МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
КАФЕДРА КОМП’ЮТЕРНИХ НАУК

МЕТОДИЧНІ ВКАЗІВКИ ДО КУРСОВОЇ РОБОТИ З ДИЦИПЛІНИ
"Прикладні алгоритми та структури даних"

для студентів спеціальності 122 - Комп’ютерні науки

Укладач
проф. Ільїн О.О.

Київ 2020

1. Вступ

В межах курсової роботи з дисципліни "Прикладні алгоритми та структури даних" студентам пропонується створити прикладний додаток, який дозволяє за допомогою способу графічної візуалізації та анімаційних ефектів дослідити роботу структури даних «двійкове дерево» (Binary Search Tree,BST). Концептуально, додаток являє собою програму-симулятор для навчальних цілей, за допомогою якої можна зрозуміти особливості побудови ієрархічної структури бінарного дерева під час основних операцій із деревом – додавання та видалення вузлів, обхід дерева. Операція обходу дерева (у різні способи) демонструється у додатку за допомогою анімації – вузли дерева, у порядку обходу, на певний час підсвічуються в тій чи інший спосіб. Реалізація даного додатку дозволяє студентам опанувати технологію створення програм-симуляторів, що включає: розробку моделі предметної області, підбір потрібних алгоритмів та обґрунтування використання структур даних, розробку графічного інтерфейсу додатку, застосування анімаційних ефектів для наочного відтворення особливостей роботи окремих внутрішніх процесів у програмі.

Вікно типового додатку зображене на рис.1,2.

Рисунок 1. Вікно додатку із побудованим деревом (обхід дерева не здійснено)

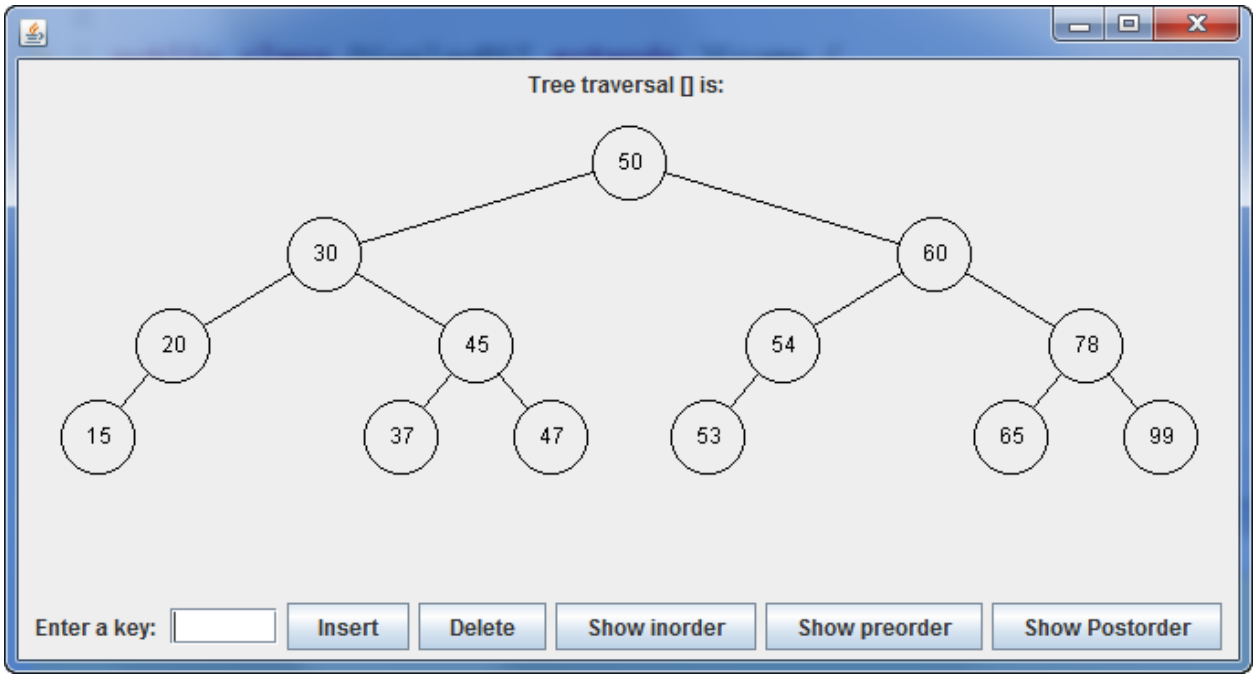
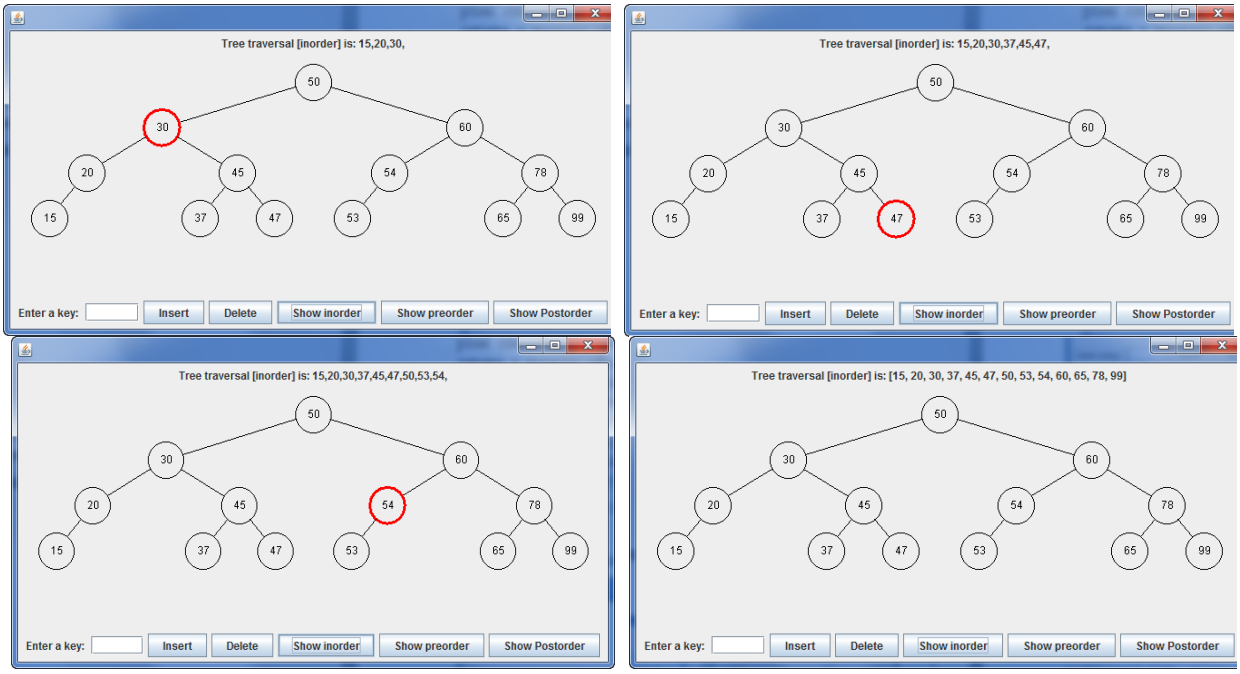


Рисунок 2. Вигляд вікна програми під час анімації обходу дерева



Реалізація кожним студентом програмного додатку відповідно до сформульованого технічного завдання, дозволяє закріпити отримані у курсі "Прикладні алгоритми та структури даних" загальні та професійні компетентності, а також компетентності, які сформульовані відповідно до вимог роботодавців, продемонструвати програмні результати навчання. Отримані компетентності стануть у нагоді при виконання дипломного проекту.

2. Технології для застосування та особливості реалізації

Базовою технологією, яка застосовуються в роботі, є Java. Для організації роботи важливе значення має необхідне програмне забезпечення розробника. Його основу складає JDK(1.8), середовище розробника IDE Eclipse або IntelliJ IDEA. Ці засоби необхідні для створення коду програми, її компіляції та запуску.

Реалізацію графічного інтерфейсу користувача необхідно здійснювати за допомогою графічного фреймворку Swing або JavaFX.

В програмі необхідно застосовувати структури даних, як ті, що містяться у Java Collection Framework, так і ті, що розробляються для даного проекту окремо.

Для контролю відвідування вузлів під час анімації обходу дерева, слід використати структуру даних «стек».

Для диференційованого способу візуалізації вузлів, які ще не були відвідані під час ободу, та тих, які ще мають бути відвідані, слід застосовувати окремі списки (масиви або множини - на власний розсуд, але треба обґрунтувати).

3. Перелік індивідуальних особливостей технічного завдання для реалізації

Кожний студент отримує індивідуальні особливості технічного завдання, які потрібно реалізувати у додатку.

Таблиця 1. Варіанти реалізації окремих елементів програми

Елемент \ Варіант		1	2	3
1	Загальне вікно додатку	Має фіксований розмір	Може змінюватись користувачем	-
2	Тип обходу, який треба реалізувати обов'язково	inorder	preorder	postorder
3	Стан кнопок під час анімації обходу	Активні (при натисненні, якщо здійснюється анімація обходу, вона переривається і здійснюється відповідна операція)	Не активні (надписи здійснено сірим кольором, при натисненні реакції немає)	-
4	Як виглядає вузол, який «підсвічено» під час анімації	Збільшена ширина обводки кола до 3, колір лінії обводки – червоний	Збільшена ширина обводки кола до 3, колір лінії обводки – червоний; всередині заливка світло-червоним кольором	Стандартна ширина обводки, всередині заливка червоним кольором, колір тексту – білий, стиль-напівжирний
5	Як виглядають вузли під час статичної візуалізації дерева (або не відвідані вузли під час анімації обходу дерева)	Коло із сірим коліром обводки, ширина 1, всередині текст сірим кольором, заповнення кольором немає	Коло із чорним коліром обводки, ширина 1, всередині текст чорним кольором, заповнення кольором немає	Коло заповнене сірим кольором, всередині текст чорним кольором, стиль-нормальний
6	Як виглядають вузли, які є вже відвіданими під час обходу	Коло із чорним коліром обводки, ширина 1, всередині текст чорним кольором, заповнення кольором немає	Коло заповнене сірим кольором, всередині текст чорним кольором, стиль-нормальний	Коло із сірим коліром обводки, ширина 1, всередині текст сірим кольором, заповнення кольором немає
7	Як виглядає сигналізація завершення анімації обходу дерева	Всі вузли отримують червону обводку шириною 3 на 1 секунду, потім оформлення вузла стає стандартним як для статичної візуалізації дерева	Всі вузли отримують заливку червоним кольором на 1 секунду, потім оформлення вузла стає стандартним як для статичної візуалізації дерева	У вузлах текст отримує червоний колір на 1 секунду, потім оформлення вузла стає стандартним як для статичної візуалізації дерева

Для кожного студента індивідуальне завдання задається числом із 7 цифр, кожна з яких означає відповідний спосіб реалізації елементу додатку (порядковий номер кожної цифри у коді означає номер елементу з таблиці 2, значення кожної цифри – номер варіанту з таблиці 2). Наприклад, завдання «1323213» означає, що елемент 1 «Загальне вікно додатку» має бути реалізоване відповідно до варіанту 1 («**1**323213»); елемент 3 «Тип обходу…» має бути реалізоване відповідно до варіанту 3 («1**3**23213») і т.п. Останній

елемент 7 «Як виглядає сигналізація…» має бути реалізований відповідно до варіанту 3 («132321**3**»).

Таблиця коду завдань для студентів у різних групах, відповідно до порядкового номеру ПІБ студента у списку, наведена у таблиці 2.

№	КНД-21	КНД-22	КНД-23
1	1123313	1211321	1223321
2	2122333	1313113	2213332
3	1222322	1213112	2311322
4	2323121	1312332	2223121
5	2312321	1213322	2323311
6	1212211	2112231	1112133
7	1321312	2211233	2321233
8	2211232	2212213	2121313
9	2313332	2122332	2122122
10	2221233	2211311	1112332
11	1223112	1111123	1313213
12	2323113	1213333	1311233
13	1323133	2121133	2122212
14	2222321	1111132	1311213
15	1222223	1123221	2212221
16	2113232	1121222	1223221
17	1111131	2311331	2311322
18	1212222	2321113	1121122
19	1221332	2321133	2211313
20	2213113	2321323	2323122
21	2212121	1211312	2221131
22	1323223	1223313	2123332
23	2213212	2113113	2222222
24	2221313	1111323	2213222
25	2122131	2123331	1112331
26	1212313	2311213	2221131
27	2113221	1312222	2123331
28	2213112	1123211	1322221
29	2212132	2323122	1322133
30	1113132	1123113	2313123
31	1222331	1123133	2221312
32	2123232	1212312	1322113
33	1323133	2223133	1123132
34	2213112	1112331	2121113
35	2112223	2323211	1222313

4. Опис курсової роботи у документації (опис роботи)

Разом із розробкою програмного додатку, необхідно розробити опис курсової роботи. Він представляє собою документ у паперовому виді, із титульною сторінкою, як у зразку в додатку А. Опис курсової роботи містить формулювання індивідуального завдання щодо розробки, порядок розробки та обґрунтування вибору засобів виконання, технологій java, алгоритмів та структур даних, мають бути присутні необхідні графічні пояснення принципу створення дерева (розташування вузлів на різних рівнях ієрархії), пояснення разом із UML-діаграмами класів самої програми, скріншоти вікон додатку, **коди всіх класів java.**

Опис складається з кількох розділів. У розділі №1 наводиться отримане індивідуальне завдання. В розділі №2 описується предметна модель додатку, обґрунтовуються обрані алгоритми та структури даних, пояснення принципу візуалізації дерева. В розділі №3 наводиться зображення графічного інтерфейсу розробленого додатку, описуються функціональні можливості додатку, наводяться UML- діаграми, особливості реалізації індивідуального завдання. У висновку підводяться підсумки виконаної роботи, відповідно до основних етапів розробки. В розділі №4 наводяться коди всіх класів java, їх UML-діаграми.

Обсяг опису роботи, без кодів програм - не більше 10 сторінок.
Порядок слідування частин роботи:

1) титульний лист (додаток А даних методичних рекомендацій);
2) зміст роботи;
3) розділ 1,2,3,…
4) бібліографічний перелік літератури або посилань Інтернет;

Робота оформлюється на листах А4, нумерується (перший лист без номера), скріплюється степлером з боку більшого відступу (ліве поле).

Для основного тексту роботи використовується шрифт Times New Roman 12 пт. Для програмного коду – моноширинийшрифт Courier New 10пт.

Код програми має містити коментарі. Коментар у заголовку кожного java-файлу, має ідентифікувати автора.

5. Порядок захисту роботи

Для захисту курсової роботи, її опис разом із кодом програмного додатку, має бути заздалегідь (за 3 дні до захисту) завантажений через систему електронного навчання. Код програми має бути в архіві zip, у вигляді папки src, в яких розташовані інші пакети та коди класів java. Оцінка складається з кількох частин:

Що оцінюється	Макс. бал
1. Функціонування програмного додатку	40
2. Відповідність вимогам щодо індивідуального завдання	35
3. Відповіді на питання по внутрішньому устрою та принципу роботи додатку (питання по коду java, наведеному у описі)	25

ДОДАТКИ

Додаток А

Приклад оформлення титульного листа курсової роботи

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Курсова робота

з дисципліни "Прикладне програмування Java" на тему: розробка програмного додатку моделювання та візуалізації подій

Роботу виконав здобувач вищої освіти Іванов В.О.

Перевірив: _____

Київ 2020

Додаток Б

Приклад схеми екранної форми додатку

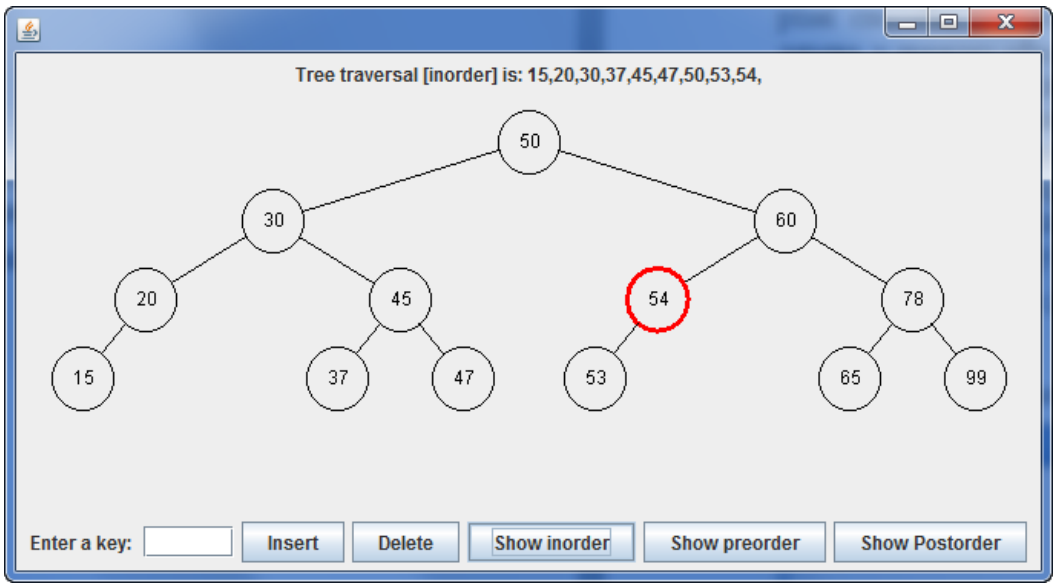


Рисунок 3. Екранна форма додатку із введеними даним та результатом їх обробки

Приклад оформлення коду програми

Код класу DisplayBST.java

```
// KND-11 Ivanov Alex, 22-12-2020
package cursova;

import javax.swing.*;

public class DisplayBST extends JFrame {
  public DisplayBST() {
    add(new TreeControl(new BST<Integer>()));
  }

  public static void main(String[] args) {
    JFrame frame = new DisplayBST();
//    frame.add(frame);
    frame.setSize(700, 500);
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setLocationRelativeTo(null);
    frame.setVisible(true);
  }
}
```