeleteFix(s.parent);
            }
        }
```

```
            else if (s.color == 0)
            {
               if (s == s.parent.left)
               {
                  s.parent.color = 0;
                  s.color = 1;
                  Rotate(s, Direction.left);
               }
               else
               {
                  s.parent.color = 0;
                  s.color = 1;
                  Rotate(s, Direction.right);
               }
            }
         }
         else if (u == root)
         {
            u.color = 1;
         }
      }
      public void Display(System.Windows.Forms.PictureBox pb)
      {
         this.Display(root);
      }
      private void Display(RBNode temp)
      {
         if(temp != freshNode)
         {
            Display(temp.left);
            Display(temp);
            Display(temp.right);
         }
      }
   }
}

namespace RedBlackTrees
{
   partial class Quiz
   {
      /// <summary>
      /// Required designer variable.
      /// </summary>
      private System.ComponentModel.IContainer components = null;
```

```
        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            this.introLabel = new System.Windows.Forms.Label();
            this.start = new System.Windows.Forms.Button();
            this.questionLabel = new System.Windows.Forms.Label();
            this.qNum = new System.Windows.Forms.Label();
            this.pictureBox1 = new System.Windows.Forms.PictureBox();
            this.aLabel = new System.Windows.Forms.Label();
            this.bLabel = new System.Windows.Forms.Label();
            this.cLabel = new System.Windows.Forms.Label();
            this.dLabel = new System.Windows.Forms.Label();
            this.aRadio = new System.Windows.Forms.RadioButton();
            this.bRadio = new System.Windows.Forms.RadioButton();
            this.cRadio = new System.Windows.Forms.RadioButton();
            this.dRadio = new System.Windows.Forms.RadioButton();
            this.answerButton = new System.Windows.Forms.Button();
            this.total = new System.Windows.Forms.Label();
            this.timer1 = new System.Windows.Forms.Timer(this.components);
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();
            this.SuspendLayout();
            //
            // introLabel
            //
```

```
            this.introLabel.Font = new System.Drawing.Font("Arial", 24F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(204)));
            this.introLabel.Location = new System.Drawing.Point(145, 164);
            this.introLabel.Name = "introLabel";
            this.introLabel.Size = new System.Drawing.Size(710, 77);
            this.introLabel.TabIndex = 0;
            this.introLabel.Text = "Вам пропонується дати відповідь на 5 питань за
структурою даних \"Червоно-чорне де" +
               "рево\"";
            this.introLabel.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // start
            //
            this.start.Font = new System.Drawing.Font("Microsoft Sans Serif", 22F);
            this.start.Location = new System.Drawing.Point(380, 357);
            this.start.Name = "start";
            this.start.Size = new System.Drawing.Size(240, 83);
            this.start.TabIndex = 1;
            this.start.Text = "Почати";
            this.start.UseVisualStyleBackColor = true;
            this.start.Click += new System.EventHandler(this.button1_Click);
            this.start.MouseEnter += new
System.EventHandler(this.button1_MouseEnter);
            this.start.MouseLeave += new
System.EventHandler(this.button1_MouseLeave);
            //
            // questionLabel
            //
            this.questionLabel.AutoSize = true;
            this.questionLabel.Font = new System.Drawing.Font("Microsoft Sans Serif",
18F);
            this.questionLabel.Location = new System.Drawing.Point(150, 75);
            this.questionLabel.MaximumSize = new System.Drawing.Size(700, 1000);
            this.questionLabel.MinimumSize = new System.Drawing.Size(700, 50);
            this.questionLabel.Name = "questionLabel";
            this.questionLabel.Size = new System.Drawing.Size(700, 58);
            this.questionLabel.TabIndex = 2;
            this.questionLabel.Text = "Яка часова складність операції пошуку для
структури даних червоно-чорного дерева " +
               "у нотації великого O?";
            this.questionLabel.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            this.questionLabel.Visible = false;
```

```
            //
            // qNum
            //
            this.qNum.Font = new System.Drawing.Font("Arial", 24F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point,
((byte)(204)));
            this.qNum.Location = new System.Drawing.Point(380, 16);
            this.qNum.Name = "qNum";
            this.qNum.Size = new System.Drawing.Size(240, 37);
            this.qNum.TabIndex = 0;
            this.qNum.Text = "Питання №";
            this.qNum.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            this.qNum.Visible = false;
            //
            // pictureBox1
            //
            this.pictureBox1.Location = new System.Drawing.Point(12, 361);
            this.pictureBox1.Name = "pictureBox1";
            this.pictureBox1.Size = new System.Drawing.Size(960, 186);
            this.pictureBox1.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox1.TabIndex = 3;
            this.pictureBox1.TabStop = false;
            this.pictureBox1.Visible = false;
            //
            // aLabel
            //
            this.aLabel.AutoSize = true;
            this.aLabel.Font = new System.Drawing.Font("Microsoft Sans Serif", 18F);
            this.aLabel.Location = new System.Drawing.Point(140, 451);
            this.aLabel.Name = "aLabel";
            this.aLabel.Size = new System.Drawing.Size(40, 29);
            this.aLabel.TabIndex = 4;
            this.aLabel.Text = "A: ";
            this.aLabel.Visible = false;
            this.aLabel.Click += new System.EventHandler(this.aLabel_Click);
            //
            // bLabel
            //
            this.bLabel.AutoSize = true;
            this.bLabel.Font = new System.Drawing.Font("Microsoft Sans Serif", 18F);
            this.bLabel.Location = new System.Drawing.Point(140, 480);
            this.bLabel.Name = "bLabel";
            this.bLabel.Size = new System.Drawing.Size(41, 29);
            this.bLabel.TabIndex = 4;
```

```
this.bLabel.Text = "B: ";
this.bLabel.Visible = false;
this.bLabel.Click += new System.EventHandler(this.bLabel_Click);
//
// cLabel
//
this.cLabel.AutoSize = true;
this.cLabel.Font = new System.Drawing.Font("Microsoft Sans Serif", 18F);
this.cLabel.Location = new System.Drawing.Point(140, 509);
this.cLabel.Name = "cLabel";
this.cLabel.Size = new System.Drawing.Size(42, 29);
this.cLabel.TabIndex = 4;
this.cLabel.Text = "C: ";
this.cLabel.Visible = false;
this.cLabel.Click += new System.EventHandler(this.cLabel_Click);
//
// dLabel
//
this.dLabel.AutoSize = true;
this.dLabel.Font = new System.Drawing.Font("Microsoft Sans Serif", 18F);
this.dLabel.Location = new System.Drawing.Point(140, 538);
this.dLabel.Name = "dLabel";
this.dLabel.Size = new System.Drawing.Size(42, 29);
this.dLabel.TabIndex = 4;
this.dLabel.Text = "D: ";
this.dLabel.Visible = false;
this.dLabel.Click += new System.EventHandler(this.dLabel_Click);
//
// aRadio
//
this.aRadio.AutoSize = true;
this.aRadio.Location = new System.Drawing.Point(127, 459);
this.aRadio.Name = "aRadio";
this.aRadio.Size = new System.Drawing.Size(14, 13);
this.aRadio.TabIndex = 5;
this.aRadio.TabStop = true;
this.aRadio.UseVisualStyleBackColor = true;
this.aRadio.Visible = false;
//
// bRadio
//
this.bRadio.AutoSize = true;
this.bRadio.Location = new System.Drawing.Point(127, 488);
this.bRadio.Name = "bRadio";
this.bRadio.Size = new System.Drawing.Size(14, 13);
```

```
        this.bRadio.TabIndex = 5;
        this.bRadio.TabStop = true;
        this.bRadio.UseVisualStyleBackColor = true;
        this.bRadio.Visible = false;
        //
        // cRadio
        //
        this.cRadio.AutoSize = true;
        this.cRadio.Location = new System.Drawing.Point(127, 517);
        this.cRadio.Name = "cRadio";
        this.cRadio.Size = new System.Drawing.Size(14, 13);
        this.cRadio.TabIndex = 5;
        this.cRadio.TabStop = true;
        this.cRadio.UseVisualStyleBackColor = true;
        this.cRadio.Visible = false;
        //
        // dRadio
        //
        this.dRadio.AutoSize = true;
        this.dRadio.Location = new System.Drawing.Point(127, 546);
        this.dRadio.Name = "dRadio";
        this.dRadio.Size = new System.Drawing.Size(14, 13);
        this.dRadio.TabIndex = 5;
        this.dRadio.TabStop = true;
        this.dRadio.UseVisualStyleBackColor = true;
        this.dRadio.Visible = false;
        //
        // answerButton
        //
        this.answerButton.Font = new System.Drawing.Font("Microsoft Sans Serif",
20F);
        this.answerButton.Location = new System.Drawing.Point(415, 568);
        this.answerButton.Name = "answerButton";
        this.answerButton.Size = new System.Drawing.Size(170, 48);
        this.answerButton.TabIndex = 6;
        this.answerButton.Text = "Відповісти";
        this.answerButton.UseVisualStyleBackColor = true;
        this.answerButton.Visible = false;
        this.answerButton.Click += new
System.EventHandler(this.answerButton_Click);
        //
        // total
        //
        this.total.AutoSize = true;
        this.total.Font = new System.Drawing.Font("Microsoft Sans Serif", 20F);
```

```
    this.total.Location = new System.Drawing.Point(435, 621);
    this.total.Name = "total";
    this.total.Size = new System.Drawing.Size(131, 31);
    this.total.TabIndex = 7;
    this.total.Text = "Оцінка: 0";
    this.total.Visible = false;
    //
    // timer1
    //
    this.timer1.Interval = 1000;
    this.timer1.Tick += new System.EventHandler(this.timer1_Tick);
    //
    // Quiz
    //
    this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
    this.ClientSize = new System.Drawing.Size(984, 661);
    this.Controls.Add(this.total);
    this.Controls.Add(this.answerButton);
    this.Controls.Add(this.dRadio);
    this.Controls.Add(this.cRadio);
    this.Controls.Add(this.bRadio);
    this.Controls.Add(this.aRadio);
    this.Controls.Add(this.dLabel);
    this.Controls.Add(this.cLabel);
    this.Controls.Add(this.bLabel);
    this.Controls.Add(this.aLabel);
    this.Controls.Add(this.pictureBox1);
    this.Controls.Add(this.questionLabel);
    this.Controls.Add(this.start);
    this.Controls.Add(this.qNum);
    this.Controls.Add(this.introLabel);
    this.MaximizeBox = false;
    this.MinimizeBox = false;
    this.Name = "Quiz";
    this.ShowIcon = false;
    this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
    this.Text = "Перевір себе";
    ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).EndInit();
    this.ResumeLayout(false);
    this.PerformLayout();

}

#endregion
```

```
        private System.Windows.Forms.Label introLabel;
        private System.Windows.Forms.Button start;
        private System.Windows.Forms.Label questionLabel;
        private System.Windows.Forms.Label qNum;
        private System.Windows.Forms.PictureBox pictureBox1;
        private System.Windows.Forms.Label aLabel;
        private System.Windows.Forms.Label bLabel;
        private System.Windows.Forms.Label cLabel;
        private System.Windows.Forms.Label dLabel;
        private System.Windows.Forms.RadioButton aRadio;
        private System.Windows.Forms.RadioButton bRadio;
        private System.Windows.Forms.RadioButton cRadio;
        private System.Windows.Forms.RadioButton dRadio;
        private System.Windows.Forms.Button answerButton;
        private System.Windows.Forms.Label total;
        private System.Windows.Forms.Timer timer1;
    }
}

namespace RedBlackTrees_Mutel
{
    partial class Main
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code
```

```
        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.button1 = new System.Windows.Forms.Button();
            this.button2 = new System.Windows.Forms.Button();
            this.button3 = new System.Windows.Forms.Button();
            this.button4 = new System.Windows.Forms.Button();
            this.SuspendLayout();
            //
            // button1
            //
            this.button1.BackColor = System.Drawing.Color.OrangeRed;
            this.button1.Font = new System.Drawing.Font("Microsoft Sans Serif", 18F);
            this.button1.Location = new System.Drawing.Point(520, 370);
            this.button1.Name = "button1";
            this.button1.Size = new System.Drawing.Size(200, 200);
            this.button1.TabIndex = 0;
            this.button1.Text = "Перевір себе";
            this.button1.UseVisualStyleBackColor = false;
            this.button1.Click += new System.EventHandler(this.button1_Click);
            this.button1.MouseEnter += new
System.EventHandler(this.button1_MouseEnter);
            this.button1.MouseLeave += new
System.EventHandler(this.button1_MouseLeave);
            //
            // button2
            //
            this.button2.BackColor = System.Drawing.Color.OrangeRed;
            this.button2.Font = new System.Drawing.Font("Microsoft Sans Serif", 18F);
            this.button2.Location = new System.Drawing.Point(280, 370);
            this.button2.Name = "button2";
            this.button2.Size = new System.Drawing.Size(200, 200);
            this.button2.TabIndex = 0;
            this.button2.Text = "Моделювання";
            this.button2.UseVisualStyleBackColor = false;
            this.button2.Click += new System.EventHandler(this.button2_Click);
            this.button2.MouseEnter += new
System.EventHandler(this.button2_MouseEnter);
            this.button2.MouseLeave += new
System.EventHandler(this.button2_MouseLeave);
            //
            // button3
```

```
            //
            this.button3.BackColor = System.Drawing.Color.OrangeRed;
            this.button3.Font = new System.Drawing.Font("Microsoft Sans Serif", 18F);
            this.button3.Location = new System.Drawing.Point(280, 130);
            this.button3.Name = "button3";
            this.button3.Size = new System.Drawing.Size(200, 200);
            this.button3.TabIndex = 0;
            this.button3.Text = "Теорія і код";
            this.button3.UseVisualStyleBackColor = false;
            this.button3.Click += new System.EventHandler(this.button3_Click);
            this.button3.MouseEnter += new
System.EventHandler(this.button3_MouseEnter);
            this.button3.MouseLeave += new
System.EventHandler(this.button3_MouseLeave);
            //
            // button4
            //
            this.button4.BackColor = System.Drawing.Color.OrangeRed;
            this.button4.Font = new System.Drawing.Font("Microsoft Sans Serif", 18F);
            this.button4.Location = new System.Drawing.Point(520, 130);
            this.button4.Name = "button4";
            this.button4.Size = new System.Drawing.Size(200, 200);
            this.button4.TabIndex = 0;
            this.button4.Text = "Інструкція до моделювання";
            this.button4.UseVisualStyleBackColor = false;
            this.button4.Click += new System.EventHandler(this.button4_Click);
            this.button4.MouseEnter += new
System.EventHandler(this.button4_MouseEnter);
            this.button4.MouseLeave += new
System.EventHandler(this.button4_MouseLeave);
            //
            // Main
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.BackColor = System.Drawing.Color.Black;
            this.ClientSize = new System.Drawing.Size(984, 661);
            this.Controls.Add(this.button2);
            this.Controls.Add(this.button4);
            this.Controls.Add(this.button3);
            this.Controls.Add(this.button1);
            this.MaximizeBox = false;
            this.MaximumSize = new System.Drawing.Size(1000, 700);
            this.MinimizeBox = false;
            this.MinimumSize = new System.Drawing.Size(1000, 700);
```

```
            this.Name = "Main";
            this.ShowIcon = false;
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Головна";
            this.ResumeLayout(false);

        }

        #endregion

        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Button button3;
        private System.Windows.Forms.Button button4;
    }
}
namespace RedBlackTrees
{
    partial class Modelling
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
```

```
        private void InitializeComponent()
        {
            this.menuStrip1 = new System.Windows.Forms.MenuStrip();
            this.toolStripTextBox1 = new System.Windows.Forms.ToolStripTextBox();
            this.вставитиToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.toolStripTextBox2 = new System.Windows.Forms.ToolStripTextBox();
            this.видалитизаЗначеннямToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.toolStripTextBox3 = new System.Windows.Forms.ToolStripTextBox();
            this.шукатизаЗначеннямToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.очиститиToolStripMenuItem = new
System.Windows.Forms.ToolStripMenuItem();
            this.pictureBox1 = new System.Windows.Forms.PictureBox();
            this.listBox1 = new System.Windows.Forms.ListBox();
            this.label1 = new System.Windows.Forms.Label();
            this.button1 = new System.Windows.Forms.Button();
            this.numericUpDown1 = new System.Windows.Forms.NumericUpDown();
            this.label2 = new System.Windows.Forms.Label();
            this.textBox1 = new System.Windows.Forms.TextBox();
            this.label3 = new System.Windows.Forms.Label();
            this.checkBox1 = new System.Windows.Forms.CheckBox();
            this.textBox2 = new System.Windows.Forms.TextBox();
            this.textBox3 = new System.Windows.Forms.TextBox();
            this.label4 = new System.Windows.Forms.Label();
            this.label5 = new System.Windows.Forms.Label();
            this.checkBox2 = new System.Windows.Forms.CheckBox();
            this.menuStrip1.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.numericUpDown1)).BeginInit();
            this.SuspendLayout();
            //
            // menuStrip1
            //
            this.menuStrip1.Items.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.toolStripTextBox1,
            this.вставитиToolStripMenuItem,
            this.toolStripTextBox2,
            this.видалитизаЗначеннямToolStripMenuItem,
            this.toolStripTextBox3,
            this.шукатизаЗначеннямToolStripMenuItem,
            this.очиститиToolStripMenuItem});
```

```
            this.menuStrip1.Location = new System.Drawing.Point(0, 0);
            this.menuStrip1.Name = "menuStrip1";
            this.menuStrip1.Padding = new System.Windows.Forms.Padding(6, 4, 0, 4);
            this.menuStrip1.Size = new System.Drawing.Size(984, 35);
            this.menuStrip1.TabIndex = 0;
            this.menuStrip1.Text = "menuStrip1";
            //
            // toolStripTextBox1
            //
            this.toolStripTextBox1.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
            this.toolStripTextBox1.Name = "toolStripTextBox1";
            this.toolStripTextBox1.Size = new System.Drawing.Size(100, 27);
            this.toolStripTextBox1.TextChanged += new
System.EventHandler(this.toolStripTextBox1_TextChanged);
            //
            // вставитиToolStripMenuItem
            //
            this.вставитиToolStripMenuItem.BackColor =
System.Drawing.SystemColors.ControlDark;
            this.вставитиToolStripMenuItem.Margin = new
System.Windows.Forms.Padding(0, 0, 20, 0);
            this.вставитиToolStripMenuItem.Name = "вставитиToolStripMenuItem";
            this.вставитиToolStripMenuItem.Size = new System.Drawing.Size(77, 27);
            this.вставитиToolStripMenuItem.Text = "Вставити";
            this.вставитиToolStripMenuItem.Click += new
System.EventHandler(this.вставитиToolStripMenuItem_Click);
            //
            // toolStripTextBox2
            //
            this.toolStripTextBox2.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
            this.toolStripTextBox2.Enabled = false;
            this.toolStripTextBox2.Name = "toolStripTextBox2";
            this.toolStripTextBox2.Size = new System.Drawing.Size(100, 27);
            this.toolStripTextBox2.TextChanged += new
System.EventHandler(this.toolStripTextBox2_TextChanged);
            //
            // видалитизаЗначеннямToolStripMenuItem
            //
            this.видалитизаЗначеннямToolStripMenuItem.BackColor =
System.Drawing.SystemColors.ControlDark;
            this.видалитизаЗначеннямToolStripMenuItem.Enabled = false;
            this.видалитизаЗначеннямToolStripMenuItem.Margin = new
System.Windows.Forms.Padding(0, 0, 20, 0);
```

```
            this.видалитизаЗначеннямToolStripMenuItem.Name =
"видалитизаЗначеннямToolStripMenuItem";
            this.видалитизаЗначеннямToolStripMenuItem.Size = new
System.Drawing.Size(179, 27);
            this.видалитизаЗначеннямToolStripMenuItem.Text = "Видалити (за
значенням)";
            this.видалитизаЗначеннямToolStripMenuItem.Click += new
System.EventHandler(this.видалитизаЗначеннямToolStripMenuItem_Click);
            //
            // toolStripTextBox3
            //
            this.toolStripTextBox3.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
            this.toolStripTextBox3.Enabled = false;
            this.toolStripTextBox3.Name = "toolStripTextBox3";
            this.toolStripTextBox3.Size = new System.Drawing.Size(100, 27);
            this.toolStripTextBox3.TextChanged += new
System.EventHandler(this.toolStripTextBox3_TextChanged);
            //
            // шукатизаЗначеннямToolStripMenuItem
            //
            this.шукатизаЗначеннямToolStripMenuItem.BackColor =
System.Drawing.SystemColors.ControlDark;
            this.шукатизаЗначеннямToolStripMenuItem.Enabled = false;
            this.шукатизаЗначеннямToolStripMenuItem.Name =
"шукатизаЗначеннямToolStripMenuItem";
            this.шукатизаЗначеннямToolStripMenuItem.Size = new
System.Drawing.Size(167, 27);
            this.шукатизаЗначеннямToolStripMenuItem.Text = "Шукати (за
значенням)";
            this.шукатизаЗначеннямToolStripMenuItem.Click += new
System.EventHandler(this.шукатизаЗначеннямToolStripMenuItem_Click);
            //
            // очиститиToolStripMenuItem
            //
            this.очиститиToolStripMenuItem.BackColor =
System.Drawing.SystemColors.ControlDark;
            this.очиститиToolStripMenuItem.Enabled = false;
            this.очиститиToolStripMenuItem.Margin = new
System.Windows.Forms.Padding(20, 0, 0, 0);
            this.очиститиToolStripMenuItem.Name = "очиститиToolStripMenuItem";
            this.очиститиToolStripMenuItem.Padding = new
System.Windows.Forms.Padding(4, 4, 4, 0);
            this.очиститиToolStripMenuItem.Size = new System.Drawing.Size(145, 27);
            this.очиститиToolStripMenuItem.Text = "Нове моделювання";
```

```
        this.очиститиToolStripMenuItem.Click += new
System.EventHandler(this.очиститиToolStripMenuItem_Click);
        //
        // pictureBox1
        //
        this.pictureBox1.BackColor = System.Drawing.Color.White;
        this.pictureBox1.BorderStyle =
System.Windows.Forms.BorderStyle.FixedSingle;
        this.pictureBox1.Location = new System.Drawing.Point(12, 45);
        this.pictureBox1.Name = "pictureBox1";
        this.pictureBox1.Size = new System.Drawing.Size(831, 604);
        this.pictureBox1.TabIndex = 1;
        this.pictureBox1.TabStop = false;
        //
        // listBox1
        //
        this.listBox1.FormattingEnabled = true;
        this.listBox1.Location = new System.Drawing.Point(852, 61);
        this.listBox1.Name = "listBox1";
        this.listBox1.Size = new System.Drawing.Size(120, 355);
        this.listBox1.TabIndex = 2;
        //
        // label1
        //
        this.label1.AutoSize = true;
        this.label1.Location = new System.Drawing.Point(849, 45);
        this.label1.Name = "label1";
        this.label1.Size = new System.Drawing.Size(96, 13);
        this.label1.TabIndex = 3;
        this.label1.Text = "Елементи дерева";
        //
        // button1
        //
        this.button1.BackColor = System.Drawing.SystemColors.ControlDark;
        this.button1.Location = new System.Drawing.Point(852, 422);
        this.button1.Name = "button1";
        this.button1.Size = new System.Drawing.Size(120, 72);
        this.button1.TabIndex = 4;
        this.button1.Text = "Вставити х випадкових";
        this.button1.UseVisualStyleBackColor = false;
        this.button1.Click += new System.EventHandler(this.button1_Click);
        //
        // numericUpDown1
        //
        this.numericUpDown1.Location = new System.Drawing.Point(849, 588);
```

```
            this.numericUpDown1.Name = "numericUpDown1";
            this.numericUpDown1.Size = new System.Drawing.Size(123, 20);
            this.numericUpDown1.TabIndex = 5;
            this.numericUpDown1.Value = new decimal(new int[] {
            13,
            0,
            0,
            0});
            this.numericUpDown1.ValueChanged += new
System.EventHandler(this.numericUpDown1_ValueChanged);
            //
            // label2
            //
            this.label2.AutoSize = true;
            this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F);
            this.label2.Location = new System.Drawing.Point(846, 568);
            this.label2.Name = "label2";
            this.label2.Size = new System.Drawing.Size(116, 17);
            this.label2.TabIndex = 6;
            this.label2.Text = "Радіус вершини:";
            //
            // textBox1
            //
            this.textBox1.Location = new System.Drawing.Point(880, 500);
            this.textBox1.Name = "textBox1";
            this.textBox1.Size = new System.Drawing.Size(92, 20);
            this.textBox1.TabIndex = 7;
            this.textBox1.Text = "30";
            this.textBox1.TextChanged += new
System.EventHandler(this.textBox1_TextChanged);
            //
            // label3
            //
            this.label3.AutoSize = true;
            this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F);
            this.label3.Location = new System.Drawing.Point(849, 501);
            this.label3.Name = "label3";
            this.label3.Size = new System.Drawing.Size(29, 17);
            this.label3.TabIndex = 6;
            this.label3.Text = "X =";
            //
            // checkBox1
            //
            this.checkBox1.AutoSize = true;
            this.checkBox1.Checked = true;
```

```
            this.checkBox1.CheckState = System.Windows.Forms.CheckState.Checked;
            this.checkBox1.Location = new System.Drawing.Point(849, 614);
            this.checkBox1.Name = "checkBox1";
            this.checkBox1.Size = new System.Drawing.Size(91, 17);
            this.checkBox1.TabIndex = 8;
            this.checkBox1.Text = "Балансувати";
            this.checkBox1.UseVisualStyleBackColor = true;
            this.checkBox1.CheckedChanged += new
System.EventHandler(this.checkBox1_CheckedChanged);
            //
            // textBox2
            //
            this.textBox2.Location = new System.Drawing.Point(849, 544);
            this.textBox2.Name = "textBox2";
            this.textBox2.Size = new System.Drawing.Size(43, 20);
            this.textBox2.TabIndex = 7;
            this.textBox2.Text = "0";
            this.textBox2.TextChanged += new
System.EventHandler(this.textBox2_TextChanged);
            //
            // textBox3
            //
            this.textBox3.Location = new System.Drawing.Point(917, 545);
            this.textBox3.Name = "textBox3";
            this.textBox3.Size = new System.Drawing.Size(49, 20);
            this.textBox3.TabIndex = 7;
            this.textBox3.Text = "1000";
            this.textBox3.TextChanged += new
System.EventHandler(this.textBox3_TextChanged);
            //
            // label4
            //
            this.label4.AutoSize = true;
            this.label4.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F);
            this.label4.Location = new System.Drawing.Point(849, 524);
            this.label4.Name = "label4";
            this.label4.Size = new System.Drawing.Size(123, 17);
            this.label4.TabIndex = 9;
            this.label4.Text = "Інтервал значень";
            //
            // label5
            //
            this.label5.AutoSize = true;
            this.label5.Font = new System.Drawing.Font("Microsoft Sans Serif", 10F);
            this.label5.Location = new System.Drawing.Point(898, 544);
```

```
        this.label5.Name = "label5";
        this.label5.Size = new System.Drawing.Size(13, 17);
        this.label5.TabIndex = 6;
        this.label5.Text = "-";
        //
        // checkBox2
        //
        this.checkBox2.AutoSize = true;
        this.checkBox2.Checked = true;
        this.checkBox2.CheckState = System.Windows.Forms.CheckState.Checked;
        this.checkBox2.Location = new System.Drawing.Point(849, 632);
        this.checkBox2.Name = "checkBox2";
        this.checkBox2.Size = new System.Drawing.Size(78, 17);
        this.checkBox2.TabIndex = 8;
        this.checkBox2.Text = "Малювати";
        this.checkBox2.UseVisualStyleBackColor = true;
        this.checkBox2.CheckedChanged += new
System.EventHandler(this.checkBox2_CheckedChanged);
        //
        // Modelling
        //
        this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
        this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
        this.ClientSize = new System.Drawing.Size(984, 661);
        this.Controls.Add(this.label4);
        this.Controls.Add(this.checkBox2);
        this.Controls.Add(this.checkBox1);
        this.Controls.Add(this.textBox3);
        this.Controls.Add(this.textBox2);
        this.Controls.Add(this.textBox1);
        this.Controls.Add(this.label5);
        this.Controls.Add(this.label3);
        this.Controls.Add(this.label2);
        this.Controls.Add(this.numericUpDown1);
        this.Controls.Add(this.button1);
        this.Controls.Add(this.label1);
        this.Controls.Add(this.listBox1);
        this.Controls.Add(this.menuStrip1);
        this.Controls.Add(this.pictureBox1);
        this.MainMenuStrip = this.menuStrip1;
        this.MaximizeBox = false;
        this.MaximumSize = new System.Drawing.Size(1400, 1200);
        this.MinimizeBox = false;
        this.MinimumSize = new System.Drawing.Size(800, 600);
        this.Name = "Modelling";
```

```
            this.ShowIcon = false;
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Моделювання";
            this.Load += new System.EventHandler(this.Modelling_Load);
            this.menuStrip1.ResumeLayout(false);
            this.menuStrip1.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pictureBox1)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.numericUpDown1)).EndInit();
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.MenuStrip menuStrip1;
        private System.Windows.Forms.ToolStripTextBox toolStripTextBox1;
        private System.Windows.Forms.ToolStripMenuItem
вставитиToolStripMenuItem;
        private System.Windows.Forms.ToolStripTextBox toolStripTextBox2;
        private System.Windows.Forms.ToolStripMenuItem
видалитизаЗначеннямToolStripMenuItem;
        private System.Windows.Forms.ToolStripTextBox toolStripTextBox3;
        private System.Windows.Forms.ToolStripMenuItem
шукатизаЗначеннямToolStripMenuItem;
        private System.Windows.Forms.ToolStripMenuItem
очиститиToolStripMenuItem;
        private System.Windows.Forms.PictureBox pictureBox1;
        private System.Windows.Forms.ListBox listBox1;
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.NumericUpDown numericUpDown1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.TextBox textBox1;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.CheckBox checkBox1;
        private System.Windows.Forms.TextBox textBox2;
        private System.Windows.Forms.TextBox textBox3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.CheckBox checkBox2;
    }
}
```

```
namespace RedBlackTrees
{
   partial class Theory
   {
      /// <summary>
      /// Required designer variable.
      /// </summary>
      private System.ComponentModel.IContainer components = null;

      /// <summary>
      /// Clean up any resources being used.
      /// </summary>
      /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
      protected override void Dispose(bool disposing)
      {
         if (disposing && (components != null))
         {
            components.Dispose();
         }
         base.Dispose(disposing);
      }

      #region Windows Form Designer generated code

      /// <summary>
      /// Required method for Designer support - do not modify
      /// the contents of this method with the code editor.
      /// </summary>
      private void InitializeComponent()
      {
         this.components = new System.ComponentModel.Container();
         System.Windows.Forms.TreeNode treeNode1 = new
System.Windows.Forms.TreeNode("Вступ до структур даних");
         System.Windows.Forms.TreeNode treeNode2 = new
System.Windows.Forms.TreeNode("Двійкове дерево пошуку");
         System.Windows.Forms.TreeNode treeNode3 = new
System.Windows.Forms.TreeNode("RB-властивості");
         System.Windows.Forms.TreeNode treeNode4 = new
System.Windows.Forms.TreeNode("Вставка");
         System.Windows.Forms.TreeNode treeNode5 = new
System.Windows.Forms.TreeNode("Видалення");
         System.Windows.Forms.TreeNode treeNode6 = new
System.Windows.Forms.TreeNode("Червоно-чорне дерево", new
System.Windows.Forms.TreeNode[] {
```

```
        treeNode3,
        treeNode4,
        treeNode5});
        System.Windows.Forms.TreeNode treeNode7 = new
System.Windows.Forms.TreeNode("Клас RBNode");
        System.Windows.Forms.TreeNode treeNode8 = new
System.Windows.Forms.TreeNode("Клас RBTree");
        System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(Theory));
        this.treeView1 = new System.Windows.Forms.TreeView();
        this.imageList1 = new System.Windows.Forms.ImageList(this.components);
        this.webBrowser1 = new System.Windows.Forms.WebBrowser();
        this.SuspendLayout();
        //
        // treeView1
        //
        this.treeView1.Font = new System.Drawing.Font("Microsoft Sans Serif",
11F);
        this.treeView1.ImageIndex = 1;
        this.treeView1.ImageList = this.imageList1;
        this.treeView1.Location = new System.Drawing.Point(12, 12);
        this.treeView1.Name = "treeView1";
        treeNode1.Name = "basic";
        treeNode1.Text = "Вступ до структур даних";
        treeNode2.Name = "BST";
        treeNode2.Text = "Двійкове дерево пошуку";
        treeNode3.Name = "RBProperties";
        treeNode3.Text = "RB-властивості";
        treeNode4.Name = "RBInsert";
        treeNode4.Text = "Вставка";
        treeNode5.Name = "RBDelete";
        treeNode5.Text = "Видалення";
        treeNode6.Name = "RBTree";
        treeNode6.Text = "Червоно-чорне дерево";
        treeNode7.Name = "RBNode";
        treeNode7.Text = "Клас RBNode";
        treeNode8.Name = "RBTreeCode";
        treeNode8.Text = "Клас RBTree";
        this.treeView1.Nodes.AddRange(new System.Windows.Forms.TreeNode[] {
        treeNode1,
        treeNode2,
        treeNode6,
        treeNode7,
        treeNode8});
        this.treeView1.SelectedImageIndex = 0;
```

```
            this.treeView1.ShowNodeToolTips = true;
            this.treeView1.ShowPlusMinus = false;
            this.treeView1.ShowRootLines = false;
            this.treeView1.Size = new System.Drawing.Size(219, 248);
            this.treeView1.TabIndex = 0;
            this.treeView1.AfterSelect += new
System.Windows.Forms.TreeViewEventHandler(this.treeView1_AfterSelect);
            //
            // imageList1
            //
            this.imageList1.ImageStream =
((System.Windows.Forms.ImageListStreamer)(resources.GetObject("imageList1.Ima
geStream")));
            this.imageList1.TransparentColor = System.Drawing.Color.Transparent;
            this.imageList1.Images.SetKeyName(0, "Cheked.png");
            this.imageList1.Images.SetKeyName(1, "Uncheked.png");
            //
            // webBrowser1
            //
            this.webBrowser1.Location = new System.Drawing.Point(237, 12);
            this.webBrowser1.MinimumSize = new System.Drawing.Size(20, 20);
            this.webBrowser1.Name = "webBrowser1";
            this.webBrowser1.Size = new System.Drawing.Size(735, 637);
            this.webBrowser1.TabIndex = 1;
            //
            // Theory
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(984, 661);
            this.Controls.Add(this.webBrowser1);
            this.Controls.Add(this.treeView1);
            this.MaximizeBox = false;
            this.MinimizeBox = false;
            this.Name = "Theory";
            this.ShowIcon = false;
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Теорія і код";
            this.Load += new System.EventHandler(this.Theory_Load);
            this.ResumeLayout(false);

        }

        #endregion
```

```
        private System.Windows.Forms.TreeView treeView1;
        private System.Windows.Forms.ImageList imageList1;
        private System.Windows.Forms.WebBrowser webBrowser1;
    }
}

namespace RedBlackTrees
{
    partial class Time
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.label1 = new System.Windows.Forms.Label();
            this.dataLabel = new System.Windows.Forms.Label();
            this.childrenLabel = new System.Windows.Forms.Label();
            this.label2 = new System.Windows.Forms.Label();
            this.parentLabel = new System.Windows.Forms.Label();
            this.label4 = new System.Windows.Forms.Label();
            this.label3 = new System.Windows.Forms.Label();
            this.label5 = new System.Windows.Forms.Label();
```

```
this.label6 = new System.Windows.Forms.Label();
this.label7 = new System.Windows.Forms.Label();
this.label8 = new System.Windows.Forms.Label();
this.colorLabel = new System.Windows.Forms.Label();
this.RBT = new System.Windows.Forms.Label();
this.BST = new System.Windows.Forms.Label();
this.ARR = new System.Windows.Forms.Label();
this.SuspendLayout();
//
// label1
//
this.label1.AutoSize = true;
this.label1.Location = new System.Drawing.Point(12, 9);
this.label1.Name = "label1";
this.label1.Size = new System.Drawing.Size(260, 24);
this.label1.TabIndex = 0;
this.label1.Text = "Значення шуканої вершини:";
//
// dataLabel
//
this.dataLabel.AutoSize = true;
this.dataLabel.Location = new System.Drawing.Point(267, 9);
this.dataLabel.Name = "dataLabel";
this.dataLabel.Size = new System.Drawing.Size(0, 24);
this.dataLabel.TabIndex = 1;
//
// childrenLabel
//
this.childrenLabel.AutoSize = true;
this.childrenLabel.Location = new System.Drawing.Point(173, 33);
this.childrenLabel.Name = "childrenLabel";
this.childrenLabel.Size = new System.Drawing.Size(0, 24);
this.childrenLabel.TabIndex = 0;
//
// label2
//
this.label2.AutoSize = true;
this.label2.Location = new System.Drawing.Point(12, 33);
this.label2.Name = "label2";
this.label2.Size = new System.Drawing.Size(165, 24);
this.label2.TabIndex = 0;
this.label2.Text = "Дочірні вершини:";
//
// parentLabel
//
```

```
this.parentLabel.AutoSize = true;
this.parentLabel.Location = new System.Drawing.Point(311, 57);
this.parentLabel.Name = "parentLabel";
this.parentLabel.Size = new System.Drawing.Size(0, 24);
this.parentLabel.TabIndex = 0;
//
// label4
//
this.label4.AutoSize = true;
this.label4.Location = new System.Drawing.Point(12, 57);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(303, 24);
this.label4.TabIndex = 0;
this.label4.Text = "Значення батьківської вершини:";
//
// label3
//
this.label3.AutoSize = true;
this.label3.Location = new System.Drawing.Point(12, 131);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(461, 24);
this.label3.TabIndex = 0;
this.label3.Text = "Витрачений час (мілісекунд/1000 операцій пошуку)";
//
// label5
//
this.label5.AutoSize = true;
this.label5.Location = new System.Drawing.Point(31, 155);
this.label5.Name = "label5";
this.label5.Size = new System.Drawing.Size(224, 24);
this.label5.TabIndex = 0;
this.label5.Text = "Червоно-чорне дерево:";
//
// label6
//
this.label6.AutoSize = true;
this.label6.Location = new System.Drawing.Point(31, 179);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(237, 24);
this.label6.TabIndex = 0;
this.label6.Text = "Двійкове дерево пошуку:";
//
// label7
//
this.label7.AutoSize = true;
```

```
this.label7.Location = new System.Drawing.Point(31, 203);
this.label7.Name = "label7";
this.label7.Size = new System.Drawing.Size(226, 24);
this.label7.TabIndex = 0;
this.label7.Text = "Вбудований у C# масив:";
//
// label8
//
this.label8.AutoSize = true;
this.label8.Location = new System.Drawing.Point(12, 81);
this.label8.Name = "label8";
this.label8.Size = new System.Drawing.Size(63, 24);
this.label8.TabIndex = 0;
this.label8.Text = "Колір:";
//
// colorLabel
//
this.colorLabel.AutoSize = true;
this.colorLabel.Location = new System.Drawing.Point(71, 81);
this.colorLabel.Name = "colorLabel";
this.colorLabel.Size = new System.Drawing.Size(0, 24);
this.colorLabel.TabIndex = 0;
//
// RBT
//
this.RBT.AutoSize = true;
this.RBT.Location = new System.Drawing.Point(248, 155);
this.RBT.Name = "RBT";
this.RBT.Size = new System.Drawing.Size(0, 24);
this.RBT.TabIndex = 0;
//
// BST
//
this.BST.AutoSize = true;
this.BST.Location = new System.Drawing.Point(264, 179);
this.BST.Name = "BST";
this.BST.Size = new System.Drawing.Size(0, 24);
this.BST.TabIndex = 0;
//
// ARR
//
this.ARR.AutoSize = true;
this.ARR.Location = new System.Drawing.Point(253, 203);
this.ARR.Name = "ARR";
this.ARR.Size = new System.Drawing.Size(0, 24);
```

```
    this.ARR.TabIndex = 0;
    //
    // Time
    //
    this.AutoScaleDimensions = new System.Drawing.SizeF(11F, 24F);
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
    this.ClientSize = new System.Drawing.Size(784, 261);
    this.Controls.Add(this.dataLabel);
    this.Controls.Add(this.label8);
    this.Controls.Add(this.label7);
    this.Controls.Add(this.label6);
    this.Controls.Add(this.ARR);
    this.Controls.Add(this.BST);
    this.Controls.Add(this.RBT);
    this.Controls.Add(this.label5);
    this.Controls.Add(this.label3);
    this.Controls.Add(this.label4);
    this.Controls.Add(this.colorLabel);
    this.Controls.Add(this.parentLabel);
    this.Controls.Add(this.label2);
    this.Controls.Add(this.childrenLabel);
    this.Controls.Add(this.label1);
    this.Font = new System.Drawing.Font("Microsoft Sans Serif", 14F);
    this.Margin = new System.Windows.Forms.Padding(6, 6, 6, 6);
    this.MaximizeBox = false;
    this.MinimizeBox = false;
    this.Name = "Time";
    this.ShowIcon = false;
    this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
    this.Text = "Пошук";
    this.Load += new System.EventHandler(this.Time_Load);
    this.ResumeLayout(false);
    this.PerformLayout();

}

#endregion

private System.Windows.Forms.Label label1;
private System.Windows.Forms.Label dataLabel;
private System.Windows.Forms.Label childrenLabel;
private System.Windows.Forms.Label label2;
private System.Windows.Forms.Label parentLabel;
private System.Windows.Forms.Label label4;
private System.Windows.Forms.Label label3;
```

```
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.Label label8;
        private System.Windows.Forms.Label colorLabel;
        private System.Windows.Forms.Label RBT;
        private System.Windows.Forms.Label BST;
        private System.Windows.Forms.Label ARR;
    }
}

namespace RedBlackTrees
{
    partial class Title
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(Title));
            this.label1 = new System.Windows.Forms.Label();
```

```
        this.label2 = new System.Windows.Forms.Label();
        this.label3 = new System.Windows.Forms.Label();
        this.label4 = new System.Windows.Forms.Label();
        this.label6 = new System.Windows.Forms.Label();
        this.label8 = new System.Windows.Forms.Label();
        this.label9 = new System.Windows.Forms.Label();
        this.label10 = new System.Windows.Forms.Label();
        this.label11 = new System.Windows.Forms.Label();
        this.SuspendLayout();
        //
        // label1
        //
        this.label1.AutoSize = true;
        this.label1.BackColor = System.Drawing.Color.Transparent;
        this.label1.Font = new System.Drawing.Font("Microsoft Sans Serif", 28F);
        this.label1.ForeColor = System.Drawing.Color.Snow;
        this.label1.Location = new System.Drawing.Point(119, 314);
        this.label1.Name = "label1";
        this.label1.Size = new System.Drawing.Size(755, 44);
        this.label1.TabIndex = 0;
        this.label1.Text = "Структура даних \"Червоно-чорне дерево\"";
        this.label1.Click += new System.EventHandler(this.label1_Click);
        this.label1.MouseEnter += new
System.EventHandler(this.label1_MouseEnter);
        this.label1.MouseLeave += new
System.EventHandler(this.label1_MouseLeave);
        //
        // label2
        //
        this.label2.BackColor = System.Drawing.Color.Transparent;
        this.label2.Font = new System.Drawing.Font("Microsoft Sans Serif", 14F);
        this.label2.ForeColor = System.Drawing.SystemColors.Control;
        this.label2.Location = new System.Drawing.Point(350, 246);
        this.label2.Name = "label2";
        this.label2.Size = new System.Drawing.Size(300, 79);
        this.label2.TabIndex = 1;
        this.label2.Text = "Науково-дослідницька робота на тему:";
        this.label2.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
        //
        // label3
        //
        this.label3.AutoSize = true;
        this.label3.BackColor = System.Drawing.Color.Transparent;
        this.label3.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F);
        this.label3.ForeColor = System.Drawing.SystemColors.Control;
```

```
this.label3.Location = new System.Drawing.Point(830, 494);
this.label3.Name = "label3";
this.label3.Size = new System.Drawing.Size(110, 18);
this.label3.TabIndex = 1;
this.label3.Text = "Мутель Роман";
//
// label4
//
this.label4.AutoSize = true;
this.label4.BackColor = System.Drawing.Color.Transparent;
this.label4.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F);
this.label4.ForeColor = System.Drawing.SystemColors.Control;
this.label4.Location = new System.Drawing.Point(452, 634);
this.label4.Name = "label4";
this.label4.Size = new System.Drawing.Size(96, 18);
this.label4.TabIndex = 1;
this.label4.Text = "Дніпро, 2021";
//
// label6
//
this.label6.BackColor = System.Drawing.Color.Transparent;
this.label6.Font = new System.Drawing.Font("Microsoft Sans Serif", 14F);
this.label6.ForeColor = System.Drawing.SystemColors.Control;
this.label6.Location = new System.Drawing.Point(150, 9);
this.label6.Name = "label6";
this.label6.Size = new System.Drawing.Size(700, 158);
this.label6.TabIndex = 1;
this.label6.Text = resources.GetString("label6.Text");
this.label6.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
//
// label8
//
this.label8.AutoSize = true;
this.label8.BackColor = System.Drawing.Color.Transparent;
this.label8.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F);
this.label8.ForeColor = System.Drawing.SystemColors.Control;
this.label8.Location = new System.Drawing.Point(830, 476);
this.label8.Name = "label8";
this.label8.Size = new System.Drawing.Size(95, 18);
this.label8.TabIndex = 1;
this.label8.Text = "Виконавець:";
//
// label9
//
this.label9.AutoSize = true;
```

```
            this.label9.BackColor = System.Drawing.Color.Transparent;
            this.label9.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F);
            this.label9.ForeColor = System.Drawing.SystemColors.Control;
            this.label9.Location = new System.Drawing.Point(830, 580);
            this.label9.Name = "label9";
            this.label9.Size = new System.Drawing.Size(78, 18);
            this.label9.TabIndex = 1;
            this.label9.Text = "Лавренюк";
            //
            // label10
            //
            this.label10.AutoSize = true;
            this.label10.BackColor = System.Drawing.Color.Transparent;
            this.label10.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F);
            this.label10.ForeColor = System.Drawing.SystemColors.Control;
            this.label10.Location = new System.Drawing.Point(830, 562);
            this.label10.Name = "label10";
            this.label10.Size = new System.Drawing.Size(142, 18);
            this.label10.TabIndex = 1;
            this.label10.Text = "Науковий керівник:";
            //
            // label11
            //
            this.label11.AutoSize = true;
            this.label11.BackColor = System.Drawing.Color.Transparent;
            this.label11.Font = new System.Drawing.Font("Microsoft Sans Serif", 11F);
            this.label11.ForeColor = System.Drawing.SystemColors.Control;
            this.label11.Location = new System.Drawing.Point(830, 598);
            this.label11.Name = "label11";
            this.label11.Size = new System.Drawing.Size(120, 18);
            this.label11.TabIndex = 1;
            this.label11.Text = "Ірина Валеріївна";
            //
            // Title
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.BackgroundImage =
global::RedBlackTrees_Mutel.Properties.Resources.bg;
            this.ClientSize = new System.Drawing.Size(984, 661);
            this.Controls.Add(this.label4);
            this.Controls.Add(this.label11);
            this.Controls.Add(this.label3);
            this.Controls.Add(this.label10);
            this.Controls.Add(this.label9);
```

```
            this.Controls.Add(this.label8);
            this.Controls.Add(this.label6);
            this.Controls.Add(this.label2);
            this.Controls.Add(this.label1);
            this.MaximizeBox = false;
            this.MaximumSize = new System.Drawing.Size(1000, 700);
            this.MinimizeBox = false;
            this.MinimumSize = new System.Drawing.Size(1000, 700);
            this.Name = "Title";
            this.ShowIcon = false;
            this.SizeGripStyle = System.Windows.Forms.SizeGripStyle.Hide;
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Титульна сторінка";
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.Label label2;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Label label8;
        private System.Windows.Forms.Label label9;
        private System.Windows.Forms.Label label10;
        private System.Windows.Forms.Label label11;
    }
}
```

ДОДАТОК Б

ВІДГУК

керівника економічного розділу

на кваліфікаційну роботу бакалавра

на тему: «Розробка інформаційно-навчального додатоку для вивчення структур даних на прикладі «червоно-чорного дерева»»

студента групи 122-18ск-1 Горгальова Дмитра Євгенійовича

Керівник економічного розділу
Зав. каф. ПЕП та ПУ, д.е.н. **О. Г. Вагонова**

ДОДАТОК В

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
Диплом Горгальов.doc	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
Диплом Горгальов.pdf	Пояснювальна записка до кваліфікаційної роботив форматі PDF.
Програма	
Program Горгальов.zip	Архів. Містить коди програми і откомпільовану програму.
Презентаці	
Презентація Горгальов.ppt	Презентація кваліфікаційної роботи.