МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ



**Дніпровський національний університет  
залізничного транспорту імені академіка В. Лазаряна**

Кафедра «Комп’ютерні інформаційні технології»

**Лабораторна робота №4**

**з дисципліни «Алгоритми та структури даних»**

**на тему: «Робота з шаблонами (generic java)»**

Виконав: студент гр. ПЗ1911

Сіньков Г.О.

Прийняла: ас. каф. КІТ

Куроп’ятник О. С.

Дніпро, 2020

**Лабораторна робота №4**

**Тема.** Робота з шаблонами (generic java).

**Мета**. Ознайомитися з поняттям узагальнення, варіантами його застосування. Отримати практичні навички роботи з колекціями як різновидом узагальнень.

**Постановка задачі**

|  |  |  |
| --- | --- | --- |
| Вариант | Ситуація | Значення параметризованого типу |
| 7 | Фотограф шикує дітей у колону по одному таким чином, щоб усіх було видно у кадрі. Нетерплячі діти можуть виходити з колони. | Дитина |

**Текст програми**

Child.java

package com.company;  
  
  
public class Child implements Comparable<Child>{  
 private final int height;  
  
 public Child(int height) {  
 this.height = height;  
 }  
  
 public int getHeight() {  
 return height;  
 }  
  
 @Override  
 public int compareTo(Child o) {  
 //давайте будем сортировать объекты Employee по значению поля id от меньшего к большему  
 //будем возвращать отрицательное число, 0 или положительное число по каждому сравнению объектов  
 // здесь мы просто отнимаем значение поля одного объекта от значения поля другого объекта  
 // в результате получим 1 из 3 вариантов описанных выше вариантов  
 return (this.height - o.height);  
 }  
}

Main.java

package com.company;  
  
import java.util.\*;  
  
public class Main {  
  
 public static int randomHeight() {  
 return (int) (Math.*random*() \* 50 + 150);  
 }  
 public static int randomNumber() {  
 return (int) (Math.*random*() \* 9 + 0);  
 }  
 public static void sort(int[] numberImpatientChildren) {  
 for (int left = 0; left < numberImpatientChildren.length; left++) {  
 int value = numberImpatientChildren[left]; // Вытаскиваем значение элемента  
 int i = left - 1;// Перемещаемся по элементам, которые перед вытащенным элементом  
 for (; i >= 0; i--) {  
 if (value > numberImpatientChildren[i]) // Если вытащили значение меньшее — передвигаем больший элемент дальше  
 numberImpatientChildren[i + 1] = numberImpatientChildren[i];  
 else// Если вытащенный элемент больше — останавливаемся  
 break;  
 }  
 numberImpatientChildren[i + 1] = value;// В освободившееся место вставляем вытащенное значение  
 }  
 }  
  
 public static void main(String[] args) {  
 LinkedList<Child> column = new LinkedList<>();  
 int tmp;  
 System.*out*.println("Колона отсортированых детей");  
  
 for (int i = 0; i < 10; i++) {//заполнение масива ростом детей  
 tmp = *randomHeight*();  
 if (column.isEmpty())  
 column.add(new Child(tmp));  
 else {  
 int pos = 0;  
 boolean k = true;  
 for (ListIterator<Child> itr = column.listIterator(); itr.hasNext() && k; pos++) {  
 Child element = itr.next();  
 if (tmp < element.getHeight()) {  
 column.add(pos, new Child(tmp));  
 k = false;  
 } else if (column.size() == 1) {  
 column.add(new Child(tmp));  
 k = false;  
 } else if (pos + 1 == column.size()){  
 column.add(new Child(tmp));  
 k = false;  
 }  
 }  
 }  
 }  
 for (Child t : column)//вывод колоны  
 System.*out*.println(t.getHeight());  
  
 int quantityImpatientChildren = *randomNumber*();//количество непослушних детей  
 if (quantityImpatientChildren != 0) {  
 int[] numberImpatientChildren = new int[quantityImpatientChildren];//номера непослушних детей  
  
 for (int i = 0; i < quantityImpatientChildren; i++) {//выбор номеров непослушных детей  
 numberImpatientChildren[i] = *randomNumber*();  
 }  
  
 *sort*(numberImpatientChildren);  
  
 for (int i = 0; i < numberImpatientChildren.length; i++)  
 for (int j = i + 1; j < numberImpatientChildren.length; j++)  
 if(numberImpatientChildren[i] == numberImpatientChildren[j])  
 numberImpatientChildren[i] = *randomNumber*();  
  
 *sort*(numberImpatientChildren);  
  
 //вывод количества и номеров непослушных детей  
 System.*out*.println();  
 System.*out*.println("Количество непослушних детей");  
 System.*out*.println(quantityImpatientChildren);  
 System.*out*.println("Ихнее номера");  
 for (int i = 0; i < quantityImpatientChildren; i++)  
 System.*out*.print(numberImpatientChildren[i] + " ");  
  
  
 //удаление детей по номерам из колоны  
 for (int i = 0; i < quantityImpatientChildren; i++) {  
 column.remove(numberImpatientChildren[i]);  
 }  
 } else  
 System.*out*.println("Детей которые вышли из колоны 0!");  
  
 System.*out*.println();  
 System.*out*.println("Колона без непослушних детей");  
 for (Child t : column)  
 System.*out*.println(t.getHeight());  
  
 }  
}

Висновок: Java Collection Framework - ієрархія інтерфейсів і їх реалізацій, яка є частиною JDK і дозволяє розробнику користуватися великим кількість структур даних з «коробки».

При його розробці були пред'явлені наступні вимоги:

* Повинен дозволяти працювати з різними типами даних однаково і з високим ступенем сумісності.
* Повинен мати високу продуктивність і ефективно реалізовувати фундаментальні структури даних такі, як зв'язний список, дерева, хеш-таблиці і динамічні масиви.
* Повинен дозволяти легко створювати власні колекції для специфічних завдань.