Отчет по практике №2

По ИЗВП

Выполнил студент К17.1

Шишелов Владимир

Листинг программы

public class Main {

public static class Item {

public String name;

public float price;

public Item(String name, float price) {

boolean priceLowerThan0 = price < 0; //Variable indicates whether price less than 0

if (priceLowerThan0) { //If yes - assign the price 0

this.name = name;

this.price = 0;

return;

}

this.name = name; //If not, write data entered by the user

this.price = price;

}

public void increasePrice(float percent) {

//Add a percent to price

float increasedPrice = this.price + this.price \* (0.01f \* percent);

//Return result

this.price = increasedPrice;

}

public void decreasePrice(float percent) {

//Subtract a percent of price

float decreasedPrice = this.price - this.price \* (0.01f \* percent);

boolean priceLowerThan0 = decreasedPrice < 0;

//If price lower than 0 set it by 0

if (priceLowerThan0) {

this.price = 0;

return;

}

this.price = decreasedPrice; //Return result

}

}

public static class Cart {

private Stack<Item> container = new Stack<Item>();

private float sumPrice = 0; //Sum of all basket

private int size; //Max count of items

public Cart(int size) {

this.size = size; //Initialize max count of items in cart

}

public void addItem(Item toAdd) {

//Variable indicates whether cart is crowded

boolean cartNotFull = this.container.size() != this.size;

//If cart is not crowded - add Item

if (cartNotFull) this.container.push(toAdd);

}

public void deleteItem() {

//Variable indicates whether cart is empty

boolean cartNotEmpty = this.container.size() != 0;

//If cart is not empty - pop item

if (cartNotEmpty) this.container.pop();

}

public void recalculateCartPrice() {

this.sumPrice = 0; //Resetting total cart value

Stack<Item> temp = (Stack<Item>) this.container.clone(); //Create a copy of cart to iterate it

int countOfItems = this.container.size(); //Find number of items in cart

for (int i = 0; i < countOfItems; i++) {

Item current = temp.pop(); //Take item out of basket and sum his price to total price

this.sumPrice += current.price;

}

}

public void increaseSum(float percent) {

Stack<Item> copyStack = (Stack<Item>) container.clone(); //Create a copy of cart to iterate it

//Create one more clone that will store the changed values

Stack<Item> tempStack = new Stack<Item>();

for (int i = 0; i < container.size(); i++) { //Iterate all cart items

Item current = copyStack.pop(); //Pop item from cart

current.increasePrice(percent); //Increasing price

tempStack.push(current);//Input item into stack with changed values

}

this.container = tempStack; //Rewrite cart values

}

public void decreaseSum(float percent) {

Stack<Item> copyStack = (Stack<Item>) container.clone(); //Create a copy of cart to iterate it

//Create one more clone that will store the changed values

Stack<Item> tempStack = new Stack<Item>();

for (int i = 0; i < container.size(); i++) { //Iterate all cart items

Item current = copyStack.pop();//Pop item from cart

current.decreasePrice(percent);//Decreasing price

tempStack.push(current);//Input item into stack with changed values

}

this.container = tempStack;//Rewrite cart values

}

public void showPrice() { //Method that shows cart price

System.out.println(this.sumPrice);

}

}

//---------------------------------------------2------------------------------------//

public static int sumOfPositiveArrayElements(int[] arr){

int sum =0; //Variable that saves sum of positive elements

for(int i = 0; i<arr.length; i++) {

if (arr[i] > 0) //If element more than 0 sum += element value

sum += arr[i];

}

return sum; //Return value

}

public static int multiplyElementsBetweenModMinAndModMaxElement(int[] arr){

int min = arr[0], indexMin = 0; //Array max element value and index

int max = arr[0], indexMax = 0; //Array min element value and index

for(int i = 0; i<arr.length; i++){

if(arr[i] > max){ //Search for max element

max = arr[i];

indexMax = i;

}

if(arr[i] < min){ //Search for min element

min = arr[i];

indexMin = i;

}

}

int start,end; //Temp Variables, need for definition start and end of loop

if(indexMax > indexMin){

start = indexMin;

end = indexMax;

}else{

start = indexMax;

end = indexMin;

}

int multiply = 1; //Variable that saves result of multiplying

for(int i = start; i < end; i++){

multiply \*= arr[i];

}

return multiply; //Return result

}

public static int binarySearch(int[] array, int searchNumber){

for (int left = 0; left < array.length; left++) {

//Pull out the array element value

int value = array[left];

//Moving by elements before pulled before

int i = left - 1;

for (; i >= 0; i--) {

//If value less than pulled - move the larger element next

if (value < array[i]) {

array[i + 1] = array[i];

} else {

//If value more than pulled - stop

break;

}

}

//Insert pulled element to the free space

array[i + 1] = value;

}

int indexOfSearchedNumber = Arrays.binarySearch(array, searchNumber); //Realize binary search

return indexOfSearchedNumber; //Return values

}

public static char[] sortWord(char[] wordToSort){

for(int i = 0; i < wordToSort.length - 1; i++){

for(int j = 0; j < wordToSort.length - i - 1; j++) {

if((int) wordToSort[j] > (int) wordToSort[j + 1]) {

char temp = wordToSort[j];

wordToSort[j] = wordToSort[j + 1];

wordToSort[j + 1] = temp;

}

}

}

return wordToSort;

}

public static void main(String[] args) {

//--------------------------2.1-------------------------------//

System.out.println("--------------------2.1-------------------");

Cart cart = new Cart(7); //Creating cart

Item bread = new Item("bread", 12); //Creating goods

Item coffee = new Item("coffee", 150.5f);

Item meat = new Item("meat", 115);

Item apple = new Item("apple", 17);

Item sugar = new Item("sugar", 32);

cart.addItem(bread); //Input goods into cart

cart.addItem(coffee);

cart.addItem(meat);

cart.addItem(apple);

cart.addItem(sugar);

cart.recalculateCartPrice(); //calculate total price of goods in cart

cart.showPrice();

cart.increaseSum(15); //Increasing price by 15%

cart.recalculateCartPrice(); //Recalculate total price

cart.showPrice();

cart.decreaseSum(50); //Decreasing price by 50%

cart.recalculateCartPrice(); //Recalculate total price

cart.showPrice();

//--------------------------2.2-------------------------------//

System.out.println("--------------------2.2-------------------");

/\*int[] testArray1 = new int[]{2,1,-3,4,5,6,7,8,9,-10,11};

System.out.println("Find sum of positive array elements: \n");

System.out.println(sumOfPositiveArrayElements(testArray1));

System.out.println("\n");

System.out.println("Search multiplying elements between min and max by module elements: \n");

System.out.println(multiplyElementsBetweenModMinAndModMaxElement(testArray1));

System.out.println("\n");\*/

int[] testArray1 = new int[]{2,1,-3,4,5,6,7,8,9,-10,11};

System.out.println(binarySearch(testArray1, 6));

char[] word = new char[]{'a','g','f','d'};

System.out.println(sortWord(word));

}



Рисунок 1 – Результат работы классов Item, Cart.

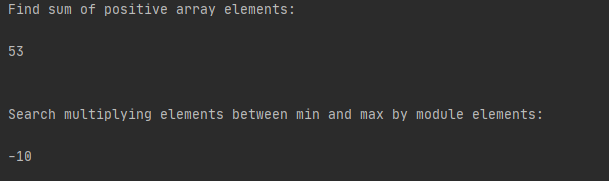


Рисунок 2 – Результат выполнения программы



Рисунок 3 – Результат работы программы

Выводы

По окончанию лабораторной работы были повторены и реализованы разные виды сортировок, бинарный поиск на языке программирования Java. Также были реализована система «Корзины» в интернет-магазине.