Звіт

Автор: Пумня О., КІТ118Б

Дата: 02.02.2020

Лабораторна робота №10

ОБРОБКА ПАРАМЕТРИЗОВАНИХ КОНТЕЙНЕРІВ

Мета. Розширення функціональності параметризованих класів.

Вимоги:

Використовуючи програму рішення завдання лабораторної роботи №9:

1. Розробити параметризовані методи (Generic Methods) для обробки колекцій об'єктів згідно прикладної задачі.
2. Продемонструвати розроблену функціональність (створення, управління та обробку власних контейнерів) в діалоговому та автоматичному режимах.
    - Автоматичний режим виконання програми задається параметром командного рядка -auto.
    - В автоматичному режимі діалог з користувачем відсутній, необхідні данні генеруються, або зчитуються з файлу.
3. Забороняється використання алгоритмів з Java Collections Framework.

ЗАВДАННЯ ДО РОБОТИ

Планувальник. Сортування за датою заходу, за тривалістю, за кількістю учасників.

ОПИС ПРОГРАМИ

Опис змінних

boolean isExit – Для виходу з програми
boolean isAuto – Для перевірки автоматичного режиму

Ієрархія та структура класів

class Pumnya10 – Точка входу в програму
class Dialog – Реалізація діалогового режиму
abstract class SortFilter – Клас, що містить у собі фільтр компаратору
class SortByDate – Сортування за датою заходу
class SortByDuration – Сортування за тривалістю заходу
class SortByPartAmount – Сортування за кількістю учасників заходу

ТЕКСТ ПРОГРАМИ

Текст файлу Pumnya10.java

```
package labs.pumnya10;
import java.io.IOException;
```

```
import java.util.Arrays;
import java.util.List;

public final class Pumnya10 {
    private Pumnya10() {
    }

    public static void main(final String[] args) throws IOException, ClassNotFoundException {
        boolean isExit = false;
        List<String> list = Arrays.asList(args);
        boolean isAuto = list.contains("-auto");

        while(!isExit) {
            isExit = Dialog.run(isAuto);
        }
    }
}
```

Текст файлу Dialog.java

```
package labs.pumnya10;
import labs.pumnya07.SchedulerEvent;
import labs.pumnya09.GenericList;
import labs.pumnya12.EventSearcher;
import java.io.*;
import java.util.Scanner;

public final class Dialog {
    private Dialog() {
    }
    private static boolean isAuto;
    /**
     * Для выбора пунктов в меню.
     */
    private static String choice;
    /**
     * Для ввода.
     */
    private static BufferedReader buffer = new BufferedReader(
            new InputStreamReader(System.in));
    /**
     * Универсальный список.
     */
    private static GenericList<SchedulerEvent> list = new GenericList<>();
    /**
     * Главный метод, сердце диалога.
     * @return true - если выбран выход
     * @throws IOException при ошибках со вводом
     */
    public static boolean run(boolean Auto) throws IOException, ClassNotFoundException {
        isAuto = Auto;
        if(isAuto) {
            autoProcessing();
            return true;
        } else {
            mainMenu();
            boolean flag = mainProcessing();
            System.out.println();
            return flag;
        }
    }
    /**
     * Главное меню диалога.
     * @throws IOException при ошибках со вводом
     */
    private static void mainMenu() throws IOException {
        System.out.println("1. Добавить мероприятие.");
        System.out.println("2. Удалить мероприятия.");
        System.out.println("3. Очистить список.");
        System.out.println("4. Вывод информации.");
        System.out.println("5. Сортировка.");
        System.out.println("6. Поиск мероприятий.");
        System.out.println("7. Сохранить в файл.");
        System.out.println("8. Загрузить из файла.");
        System.out.println("0. Выход.");
        System.out.print("Введите ваш ответ сюда: ");
        choice = buffer.readLine();
        System.out.println();
```

```
}
/**
 * Обработка выбора главного меню.
 * @return true - если выход
 * @throws IOException при ошибках со вводом
 * @throws ClassNotFoundException при ошибке с классами
 */
private static boolean mainProcessing() throws IOException, ClassNotFoundException {
    switch (choice) {
        case "1":
            onAdd();
            addProcessing();
            return false;

        case "2":
            onDelete();
            deleteProcessing();
            return false;

        case "3":
            if(list.getSize() == 0) {
                System.out.println("Список пуст!");
                return false;
            }
            System.out.println("Очистка...");
            list.clear();
            return false;

        case "4":
            if(list.getSize() == 0) {
                System.out.println("Список пуст!");
                return false;
            }
            System.out.print("Данные: ");
            System.out.print(list.toString());
            return false;

        case "5":
            if(list.getSize() == 0) {
                System.out.println("Список пуст!");
                return false;
            }
            onSort();
            sortProcessing();
            return false;

        case "6":
            if(list.getSize() == 0) {
                System.out.println("Список пуст!");
                return false;
            }
            System.out.print(EventSearcher.searchEvents(list));
            return false;

        case "7":
            System.out.println("Serialization...");
            ObjectOutputStream oos = new ObjectOutputStream(
                    new FileOutputStream("DataFile.dat"));
            oos.writeObject(list);
            oos.close();
            System.out.println("Done!\n");
            return false;

        case "8":
            System.out.println("Deserialization...");
            ObjectInputStream ois = new ObjectInputStream(
                    new FileInputStream("DataFile.dat"));
            GenericList<SchedulerEvent> list_copy =
                    (GenericList) ois.readObject();
            ois.close();
            System.out.println("Прочитанные данные: ");
            System.out.println(list_copy.toString());
            return false;
        case "0":
            System.out.print("Спасибо за работу!");
            return true;
        default:
```

```
            return false;
        }
    }
    /**
     * Меню добавления мероприятия.
     * @throws IOException при ошибках со вводом
     */
    private static void onAdd() throws IOException {
        System.out.println("1. Добавить в начало списка.");
        System.out.println("2. Добавить в конец списка.");
        System.out.println("3. Добавить по индексу.");
        System.out.println("Любая клавиша. Назад.");
        System.out.print("Введите ваш ответ сюда: ");
        choice = buffer.readLine();
        System.out.println();
    }
    /**
     * Обработка выбора меню добавления.
     * @throws IOException при ошибках со вводом
     */
    private static void addProcessing() throws IOException {
        Scanner scan = new Scanner(System.in);
        switch (choice) {
            case "1":
                list.pushFront(SchedulerEvent.generate(false));
                break;

            case "2":
                list.pushBack(SchedulerEvent.generate());
                break;

            case "3":
                System.out.print("Введите индекс: ");
                list.insert(scan.nextInt(), SchedulerEvent.generate());
                break;
        }
    }
    /**
     * Меню удаления мероприятия.
     * @throws IOException при ошибках со вводом
     */
    private static void onDelete() throws IOException {
        if (list.getSize() == 0) {
            System.out.println("Список пуст!");
        } else {
            System.out.println("1. Удалить первый.");
            System.out.println("2. Удалить последний.");
            System.out.println("3. Удалить по значению.");
            System.out.println("4. Удалить по индексу.");
            System.out.println("Любая клавиша. Назад.");
            System.out.print("Введите ваш ответ сюда: ");
            choice = buffer.readLine();
            System.out.println();
        }
    }
    /**
     * Обработка выбора меню удаления.
     * @throws IOException при ошибках со вводом
     */
    private static void deleteProcessing() throws IOException {
        Scanner scan = new Scanner(System.in);
        switch (choice) {
            case "1" :
                list.popFront();
                break;

            case "2" :
                list.popBack();
                break;

            case "3" :
                System.out.println("Введите данные:");
                list.remove(SchedulerEvent.generate());
                break;

            case "4" :
                System.out.print("Введите индекс:");
```

```
                list.remove(scan.nextInt());
                break;
        }
    }
    /**
     * Меню сортировки мероприятий.
     * @throws IOException при ошибках со вводом
     */
    private static void onSort() throws IOException {
        if (list.getSize() == 0) {
            System.out.println("Список пуст!");
        } else {
            System.out.println("1. Сортировка по дате.");
            System.out.println("2. Сортировка по длительности.");
            System.out.println("3. Сортировка по количеству участников.");
            System.out.println("Любая клавиша. Назад.");
            System.out.print("Введите ваш ответ сюда: ");
            choice = buffer.readLine();
            System.out.println();
        }
    }
    /** Обработка выбора меню сортировки. */
    private static void sortProcessing() {
        switch (choice) {
            case "1":
                list.sort(new SortByDate(null));
                break;

            case "2":
                list.sort(new SortByDuration(null));
                break;

            case "3":
                list.sort(new SortByPartAmount(null));
                break;
        }
    }
    /** Програма без диалога. */
    private static void autoProcessing() throws IOException {
        System.out.println("\nДобавим нелколько мероприятий:");
        list.addAll(SchedulerEvent.readFromFile("data.txt"));
        list.pushBack(SchedulerEvent.generate(false));
        System.out.println(list.toString());
        System.out.println("Удалим последнее мероприятие:");
        list.popBack();
        System.out.print(list.toString());
        System.out.println("Отсортируем длительности: ");
        list.sort(new SortByDuration(null));
        System.out.print(list.toString());
        System.out.println("Отсортируем дате: ");
        list.sort(new SortByDate(null));
        System.out.print(list.toString());
    }
}
```

Текст файлу SortFilter.java

```
package labs.pumnya10;
import java.util.Comparator;
public abstract class SortFilter<T> implements Comparator<T> {
    /** Фильтр компаратора. */
    Comparator <T> filter;
    public SortFilter(Comparator <T> comp) {
        filter = comp;
    }
}
```

Текст файлу SortByDate.java

```
package labs.pumnya10;
import labs.pumnya07.SchedulerEvent;
import java.util.Comparator;
public class SortByDate extends SortFilter<SchedulerEvent> {
    public SortByDate(Comparator<SchedulerEvent> comp) {
        super(comp);
    }

    @Override
```

```
    public int compare(SchedulerEvent o1, SchedulerEvent o2) {
        String[] ymd1 = o1.getDate().split("\\.");
        String[] ymd2 = o2.getDate().split("\\.");
        int result = ymd1[2].compareTo(ymd2[2]);
        if (result == 0) {
            result = ymd1[1].compareTo(ymd2[1]);
            if (result == 0) {
                result = ymd1[0].compareTo(ymd2[0]);
            }
        } else if (super.filter != null) {
            return filter.compare(o1, o2);
        }
        return result;
    }
}
```

Текст файлу SortByDuration.java

```
package labs.pumnya10;
import labs.pumnya07.SchedulerEvent;
import java.util.Comparator;
public class SortByDuration extends SortFilter<SchedulerEvent> {
    public SortByDuration(Comparator<SchedulerEvent> comp) {
        super(comp);
    }
    @Override
    public int compare(SchedulerEvent o1, SchedulerEvent o2) {
        float result = o1.getDuration() - o2.getDuration();
        if (super.filter != null) {
            return filter.compare(o1, o2);
        }
        if (result < 0) {
            return -1;
        } else if (result > 0) {
            return 1;
        } else {
            return 0;
        }
    }
}
```

Текст файлу SortByPartAmount.java

```
package labs.pumnya10;
import labs.pumnya07.SchedulerEvent;
import java.util.Comparator;
public class SortByPartAmount extends SortFilter<SchedulerEvent> {
    public SortByPartAmount(Comparator<SchedulerEvent> comp) {
        super(comp);
    }
    @Override
    public int compare(SchedulerEvent o1, SchedulerEvent o2) {
        int result = o1.getParticipants().size() - o2.getParticipants().size();
        if (super.filter != null) {
            return filter.compare(o1, o2);
        }
        return result;
    }
}
```

ВАРІАНТИ ВИКОРИСТАННЯ

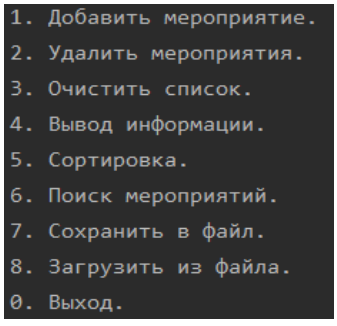
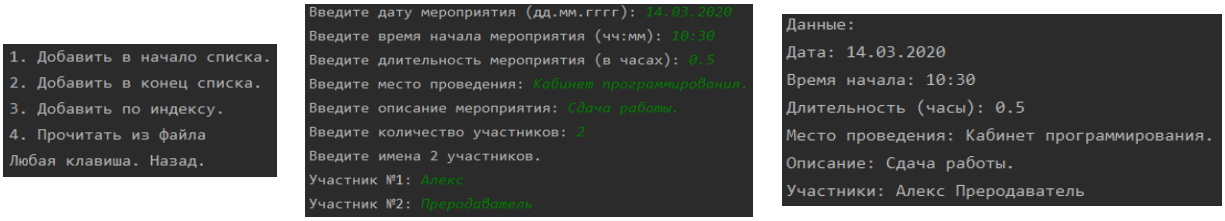


Рисунок 1 – Головне меню програми



Список додавання Процес додавання Результат

Рисунок 2 – Додавання елементів

Додамо ще декілька заходів.

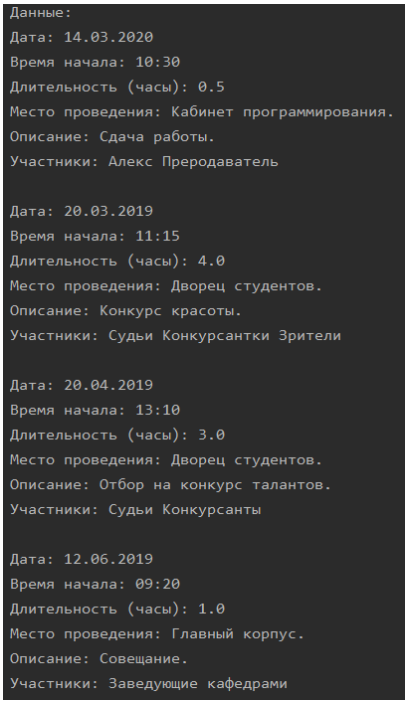
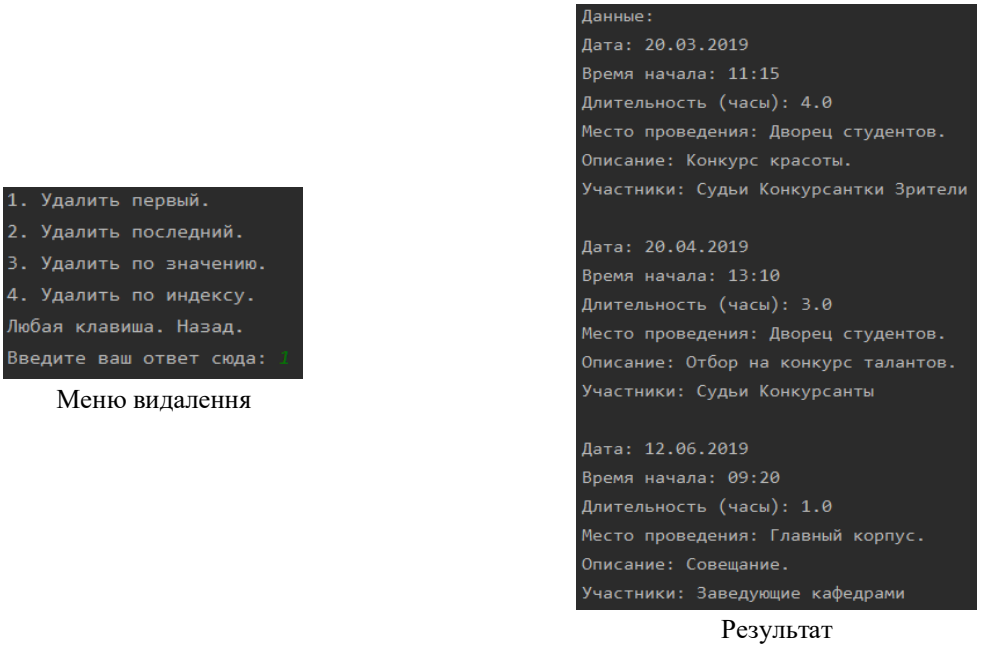


Рисунок 3 – Виведення даних



Меню видалення Результат

Рисунок 4 – Видалення елементів

```
1. Сортировка по дате.
2. Сортировка по длительности.
3. Сортировка по количеству участников.
Любая клавиша. Назад.
Введите ваш ответ сюда: 2
```

Меню сортування

```
Данные:
Дата: 12.06.2019
Время начала: 09:20
Длительность (часы): 1.0
Место проведения: Главный корпус.
Описание: Совещание.
Участники: Заведующие кафедрами

Дата: 20.04.2019
Время начала: 13:10
Длительность (часы): 3.0
Место проведения: Дворец студентов.
Описание: Отбор на конкурс талантов.
Участники: Судьи Конкурсанты

Дата: 20.03.2019
Время начала: 11:15
Длительность (часы): 4.0
Место проведения: Дворец студентов.
Описание: Конкурс красоты.
Участники: Судьи Конкурсантки Зрители
```

Результат

Рисунок 5 – Сортування елементів

```
Serialization...
Done!
```

Серіалізація

```
Deserialization...
Прочитанные данные:
Дата: 12.06.2019
Время начала: 09:20
Длительность (часы): 1.0
Место проведения: Главный корпус.
Описание: Совещание.
Участники: Заведующие кафедрами

Дата: 20.04.2019
Время начала: 13:10
Длительность (часы): 3.0
Место проведения: Дворец студентов.
Описание: Отбор на конкурс талантов.
Участники: Судьи Конкурсанты

Дата: 20.03.2019
Время начала: 11:15
Длительность (часы): 4.0
Место проведения: Дворец студентов.
Описание: Конкурс красоты.
Участники: Судьи Конкурсантки Зрители
```

Десеріалізація

Рисунок 6 – Процес серіалізації/десеріалізації

```
Добавим нелколько мероприятий:
Дата: 20.03.2019
Время начала: 11:15
Длительность (часы): 4.0
Место проведения: Дворец студентов.
Описание: Конкурс красоты.
Участники: Судьи Конкурсантки Зрители

Дата: 20.04.2019
Время начала: 13:10
Длительность (часы): 3.0
Место проведения: Дворец студентов.
Описание: Отбор на конкурс талантов.
Участники: Судьи Конкурсанты

Дата: 12.06.2019
Время начала: 09:20
Длительность (часы): 1.0
Место проведения: Главный корпус.
Описание: Совещание.
Участники: Заведующие кафедрами

Дата: 31.07.2001
Время начала: 11:20
Длительность (часы): 1.2
Место проведения: День Рождения!
Описание: Мой день рождения!
Участники: Алекс Мама Доктора
```

Додавання

```
Удалим последнее мероприятие:
Дата: 20.03.2019
Время начала: 11:15
Длительность (часы): 4.0
Место проведения: Дворец студентов.
Описание: Конкурс красоты.
Участники: Судьи Конкурсантки Зрители

Дата: 20.04.2019
Время начала: 13:10
Длительность (часы): 3.0
Место проведения: Дворец студентов.
Описание: Отбор на конкурс талантов.
Участники: Судьи Конкурсанты

Дата: 12.06.2019
Время начала: 09:20
Длительность (часы): 1.0
Место проведения: Главный корпус.
Описание: Совещание.
Участники: Заведующие кафедрами
```

Видалення

```
Отсортируем длительности:
Дата: 12.06.2019
Время начала: 09:20
Длительность (часы): 1.0
Место проведения: Главный корпус.
Описание: Совещание.
Участники: Заведующие кафедрами

Дата: 20.04.2019
Время начала: 13:10
Длительность (часы): 3.0
Место проведения: Дворец студентов.
Описание: Отбор на конкурс талантов.
Участники: Судьи Конкурсанты

Дата: 20.03.2019
Время начала: 11:15
Длительность (часы): 4.0
Место проведения: Дворец студентов.
Описание: Конкурс красоты.
Участники: Судьи Конкурсантки Зрители
```

```
Отсортируем дате:
Дата: 20.03.2019
Время начала: 11:15
Длительность (часы): 4.0
Место проведения: Дворец студентов.
Описание: Конкурс красоты.
Участники: Судьи Конкурсантки Зрители

Дата: 20.04.2019
Время начала: 13:10
Длительность (часы): 3.0
Место проведения: Дворец студентов.
Описание: Отбор на конкурс талантов.
Участники: Судьи Конкурсанты

Дата: 12.06.2019
Время начала: 09:20
Длительность (часы): 1.0
Место проведения: Главный корпус.
Описание: Совещание.
Участники: Заведующие кафедрами
```

Сортування

Рисунок 7 – Автоматичне виконання програми

ВИСНОВКИ

При виконанні лабораторної роботи набуто практичних навичок обробки параметризованих контейнерів. Створено методи сортування контейнеру.