Министерство образования Республики Беларусь

Учреждение образования

«Брестский государственный технический университет»

Кафедра ИИТ

Лабораторная работа №4

По дисциплине: «Современные системы программирования»

Тема: «Работа со списками»

Выполнил:

Студент 4 курса

Группы ИИ-16 (2)

Пешко А.С.

Проверил:

Коренкович О. Г.

Брест 2021

**Цель работы:** освоить работу со списками (объектами классов TListBox и TComboBox).

**Ход работы**

**Задача:** cоздать простейший вариант программы для торговли , позволяющий осуществить:

- выбор товара и отказ от него (c возможностью поиска);

 - подсчет суммы;

 - формирование кассового чека.

Обеспечьте вывод чека во внешний текстовый файл. Обеспечьте при нажатии кнопки Найти на второй форме вывод сообщения «Товар не найден» вместо вывода цены, если товар отсутствует в списке.

**Код программы:**

public class GUIControllerMainForm {

private Shop shop;

private MainWindow view;

private Basket basket = new Basket();

public List<Commodity> getListCommodity() {

return shop.getListCommodity();

}

public GUIControllerMainForm(Shop shop) {

this.shop = shop;

}

public void execute() {

view = new MainWindow(getListCommodity());

//размещаем форму по центру экрана

view.setLocationRelativeTo(null);

//добавляем обработчик событий для кнопки "найти товар"

JButton jButtonFindCommodity = view.getJButtonFindCommodity();

jButtonFindCommodity.addActionListener(new ListenerForButtonFindCommodity());

//добавляем обработчик событий для кнопки добадения в корзину товара

JButton jButtonAddToBasket = view.getJButtonAddToBasket();

jButtonAddToBasket.addActionListener(new ListenerForButtonAddToBasket());

//добавляем обработчик событий для кнопки "просмотр корзины"

JButton jButtonShowBasket = view.getJButtonShowBasket();

jButtonShowBasket.addActionListener(new ListenerForButtonButtonShowBasket());

}

class ListenerForButtonButtonShowBasket implements ActionListener {

@Override

public void actionPerformed(ActionEvent e) {

new BasketListFrame(basket);

}

}

class ListenerForButtonFindCommodity implements ActionListener {

@Override

public void actionPerformed(ActionEvent e) {

String response = JOptionPane.showInputDialog(null,

"Type name :",

"Searching for commodity",

JOptionPane.QUESTION\_MESSAGE);

Optional<Commodity> foundCommodity = shop.getListCommodity().stream()

.filter(c -> c.getName().equalsIgnoreCase(response))

.findFirst();

if (foundCommodity.isPresent()) {

view.setJComboBoxSelectedCommodity(foundCommodity.get());

} else {

JOptionPane.showMessageDialog(null, "Товар не найден");

}

}

}

class ListenerForButtonAddToBasket implements ActionListener {

@Override

public void actionPerformed(ActionEvent e) {

int indexSelectCommodity = view.getJComboBoxWithListCommodity().getSelectedIndex();

if (indexSelectCommodity != -1) {

Commodity selectedCommodity = shop.listCommodity.get(indexSelectCommodity);

basket.add(selectedCommodity);

} else {

JOptionPane.showMessageDialog(null, "Товары отсутствуют в магазине!");

}

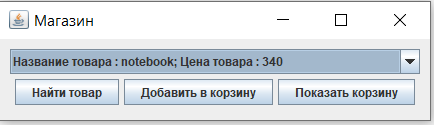
}

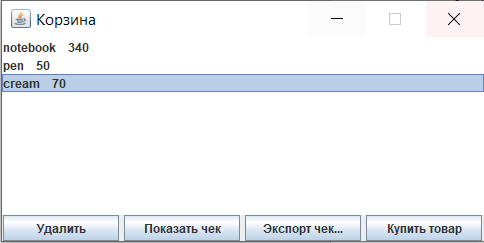
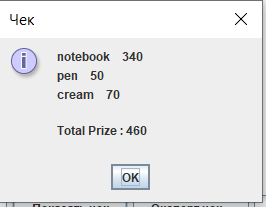
}

}

public class MainWindow extends JFrame {  
  
 private CommodityPanel commodityPanel;  
 private final Image shopIcon = new ImageIcon("../imageShop.png").getImage();  
 private JButton addToBasketButton;  
 private JButton findCommodityButton;  
 private JButton showBasketButton;  
 private JButton topUpMoney;  
 private JLabel moneyInPurseJLabel;  
 private JPanel lowCommodityPanel;  
 private JComboBox listCommodityComboBox;  
  
 public JLabel getLabelWithAmountMoney() {  
 return moneyInPurseJLabel;  
 }  
  
 public JButton getTopUpMoneyButton() {  
 return topUpMoney;  
 }  
  
 public JLabel getMoneyInPurseJLabel() {  
 return moneyInPurseJLabel;  
 }  
  
 public JButton getJButtonAddToBasket() {  
 return addToBasketButton;  
 }  
  
 public JButton getJButtonFindCommodity () {  
 return findCommodityButton;  
 }  
  
 public JButton getJButtonShowBasket () {  
 return showBasketButton;  
 }  
  
 public JComboBox getJComboBoxWithListCommodity() {  
 return listCommodityComboBox;  
 }  
  
 public void setJComboBoxWithListCommodity(List<Commodity> listCommodity) {  
 listCommodityComboBox = createComboBox(listCommodity);  
 }  
  
 public void setJComboBoxSelectedCommodity(Commodity commodity) {  
 listCommodityComboBox.getModel().setSelectedItem(commodity);  
 }  
  
 public MainWindow(List<Commodity> listCommodity) {  
 setLayout(new GridLayout(1, 2, 15, 5));  
 setTitle("Магазин");  
 setIconImage(shopIcon);  
 commodityPanel = new CommodityPanel(listCommodity);  
 add(commodityPanel);  
 setVisible(true);  
 setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);  
 pack();  
 }  
  
 class CommodityPanel extends JPanel {  
  
 public CommodityPanel(List<Commodity> listCommodity) {  
 setLayout(new BorderLayout());  
 setBorder(BorderFactory.createEmptyBorder(10, 10, 10, 10));  
 //создаём список, заполненный товарами  
 listCommodityComboBox = createComboBox(listCommodity);  
 lowCommodityPanel = new LowCommodityPanel();  
 add(listCommodityComboBox, BorderLayout.CENTER);  
 add(lowCommodityPanel, BorderLayout.SOUTH);  
 }  
  
 }  
  
 public JComboBox createComboBox(List<Commodity> commodities) {  
 String[] items = new String[commodities.size()];  
 for (int i = 0; i < commodities.size(); i++) {  
 items[i] = commodities.get(i).toString();  
 }  
 return (new JComboBox(items));  
 }  
  
 class LowCommodityPanel extends JPanel {  
  
 public LowCommodityPanel() {  
 setLayout(new FlowLayout(FlowLayout.CENTER));  
 addToBasketButton = new JButton("Добавить в корзину");  
 findCommodityButton = new JButton("Найти товар");  
 showBasketButton = new JButton("Показать корзину");  
 add(findCommodityButton);  
 add(addToBasketButton);  
 add(showBasketButton);  
 setVisible(true);  
 }  
 }  
  
}  
  
public class BasketListFrame extends JFrame {  
  
 private JPanel mainPanel = new JPanel();  
 private JButton deleteButton = new JButton("Удалить");  
 private JButton showCheckButton = new JButton("Показать чек");  
 private JButton exportCheckButton = new JButton("Экспорт чека в файл");  
 private JButton buyCommodityButton = new JButton("Купить товар");  
 private DefaultListModel<String> model = new DefaultListModel<>();  
 private JList<String> choice;  
 private Basket basket;  
  
 public BasketListFrame(Basket basket) {  
 mainPanel.setLayout(new BorderLayout());  
 add(mainPanel);  
 this.basket = basket;  
 updateList();  
 mainPanel.add(choice, BorderLayout.CENTER);  
 JPanel grid = new JPanel(new GridLayout(1, 3, 5, 0));  
 grid.add(deleteButton);  
 grid.add(showCheckButton);  
 grid.add(exportCheckButton);  
 grid.add(buyCommodityButton);  
 mainPanel.add(grid, BorderLayout.SOUTH);  
 deleteButton.addActionListener(new DeleteActionListener());  
 showCheckButton.addActionListener(new ShowCheckActionListener());  
 buyCommodityButton.addActionListener(new ListenerForButtonBuyCommodity());  
 exportCheckButton.addActionListener(new ListenerForButtonExportCheck());  
 setSize(500, 250);  
 setResizable(false);  
 setTitle("Корзина");  
 setLocationRelativeTo(null);  
 setVisible(true);  
 }  
  
 public void updateList() {  
 try {  
 for (Commodity commodity : basket.getListCommodity()) {  
 model.addElement(commodity.getName() + " " + commodity.getPrice());  
 }  
 choice = new JList<>(model);  
 } catch (Exception e1) {  
 e1.printStackTrace();  
 }  
 }  
  
 public class DeleteActionListener implements ActionListener {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 try {  
 DefaultListModel<String> model = (DefaultListModel<String>) choice.getModel();  
 basket.getListCommodity().remove(choice.getSelectedIndex());  
 model.remove(choice.getSelectedIndex());  
 } catch (Exception e1) {  
 e1.printStackTrace();  
 }  
 }  
 }  
  
 public class ShowCheckActionListener implements ActionListener {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String totalItemList = getTotalItemList();  
 JOptionPane.showMessageDialog(null, totalItemList, “Чек”, 1);  
 }  
 }  
  
 class ListenerForButtonBuyCommodity implements ActionListener {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 try {  
 DefaultListModel<String> model = (DefaultListModel<String>) choice.getModel();  
 basket.getListCommodity().remove(choice.getSelectedIndex());  
 model.remove(choice.getSelectedIndex());  
 } catch (Exception e1) {  
 e1.printStackTrace();  
 }  
 }  
 }  
  
 class ListenerForButtonExportCheck implements ActionListener {  
 @Override  
 public void actionPerformed(ActionEvent e) {  
 String totalItemList = getTotalItemList();  
 JFileChooser chooser = new JFileChooser();  
 chooser.setCurrentDirectory(new File(".\\"));  
 int retrival = chooser.showSaveDialog(null);  
 if (retrival == JFileChooser.APPROVE\_OPTION) {  
 try (FileWriter fw = new FileWriter(chooser.getSelectedFile() + ".txt")) {  
 fw.write(totalItemList);  
 } catch (IOException ioException) {  
 ioException.printStackTrace();  
 }  
 }  
 }  
 }  
  
 private String getTotalItemList() {  
 int totalPrize = 0;  
 StringBuilder totalItemList = new StringBuilder();  
  
 for (Commodity commodity : basket.getListCommodity()) {  
 totalPrize += commodity.getPrice();  
 totalItemList.append(commodity.getName()).append(" ").append(commodity.getPrice()).append("\n");  
 }  
 totalItemList.append("\n").append("Total Prize : ").append(totalPrize).append("\n\n");  
 return totalItemList.toString();  
 }  
}

**Результат работы прогаммы:**



**Вывод:** освоили работу со списками (объектами классов TListBox и TComboBox).