Національний технічний університет України

«Київський політехнічний інститут ім. Ігоря Сікорського»

Кафедра цифрових технологій в енергетиці

ЗВІТ   
з виконання лабораторної роботи №1  
з дисципліни «Технології Java конструювання програмного забезпечення»

«**РОБОТА З КОЛЕКЦІЯМИ**»

Варіант 17

Виконав: студент групи ТР-23

Ровний Г.О.

Перевірив: доцент, к.ф.-м.н.  
Тарнавський Ю.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Завдання 1

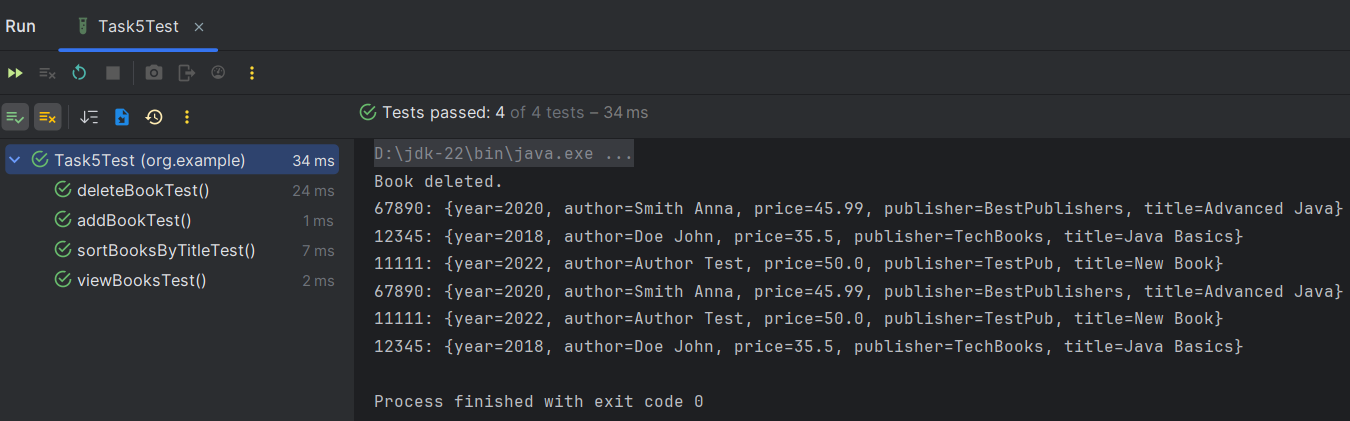
Код :

package org.example;  
  
import java.util.\*;  
  
public class Task5 {  
  
 public static final Map<Integer, Map<String, Object>> *catalog* = new HashMap<>();  
  
 public static void main(String[] args) {  
 *addInitialBooks*();  
 *showMenu*();  
 }  
  
 public static void addInitialBooks() {  
 *addBook*(12345, "Java Basics", "Doe John", "TechBooks", 2018, 35.50);  
 *addBook*(67890, "Advanced Java", "Smith Anna", "BestPublishers", 2020, 45.99);  
 *addBook*(11223, "Data Structures", "Brown Bob", "AlgoPress", 2019, 30.00);  
 *addBook*(33445, "Algorithms", "Johnson Mike", "LearnBooks", 2017, 25.00);  
 *addBook*(55667, "Design Patterns", "Williams Kate", "TechBooks", 2021, 40.75);  
 }  
  
 public static void addBook(int isbn, String title, String author, String publisher, int year, double price) {  
 Map<String, Object> book = new HashMap<>();  
 book.put("title", title);  
 book.put("author", author);  
 book.put("publisher", publisher);  
 book.put("year", year);  
 book.put("price", price);  
 *catalog*.put(isbn, book);  
 }  
  
 public static void showMenu() {  
 Scanner scanner = new Scanner(System.*in*);  
 while (true) {  
 System.*out*.println("1. View all books");  
 System.*out*.println("2. Delete a book by ISBN");  
 System.*out*.println("3. Sort books by title");  
 System.*out*.println("4. Sort books by author");  
 System.*out*.println("5. Sort books by year and price");  
 System.*out*.println("Any other key to exit");  
 String choice = scanner.nextLine();  
  
 switch (choice) {  
 case "1" -> *viewBooks*();  
 case "2" -> {  
 System.*out*.print("Enter ISBN to delete: ");  
 int isbn = Integer.*parseInt*(scanner.nextLine());  
 *deleteBook*(isbn);  
 }  
 case "3" -> *sortBooksBy*("title");  
 case "4" -> *sortBooksBy*("author");  
 case "5" -> *sortBooksBy*("year", "price");  
 default -> {  
 System.*out*.println("Exiting.");  
 return;  
 }  
 }  
 }  
 }  
  
 public static void viewBooks() {  
 *catalog*.forEach((isbn, book) -> System.*out*.println(isbn + ": " + book));  
 }  
  
 public static void deleteBook(int isbn) {  
 if (*catalog*.remove(isbn) != null) {  
 System.*out*.println("Book deleted.");  
 } else {  
 System.*out*.println("Book not found.");  
 }  
 }  
  
 public static void sortBooksBy(String... keys) {  
 List<Map.Entry<Integer, Map<String, Object>>> bookList = new ArrayList<>(*catalog*.entrySet());  
 bookList.sort(Comparator.*comparing*(entry -> entry.getValue().get(keys[0]).toString()));  
  
 for (int i = 1; i < keys.length; i++) {  
 final int index = i;  
 bookList.sort(Comparator.*comparing*(entry -> entry.getValue().get(keys[index]).toString()));  
 }  
  
 bookList.forEach(entry -> System.*out*.println(entry.getKey() + ": " + entry.getValue()));  
 }  
}

Тести:

package org.example;  
  
import org.junit.jupiter.api.Test;  
import static org.junit.jupiter.api.Assertions.\*;  
import org.junit.jupiter.api.BeforeEach;  
  
class Task5Test {  
  
 @BeforeEach  
 void setUp() {  
 Task5.*addBook*(12345, "Java Basics", "Doe John", "TechBooks", 2018, 35.50);  
 Task5.*addBook*(67890, "Advanced Java", "Smith Anna", "BestPublishers", 2020, 45.99);  
 }  
  
 @Test  
 void addBookTest() {  
 int initialSize = Task5.*catalog*.size();  
 Task5.*addBook*(11111, "New Book", "Author Test", "TestPub", 2022, 50.00);  
 *assertEquals*(initialSize + 1, Task5.*catalog*.size());  
 }  
  
 @Test  
 void deleteBookTest() {  
 *assertTrue*(Task5.*catalog*.containsKey(12345));  
 Task5.*deleteBook*(12345);  
 *assertFalse*(Task5.*catalog*.containsKey(12345));  
 }  
  
 @Test  
 void viewBooksTest() {  
 Task5.*viewBooks*();  
 *assertEquals*(3, Task5.*catalog*.size());  
 }  
  
 @Test  
 void sortBooksByTitleTest() {  
 Task5.*sortBooksBy*("title");  
 *assertEquals*("Advanced Java", Task5.*catalog*.get(67890).get("title"));  
 }  
}

Скріншот виконання тестів:



# Завдання 2

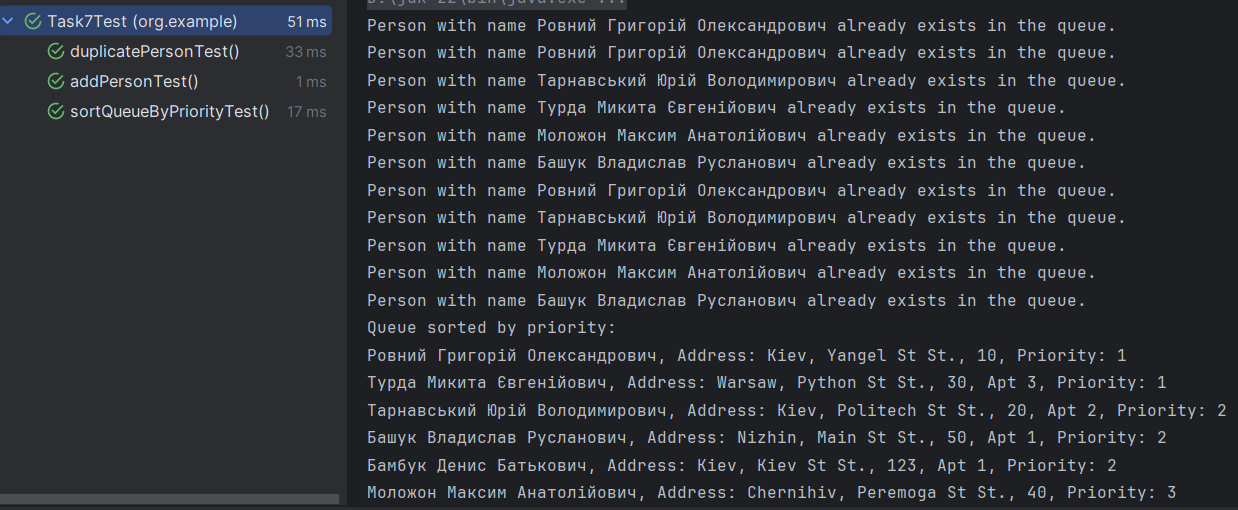
Код:

package org.example;  
  
import java.util.\*;  
  
public class Task7 {  
  
 // Клас для адреси  
 static class AddressValue {  
 String city;  
 String street;  
 int houseNumber;  
 int apartmentNumber;  
  
 public AddressValue(String city, String street, int houseNumber, int apartmentNumber) {  
 this.city = city;  
 this.street = street;  
 this.houseNumber = houseNumber;  
 this.apartmentNumber = apartmentNumber;  
 }  
  
 @Override  
 public String toString() {  
 return city + ", " + street + " St., " + houseNumber + (apartmentNumber != 0 ? ", Apt " + apartmentNumber : "");  
 }  
 }  
  
 // Клас для черговика  
 static class QueuePerson {  
 String lastName;  
 String firstName;  
 String middleName;  
 AddressValue address;  
 int priority;  
  
 public QueuePerson(String lastName, String firstName, String middleName, AddressValue address, int priority) {  
 this.lastName = lastName;  
 this.firstName = firstName;  
 this.middleName = middleName;  
 this.address = address;  
 this.priority = priority;  
 }  
  
 @Override  
 public String toString() {  
 return lastName + " " + firstName + " " + middleName + ", Address: " + address + ", Priority: " + priority;  
 }  
 }  
  
 public static final LinkedList<QueuePerson> *queue* = new LinkedList<>();  
 public static final Set<String> *uniqueNames* = new TreeSet<>();  
  
 public static void main(String[] args) {  
 *addInitialPeople*();  
 *showMenu*();  
 }  
  
 public static void addInitialPeople() {  
 *addPerson*("Ровний", "Григорій", "Олександрович", new AddressValue("Kiev", "Yangel St", 10, 0), 1);  
 *addPerson*("Тарнавський", "Юрій", "Володимирович", new AddressValue("Kiev", "Politech St", 20, 2), 2);  
 *addPerson*("Турда", "Микита", "Євгенійович", new AddressValue("Warsaw", "Python St", 30, 3), 1);  
 *addPerson*("Моложон", "Максим", "Анатолійович", new AddressValue("Chernihiv", "Peremoga St", 40, 0), 3);  
 *addPerson*("Башук", "Владислав", "Русланович", new AddressValue("Nizhin", "Main St", 50, 1), 2);  
 }  
  
 public static void addPerson(String lastName, String firstName, String middleName, AddressValue address, int priority) {  
 String fullName = lastName + " " + firstName + " " + middleName;  
 if (!*uniqueNames*.contains(fullName)) {  
 *queue*.add(new QueuePerson(lastName, firstName, middleName, address, priority));  
 *uniqueNames*.add(fullName);  
 } else {  
 System.*out*.println("Person with name " + fullName + " already exists in the queue.");  
 }  
 }  
  
 public static void showMenu() {  
 Scanner scanner = new Scanner(System.*in*);  
 while (true) {  
 System.*out*.println("1. View all queue persons");  
 System.*out*.println("2. Add a person to the queue");  
 System.*out*.println("3. Sort queue by priority");  
 System.*out*.println("Any other key to exit");  
 String choice = scanner.nextLine();  
  
 switch (choice) {  
 case "1" -> *viewQueue*();  
 case "2" -> {  
 System.*out*.print("Enter last name: ");  
 String lastName = scanner.nextLine();  
 System.*out*.print("Enter first name: ");  
 String firstName = scanner.nextLine();  
 System.*out*.print("Enter middle name: ");  
 String middleName = scanner.nextLine();  
 System.*out*.print("Enter city: ");  
 String city = scanner.nextLine();  
 System.*out*.print("Enter street: ");  
 String street = scanner.nextLine();  
 System.*out*.print("Enter house number: ");  
 int houseNumber = Integer.*parseInt*(scanner.nextLine());  
 System.*out*.print("Enter apartment number: ");  
 int apartmentNumber = Integer.*parseInt*(scanner.nextLine());  
 System.*out*.print("Enter priority: ");  
 int priority = Integer.*parseInt*(scanner.nextLine());  
 *addPerson*(lastName, firstName, middleName, new AddressValue(city, street, houseNumber, apartmentNumber), priority);  
 }  
 case "3" -> *sortQueueByPriority*();  
 default -> {  
 System.*out*.println("Exiting.");  
 return;  
 }  
 }  
 }  
 }  
  
 public static void viewQueue() {  
 if (*queue*.isEmpty()) {  
 System.*out*.println("The queue is empty.");  
 } else {  
 *queue*.forEach(System.*out*::println);  
 }  
 }  
  
 public static void sortQueueByPriority() {  
 *queue*.sort(Comparator.*comparingInt*(p -> p.priority));  
 System.*out*.println("Queue sorted by priority:");  
 *viewQueue*();  
 }  
}

Тести:

package org.example;  
  
import org.junit.jupiter.api.BeforeEach;  
import org.junit.jupiter.api.Test;  
import java.util.\*;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
class Task7Test {  
  
 @BeforeEach  
 void setUp() {  
 Task7.*addInitialPeople*();  
 }  
  
 @Test  
 void addPersonTest() {  
 Task7.*addPerson*("Бамбук", "Денис", "Батькович", new Task7.AddressValue("Kiev",  
 "Kiev St", 123, 1), 2);  
 *assertEquals*(6, Task7.*queue*.size());  
 }  
  
 @Test  
 void duplicatePersonTest() {  
 Task7.*addPerson*("Ровний", "Григорій", "Олександрович",  
 new Task7.AddressValue("Kiev", "Yangel St", 10,  
 0), 1);  
 *assertEquals*(5, Task7.*queue*.size());// дублікат не був доданий  
 }  
  
 @Test  
 void sortQueueByPriorityTest() {  
 Task7.*sortQueueByPriority*();  
 List<Integer> priorities = new ArrayList<>();  
 Task7.*queue*.forEach(person -> priorities.add(person.priority));  
 *assertEquals*(Arrays.*asList*(1, 1, 2, 2, 2, 3), priorities);  
 }  
}

Скріншот роботи тестів:



# Завдання 3

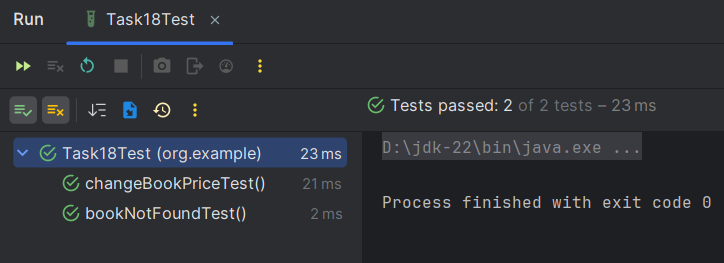
Код:

package org.example;  
  
import java.util.\*;  
  
public class Task18 {  
  
 public static final HashMap<Integer, Book> *catalog* = new HashMap<>();  
  
  
 static class Book {  
 String title;  
 String authorLastName;  
 String authorFirstName;  
 String authorMiddleName;  
 String publisher;  
 int year;  
 float price;  
  
 public Book(String title, String authorLastName, String authorFirstName, String authorMiddleName, String publisher, int year, float price) {  
 this.title = title;  
 this.authorLastName = authorLastName;  
 this.authorFirstName = authorFirstName;  
 this.authorMiddleName = authorMiddleName;  
 this.publisher = publisher;  
 this.year = year;  
 this.price = price;  
 }  
  
 @Override  
 public String toString() {  
 return "Book{" +  
 "title='" + title + '\'' +  
 ", author='" + authorLastName + " " + authorFirstName + " " + authorMiddleName + '\'' +  
 ", publisher='" + publisher + '\'' +  
 ", year=" + year +  
 ", price=" + price +  
 '}';  
 }  
 }  
  
 public static void main(String[] args) {  
 *addInitialBooks*();  
 *showMenu*();  
 }  
  
 public static void addInitialBooks() {  
 *catalog*.put(97809, new Book("Effective Java", "Bloch", "Joshua", "N/A", "Addison-Wesley", 2018, 45.99f));  
 *catalog*.put(97802, new Book("Design Patterns", "Gamma", "Erich", "Richard", "Addison-Wesley", 1994, 39.99f));  
 *catalog*.put(97805, new Book("Head First Java", "Sierra", "Kathy", "Bates", "O'Reilly Media", 2005, 29.99f));  
 *catalog*.put(97801, new Book("Clean Code", "Martin", "Robert", "Prentice Hall", "New York Times", 2008, 37.50f));  
 *catalog*.put(97807, new Book("Java Concurrency in Practice", "Goetz", "Brian", "Addison-Wesley", "O'Reilly Media", 2006, 49.99f));  
 }  
  
 public static void showMenu() {  
 Scanner scanner = new Scanner(System.*in*);  
 while (true) {  
 System.*out*.println("1. View all books");  
 System.*out*.println("2. Change book price");  
 System.*out*.println("3. Sort books by publisher");  
 System.*out*.println("4. Sort books by price");  
 System.*out*.println("Any other key to exit");  
 String choice = scanner.nextLine();  
  
 switch (choice) {  
 case "1" -> *viewBooks*();  
 case "2" -> *changeBookPrice*(scanner);  
 case "3" -> *sortBooksByPublisher*();  
 case "4" -> *sortBooksByPrice*();  
 default -> {  
 System.*out*.println("Exiting.");  
 return;  
 }  
 }  
 }  
 }  
  
 public static void viewBooks() {  
 if (*catalog*.isEmpty()) {  
 System.*out*.println("The catalog is empty.");  
 } else {  
 *catalog*.forEach((isbn, book) -> System.*out*.println("ISBN: " + isbn + ", " + book));  
 }  
 }  
  
 public static void changeBookPrice(Scanner scanner) {  
 System.*out*.print("Enter ISBN of the book: ");  
 int isbn = Integer.*parseInt*(scanner.nextLine());  
 if (*catalog*.containsKey(isbn)) {  
 System.*out*.print("Enter new price: ");  
 float newPrice = Float.*parseFloat*(scanner.nextLine());  
 *catalog*.get(isbn).price = newPrice;  
 System.*out*.println("Price updated successfully.");  
 } else {  
 System.*out*.println("Book with the given ISBN not found.");  
 }  
 }  
  
 public static void sortBooksByPublisher() {  
 List<Map.Entry<Integer, Book>> list = new ArrayList<>(*catalog*.entrySet());  
 list.sort(Comparator.*comparing*(entry -> entry.getValue().publisher));  
 System.*out*.println("Books sorted by publisher:");  
 list.forEach(entry -> System.*out*.println("ISBN: " + entry.getKey() + ", " + entry.getValue()));  
 }  
  
 public static void sortBooksByPrice() {  
 List<Map.Entry<Integer, Book>> list = new ArrayList<>(*catalog*.entrySet());  
 list.sort(Comparator.*comparingDouble*(entry -> entry.getValue().price));  
 System.*out*.println("Books sorted by price:");  
 list.forEach(entry -> System.*out*.println("ISBN: " + entry.getKey() + ", " + entry.getValue()));  
 }  
}

Тести:

package org.example;  
  
import org.junit.jupiter.api.BeforeEach;  
import org.junit.jupiter.api.Test;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
class Task18Test {  
  
 @BeforeEach  
 void setUp() {  
 Task18.*addInitialBooks*();  
 }  
  
 @Test  
 void changeBookPriceTest() {  
 int isbn = 97801;  
 float newPrice = 50.99f;  
 Task18.*catalog*.get(isbn).price = newPrice;  
 *assertEquals*(newPrice, Task18.*catalog*.get(isbn).price);  
 }  
  
 @Test  
 void bookNotFoundTest() {  
 int isbn = 1234567890;  
 *assertFalse*(Task18.*catalog*.containsKey(isbn));  
 }  
  
}

Скріншот роботи тестів:



# Завдання 4

Код:

package org.example;  
  
import java.util.LinkedList;  
import java.util.Scanner;  
  
public class Task24 {  
  
 static class StackMember {  
 String programName;  
 int memory;  
 int priority;  
  
 public StackMember(String programName, int memory, int priority) {  
 this.programName = programName;  
 this.memory = memory;  
 this.priority = priority;  
 }  
  
 @Override  
 public String toString() {  
 return "Program: " + programName + ", Memory: " + memory + " MB, Priority: " + priority;  
 }  
 }  
  
 public static LinkedList<StackMember> *stack* = new LinkedList<>();  
  
 public static void main(String[] args) {  
 *initializeStack*();  
 *interactWithUser*();  
 }  
  
 public static void initializeStack() {  
 *stack*.add(new StackMember("Program A", 512, 1));  
 *stack*.add(new StackMember("Program B", 256, 2));  
 *stack*.add(new StackMember("Program C", 1024, 0));  
 *stack*.add(new StackMember("Program D", 128, 3));  
 *stack*.add(new StackMember("Program E", 256, 2));  
 *sortStack*(); // Викликаємо метод для сортування стека  
 }  
  
 public static void interactWithUser() {  
 Scanner scanner = new Scanner(System.*in*);  
 while (true) {  
 System.*out*.println("Stack contents: " + *stack*);  
 System.*out*.println("Select an option:");  
 System.*out*.println("1. Add program to stack");  
 System.*out*.println("2. Remove program from stack");  
 System.*out*.println("3. Exit");  
 int choice = scanner.nextInt();  
 scanner.nextLine(); // consume newline  
  
 switch (choice) {  
 case 1:  
 System.*out*.print("Enter program name: ");  
 String name = scanner.nextLine();  
 System.*out*.print("Enter memory size (MB): ");  
 int memory = scanner.nextInt();  
 System.*out*.print("Enter priority (lower number means higher priority): ");  
 int priority = scanner.nextInt();  
 *addProgramToStack*(name, memory, priority);  
 break;  
 case 2:  
 *removeProgramFromStack*();  
 break;  
 case 3:  
 scanner.close();  
 return;  
 default:  
 System.*out*.println("Invalid choice. Please try again.");  
 }  
 }  
 }  
  
 public static void addProgramToStack(String name, int memory, int priority) {  
 StackMember newProgram = new StackMember(name, memory, priority);  
 *stack*.add(newProgram);  
 *sortStack*(); // Сортуємо стек після додавання нової програми  
 System.*out*.println("Program added: " + newProgram);  
 }  
  
 public static void removeProgramFromStack() {  
 if (!*stack*.isEmpty()) {  
 StackMember removedProgram = *stack*.removeFirst();  
 System.*out*.println("Removed program: " + removedProgram);  
 } else {  
 System.*out*.println("The stack is empty!");  
 }  
 }  
  
 public static void sortStack() {  
 *stack*.sort((a, b) -> Integer.*compare*(a.priority, b.priority));  
 }  
}

Тести:

package org.example;  
  
import org.junit.jupiter.api.BeforeEach;  
import org.junit.jupiter.api.Test;  
  
import java.util.LinkedList;  
  
import static org.junit.jupiter.api.Assertions.\*;  
  
class Task24Test {  
  
 @BeforeEach  
 void setUp() {  
 Task24.*stack*.clear();  
 Task24.*initializeStack*();  
 }  
  
 @Test  
 void addProgramToStackTest() {  
 int initialSize = Task24.*stack*.size();  
 Task24.*addProgramToStack*("Program F", 512, 2);  
 *assertEquals*(initialSize + 1, Task24.*stack*.size());  
 *assertEquals*("Program D", Task24.*stack*.getLast().programName);  
 }  
  
 @Test  
 void removeProgramFromStackTest() {  
 Task24.*removeProgramFromStack*();  
 *assertEquals*(4, Task24.*stack*.size());  
 }  
  
 @Test  
 void removeFromEmptyStackTest() {  
 Task24.*stack*.clear();  
 Task24.*removeProgramFromStack*();  
 *assertEquals*(0, Task24.*stack*.size());  
 }  
}

Скрін роботи тестів:

