Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

**«Пермский национальный исследовательский**

**политехнический университет»**

Электротехнический факультет

Кафедра Микропроцессорные средства автоматизации

**Программирование и автоматизация**

**(программирование на Java)**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ СТУДЕНТАМ**

по выполнению самостоятельной/курсовой работы

## Общее задание

Требуется разработать два Java –приложения. В качестве среды разработки используем NetBeans IDE. Каждое приложение должно работать со списком объектов. Вид объекта выбирается в соответствии с вариантом задания, при этом нужно учесть что для описания объекта используется два класса – класс потомок (суперкласс) и класс наследник. Используя интерфейс приложения нужно обеспечить чтение и запись членов объекта, а также выполнение их методов.

Первое из приложений использует для чтения/записи членов объектов отдельные поля (используем класс jTextField), размещенные на форме. Второе приложение использует для чтения/записи объектов таблицу (используем класс JTable). При этом используется принцип «модель-вид-контроллер».

Для выполнения индивидуального задания рекомендуется разобрать примеры проектов, которые прилагаются к данному пособию.

## Условие задания

Задачи основаны на задачах из Сборника задач по программированию (Одесса: ОНАС им. А.С. Попова, 2011). В качестве примера разберем задачу № 30 из темы 11 (Элементы объектно-ориентированного программирования). Задача состоит из двух частей Таблица 11.2 (11.1. Cоздание данных типа «класс» - Средний уровень), № вар.30 и Таблица 11.5 (11.2. Наследование классов,- Средний уровень), № вар.30.

Разберем условие индивидуального задания №30. Общие условия задачи для темы «Cоздание данных типа «класс»» для среднего уровня в задачнике в задачнике формируются так:

Задание: создать класс с полями, указанными в индивидуальном задании (табл. 11.2, столб 2). Реализовать в классе методы:

* конструктор по умолчанию;
* конструктор перезагрузки с параметрами;
* деструктор для освобождения памяти (с сообщением об уничтожении объекта);
* функции обработки данных (1 и 2), указанные в индивидуальном задании

Сразу отметим, что конструктор перезагрузки и деструктор нам не понадобиться, так же мы не будем выводить результаты в компонент Memo.

Для темы «Наследование классов» задание формулируется так:

Для класса, созданного в предыдущем задании (по вариантам табл. 11.2) создать класс-потомок с полями, указанными в индивидуальном задании (табл. 11.5, столб 2). Реализовать в классе-потомке методы:

* - конструктор;
* - функцию обработки данных, указанную в индивидуальном задании (табл.11.5, столб 3);
* - функцию формирования строки информации об объекте.

Создать проект для демонстрации работы: ввод и вывод информации об объектах: классе-родителе и классе-потомке. Строку информации об объекте, мы тоже делать не будем (не очень понятно, что имел в виду автор задачника).

Таким образом, для реализации как в базовом так и в наследуемом классе остается только один пункт, необходимо реализовать:

* - функцию обработки данных, указанную в индивидуальном задании;

Теперь приведем формулировку варианта индивидуального задания №30 из таблицы 11.2:

|  |  |  |
| --- | --- | --- |
| **Класс-родитель и его поля** | **Функция-метод 1 обработки данных** | **Функция-метод 2 обработки данных** |
| Работник: фамилия, должность, оклад | Увеличить оклад на15% (каждому работнику) | Работникам, у которых фамилия начинается с сочетания букв «Иван»,  присвоить должность «инженер» |

И из таблицы 11.5:

|  |  |  |
| --- | --- | --- |
| **Класс-родитель и его поля (из табл. 11.2)** | **Класс-потомок и его поля (поля класса-родителя выделены курсивом)** | **Функция-метод обработки данных объекта класса-потомка** |
| Работник: фамилия, должность, оклад | Работник предприятия:  фамилия, должность, оклад, *рейтинг* (в 100-бальной системе) | Увеличить оклад работников на 20%, если их рейтинг от 60 до 75 балов, на 40%, если их рейтинг  от 75 до 90 балов, на60%, если их рейтинг от 90 до 100 балов. |

Нам нужно будет написать два класса (базовый класс и его потомок) , для представления понятия «Работник». Кроме этого нам потребуется класс, представляющий форму с элементами управления. При помощи этой формы мы будем демонстрировать работу двух наших написанных классов.

Обратите внимание, что хотя в задании говориться о работнике в единственном числе, нам потребуется иметь дело со списком экземпляров нашего класса (объектами «Работник»).

Таким образом, задание №30 (и все остальные задания, также) должно предусматривать следующие шаги:

* Создание базового класса;
* Создание класса потомка;
* Создание класса формы, который должен содержать элементы интерфейса для демонстрации работы базового класса и класса потомка.
* Разработка списка экземпляров нашего класса (и методов работы со списком)
* Разработка методов, обеспечивающих интерфейс нашего класса и главной формы

## Создание проекта

Поскольку разработка Java в среде IDE всегда выполняется в рамках проектов, сначала необходимо создать новый проект , в котором будут сохраняться исходные файлы и другие файлы проекта. Проект среды IDE представляет собой группу исходных файлов Java и связанные с ними метаданные, включая файлы свойств проекта, сценарий сборки Ant, управляющий параметрами сборки и выполнения, а также файл project.xml.

Итак, расписываю по шагам:

1. Запускаем оболочку NetBeans IDE;
2. Выбираем команду "***Файл***" > "***Новый проект***". Также можно щелкнуть значок "***Новый проект***" на панели инструментов среды IDE.
3. Введите  в поле "***Имя проекта***" название приложения, которое вы желаете (я задал название по номеру варианта Example\_11\_30) и укажите местоположение проекта.
4. Выбираем категорию "***Java***" и вид проекта "***Приложение Java*** ", нажимаем "***Далее***"
5. Не устанавливайте флажок "***Использовать отдельную папку для хранения библиотек***".
6. Очистите флажок "***Создать главный класс***".
7. Нажмите кнопку "***Готово***".

Среда IDE создаст в системе папку с заданным именем по указанному пути. Эта папка содержит все файлы, связанные с проектом, включая сценарий Ant, папки для хранения исходных файлов и тестов, а также папку с метаданными проекта. Для просмотра структуры проекта используйте окно "***Файлы***" в среде IDE.

## Создание базового класса.

Сначала создадим в проекте базовый класс для представления понятия «Работник». Чтобы создать новый класс и задать его содержимое в соответствии с заданием (см. таблицу 11.2) выполните действия, описанные ниже:

1. В окне 'Проекты' щелкните правой кнопкой мыши узел с созданным проектом и выберите ***"Новый"*** >"***Класс Java*** "
2. Задайте имя класса (я задал имя Class1)
3. Нажмите кнопку "***Готово***". Будет создан новый класс с заданным именем;
4. Теперь редактируем текст созданного класса. В соответствии с заданием нам нужно создать его поля и два метода обработки данных. Двойной щелчок по имени класса в дереве проектов приводит к тому, что в окне редактора исходного кода появляется закладка со сгенерированным исходным кодом нашего класса. Для осуществления классом какой-либо деятельности следует дописать свой собственный код.
5. Редактируя исходный код , создаем поля класса:

* два поля строкового типа (фамилия, должность). одно поле целого типа (оклад)

этот код будет выглядеть так:

public class Class1 {

public String name, position;

public int salary;

Так как нам нужно будет обеспечить доступ ко всем членам и методам класса, мы должны использовать модификатор доступа public. Это означает, что мы можем использовать эти члены и методы из любого другого класса.

1. Создаем метод 1 обработки данных.

public void incSalary() {

salary = (int) (salary\*1.15);

};

Имя метода –incSalary, что означает «увеличить оклад». Слово void означает, что наш метод ничего не будет возвращать. Метод, как и требуется в задании, увеличивает оклад на 15%.

1. Создаем метод 2 обработки данных.

public void position2Engineer() {

position = "Инженер";

};

Имя метода – position2Engieer, что означает присвоить должность «инженер» (должность в инженер). Метод крайне примитивен и просто присваивает члену класса position значение «Инженер» (проверку фамилии на «иван» будем делать в главной форме).

Согласно принятым в сообществе Java соглашениям, есть ряд правил, которых желательно придерживаться при написании программы

|  |  |
| --- | --- |
| Классы | Имя класса начинается с большой буквы, если в имени несколько слов, каждое слово пишется с заглавной буквы слитно |
| Переменные | Переменные начинаются со строчной первой буквы, если в имени несколько слов, каждое следующее слово пишется с заглавной буквы слитно.  Имена переменных должны быть короткими, но со смыслом. Переменных состоящих из одного символа следует избегать, за исключением временных (одноразовых) переменных. |
| Методы | Методы начинаются со строчной первой буквы, если в имени несколько слов, каждое следующее слово пишется с заглавной буквы слитно. Методы должны быть глаголами. |
| Константы | Константы должны состоять из заглавных символов, если в имени несколько слов, каждое следующее слово отделяется от предыдущего символом подчеркивания "\_". |

|  |  |
| --- | --- |
|  |  |

## Создание класса потомка.

Теперь должен быть создан класс, расширяющий понятие «Работник» (см. таблицу 11.5).

1. Создаем новый класс, т.е. повторяем пункты 1-3 из предыдущего раздела. Имя класса задается как Class2. Редактируем сгенерированный исходный код класса Class2.
2. Делаем его потомком класса Class1, для этого в описании класса добавляем указание на предка

public class **Class2** extends Class1 {

1. Создаем член класса рейтинг

public int rating;

1. Создаем метод обработки данных класса-потомка. Он будет называться – chngSalary - Изменить оклад . Оклад изменяем в зависимости от рейтинга работника.

public void **chngSalary**() {

if (rating >= 60 && rating <75 ) {

salary = (int) (salary\*1.2);

}

else if (rating >= 75 && rating <90 ) {

salary = (int) (salary\*1.4);

}

else if (rating >= 90 && rating <=100 ) {

salary = (int) (salary\*1.6);

}

Теперь, когда классы созданы, нужно создать форму с элементами управления, при помощи которой мы будем демонстрировать работу двух наших написанных классов.

## Создание класса главной формы.

Для разработки формы с элементами управления можно воспользоваться классами, находящимися в пакете swing. В этом случае вам придется написать довольно большой объем кода, при помощи которого вы будете размещать элементы управления на форме и задавать реакцию на действия с этими элементами (например, действие при нажатии на кнопку). Есть возможность упростить разработку если часть работы по написанию кода возложить на специальный инструмент для визуальной разработки интерфейса. Это средство GUI Builder включенное в среду NetBeans IDE позволяет разработчикам проектировать и составлять интерфейсы к программам посредством перетаскивания элементов в рабочую область т.е. освобождает разработчиков от необходимости изучения особенностей использования Swing.

Для разработки формы с использованием GUI Builder выполните следующие действия:

1. В окне "***Проекты***" щелкните правой кнопкой мыши узел нашего проекта и выберите "***Новый***" > "***Форма JFrame***".
2. Введите придуманное вами название формы в поле имени класса.
3. Нажмите кнопку "***Готово***". Среда IDE создаст новую форму с заданным вами именем и класс с таким же именем. В редакторе открывается закладка с этим именем. Теперь можно приступить к разработке пользовательского интерфейса приложения.
4. Для визуального проектирования нажимаем кнопку "***Проект***" и переходим в визуальный редактор.
5. Устанавливаем желаемые размеры формы
6. При помощи окна "С***войства***" устанавливаем заголовок окна. В нашем случае свойство title устанавливаем как «Задание 11 30»

Созданный нами класс должен содержать элементы интерфейса для демонстрации работы и методы для описания поведения этих элементов. Кроме этого нам потребуется создать список объектов (работников) и методы для работы с этим списком.

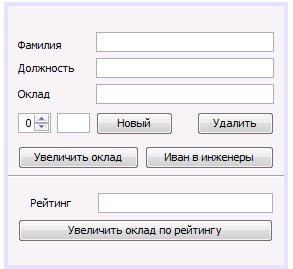
Перечислим элементы интерфейса, которые нам понадобятся для демонстрации работы двух написанных нами классов. Итак, нам потребуются:

* Поле (JtextField) для ввода/вывода фамилии работника (Class1.name)
* Поле (JtextField) для ввода/вывода должности работника (Class1.position)
* Поле (JtextField) для ввода/вывода оклада работника (Class1. Salary)
* Поле (JtextField) для ввода/вывода рейтинга работника (Class2.rating)
* Кнопка (Jbuttton) для создания нового объекта (экземпляра работника)
* Кнопка (Jbuttton) для удаления объекта
* Кнопка (Jbuttton) для выполнения метода 1 обработки данных класса-родителя (Class1.incSalary)
* Кнопка (Jbuttton) для выполнения метода 2 обработки данных класса-родителя – (Class1. position2Engieer)
* Кнопка (Jbuttton) для выполнения метода обработки данных класса-потомка – (Class2. chngSalary)
* компонент выбора числа или счетчик JSpinner предназначенный для выбора текущего объекта из списка.
* Поле для отображения количества элементов списке.

Кроме этого на форме присутствуют элементы для улучшения внешнего вида, такие как поясняющие надписи (JLabel) и разделяющая линия jSeparator

1. Размещаем на форме все элементы интерфейса.
   1. В окне "***Палитра***" находим компонент Метка и перетаскиваем его на форму. В окне "***Свойства***" заполняем text – Фамилия
   2. Компонент «Текстовое поле» размещаем рядом с меткой «Фамилия». Переименуем это поле, В окне "***Свойства***"переключимся на закладку «Код» и устанавливаем «Имя переменной» JtextFieldName.
   3. Компонент Метка перетаскиваем на форму и задаем text – Должность
   4. Компонент «Текстовое поле» размещаем рядом с меткой «Должность», переименуем его как JtextFieldPosition.
   5. Компонент Метка перетаскиваем на форму и задаем text –Оклад
   6. Компонент «Текстовое поле» размещаем рядом с меткой «Оклад», переименуем его как JtextFieldSalary.
   7. Размещаем на форме компонент JSpinner, переименуем его как JSpinnerCrn (от current - текущий)
   8. Размещаем компонент «Текстовое поле», переименуем его в JTextFieldCount.
   9. Находим компонент Кнопка и перетаскиваем его на форму. Устанавливаем текст на кнопке "***Новый***" и меняем имя на JButtonNewPers (NewPers имеется в виду Новый человек)
   10. Компонент Кнопка перетаскиваем на форму и задаем text –Удалить, Имя переменной – JbuttonDelPers
   11. Кнопку перетаскиваем на форму и задаем text –« Увеличить оклад», Имя переменной - JButtonIncSalary
   12. Кнопку перетаскиваем на форму и задаем text –«Иван в инженеры», Имя переменной - JButtonPosition2Engieer
   13. Помещаем на форму разделяющую линию jSeparator (Она зрительно отделит то что относится к классу-родителю от того что относится к классу-потомку)
   14. Перетаскиваем на форму метку, задаем text –Рейтинг
   15. Компонент «Текстовое поле» перетаскиваем на форму, переименуем его как JtextFieldRating
   16. Компонент Кнопка перетаскиваем на форму и задаем text –«Увеличить оклад по рейтингу», Имя переменной – JbuttonChngRating.

Внешний вид формы, полученной в результате размещения указанных компонент, представлен на рисунке.



## Создание класса главной формы.

В результате размещения элементов на форме мы имеем пустой каркас приложения, т.е. внешний вид полностью соответствует тому, что мы хотели, но при нажатии на кнопки ничего не происходит. Теперь требуется написать код, который будет делать все что нам требуется.

Обсудим общую идею нашего приложения.

Все объекты, соответствующие работникам будут создаваться при нажатии на кнопку "***Новый***" (JButtonNewPers) и помещаться в список. Количество элементов в списке отображается в поле JTextFieldCount. Поля текущего объекта будут отображаться в полях формы JtextFieldName, JtextFieldPosition, JtextFieldSalary и JtextFieldRating. При помощи этих же элементов мы будем присваивать значение членам класса. Т.е. все объекты хранятся в списке и при помощи текстовых полей формы мы получаем двусторонний доступ к членам текущего объекта. Текущий объект выбирается из списка при помощи счетчика JSpinnerCrn. Удаляется текущий объект из списка при нажатии на кнопку "***Удалить***" (JbuttonDelPers).

Все методы классов (Class1, Class2) выполняются при нажатии на кнопки "***Увеличить оклад***" (JButtonIncSalary), "***Иван в инженеры***" , (JButtonPosition2Engieer) и "***Увеличить оклад по рейтингу***", (JButtonChngRating). Все три метода выполняются для каждого объекта из списка.

Нажимаем на кнопку "***Источник***" в редакторе, т.е. переключаемся в режим редактирования кода. Область кода, которая была автоматически создана, будет выделена серыми областями, которые называются защитными блоками. Код в защищенных областях изменить невозможно . Редактировать код класса можно только для кода на белом фоне окна редактора. Теперь начинаем программировать:

1. Импортировать имя класса ArrayList из пакета java.util, это значит, что мы получаем возможность обращаться к классу ArrayList напрямую. Класс ArrayList это класс представляющий список.

import java.util.ArrayList;

1. Объявляем член класса типа список.

ArrayList<Class2> personList;

Наш список будет иметь название personList. Название класса Class2 в угловых скобках означает, что список будет содержать объекты типа Class2 (Класс-потомок).

Кроме этого объявим переменную currentInd, которая будет хранить что номер текущего объекта из списка personList (отметим, что нумерация объектов в списке начинается с 0).

Программируем конструктор класса, т.е. проектируем действия, которые наша форма совершает в момент своего создания. Наш конструктор должен создать экземпляр списка personList

public **Example11\_30JFrame**() {

initComponents();

personList = new ArrayList <Class2>();

currentInd = -1;

}

Первая строка конструктора была создана автоматически, в этой строке вызывается функция, которая создает все элементы, расположенные на форме. Мы дописываем две строки. В первой создается экземпляр списка personList, во второй мы указываем, что номера текущего объекта из списка еще нет (ведь список еще пустой).

## Методы, обеспечивающие интерфейс класса «Работник» и главной формы

Теперь нам потребуются два метода, которые будут осуществлять копирование данных из текущего объекта списка в поля формы и обратно. Сначала покажем метод копирования данных из объекта в поля формы, т.е. показа текущего экземпляра класса. Этот метод назван getPerson.

protected void **getPerson**(){

if (currentInd >= 0) {

jTextFieldName.setText(personList.get(currentInd).name);

jTextFieldPosit.setText(personList.get(currentInd).position);

jFormattedTextFieldSalary.setText("" + personList.get(currentInd).salary);

jFormattedTextFieldRating.setText("" + personList.get(currentInd).rating);

}

}

Копирование осуществляется, если в списке существует текущий объект, т.е. номер текущего объекта (currentInd) больше или равен нулю. Текущий элемент списка мы каждый раз находим при помощи строки personList.get(currentInd), т.е. при помощи метода get класса personList. Текст в текстовом поле устанавливаем при помощи метода setTex. Так как оклад и рейтинг имеют тип int, то необходимо сделать преобразование к строковому типу. Это осуществляется тем, что сначала указываем пустую строку ( ""), а потом добавляем к ней значение поля объекта в этом случае Java выполнит преобразование автоматически.

Метод, который решает обратную задачу, т.е. записывает в поля текущего объекта данные из текстовых полей формы, называется **setPerson:**

protected void **setPerson**(){

if ((currentInd >= 0) && (jTextFieldName.getText() != null)) {

personList.get(currentInd).name = jTextFieldName.getText();

personList.get(currentInd).position = jTextFieldPosit.getText();

try {

personList.get(currentInd).salary = new Integer(jFormattedTextFieldSalary.getText());

} catch (Exception e) {

personList.get(currentInd).salary = 0;

jFormattedTextFieldSalary.setText(null);

}

try {

personList.get(currentInd).rating = new Integer(jFormattedTextFieldRating.getText());

} catch (Exception e) {

personList.get(currentInd).rating = 0;

jFormattedTextFieldRating.setText(null);

}

}

}

Копирование в текущий объект происходит, если этот объект существует, т.е. номер текущего объекта (currentInd) больше или равен нулю и если в поле «Фамилия» на форме что-то записано (jTextFieldName.getText() != null))

Преобразование строки из текстового поля в целочисленные члены класса происходит с использованием конструктора, так называемого оболочечного класса (мы используем оболочечный класс Integer).

В примере была использована конструкция try-catch. Это необходимо для обработки ошибки, в случае, если строка содержит иные символы кроме чисел.

## Методы для работы со списком работников

Теперь запрограммируем реакцию на нажатие кнопок, связанных с созданием/удалением объектов в списке. Для того чтобы привязать к кнопке реакцию на ее нажатие нужно просто сделать на ней двойной щелчок (в окне "***Проекты***"). Будет сгенерирован метод класса jButtonNewPersActionPerformed (для кнопки jButtonNewPers). Кроме этого в защищенной области в приватном методе initComponents появится код, связывающий сгенерированный метод и саму кнопку. Также имя этого метода появится в окне свойств кнопки на закладке ***"События***" (событие ActionPerformed).

private void **jButtonNewPersActionPerformed**(java.awt.event.ActionEvent evt) {

}

Работает это так, когда пользователь нажмет кнопку, кнопка создаст объект класса ActionEvent и вызовет jButtonNewPersActionPerformed (event).

Сгенерированный метод пока пустой, чтобы приложение выполняло какое-нибудь действие в ответ на нажатие на кнопку необходимо дописать свой код. Для кнопки jButtonNewPers напишем следующее:

private void **jButtonNewPersActionPerformed**(java.awt.event.ActionEvent evt) {

setPerson();

personList.add(new Class2());

currentInd = personList.size()-1;

getPerson();

jTextFieldCount.setText(""+(currentInd+1));

jSpinnerCrn.setValue(currentInd);

}

Здесь выполняется следующее:

* Выполняется метод setPerson(), чтобы записать в поля текущего объекта списка данные из текстовых полей формы.
* В список personList добавляется вновь созданный экземпляр класса Class2
* Присвоим значение переменной currentInd, которая хранит номер текущего объекта, значение делаем равным номеру последнего объекта списка.
* Копируем поля из созданного объекта на форму. Текстовые поля будут очищены, а в численных будет записан 0.
* Выводим количество объектов списке
* В счетчике jSpinnerCrn показываем номер текущего объекта в списке.

Аналогично создаем обработчик для кнопки jButtonDelPers

private void **jButtonDelPersActionPerformed**(java.awt.event.ActionEvent evt) {

personList.remove(currentInd);

currentInd = personList.size()-1;

if (currentInd == personList.size()) {

currentInd--;

jSpinnerCrn.setValue(currentInd);

} else {

getPerson();

}

jTextFieldCount.setText("" + personList.size());

}

Процедура по разному ведет себя если удаляется не последний и последний элемент списка.

Для управления списком нужно также запрограммировать реакцию на изменение в счетчике JSpinnerCrn, счетчик указывает номер текущего элемента в списке. Нужно в окне свойств кнопки на закладке "***События***" щелкнуть на событие StateChanged. В сгенерированный шаблон добавляем свой код.

private void **jSpinnerCrnStateChanged**(javax.swing.event.ChangeEvent evt) {

setPerson();

currentInd = (int) jSpinnerCrn.getValue();

if (currentInd >= personList.size()) {

currentInd = personList.size()-1;

jSpinnerCrn.setValue(currentInd);

} else if (currentInd < 0) {

currentInd = 0;

jSpinnerCrn.setValue(currentInd);

}

getPerson();

}

Код работает следующим образом:

* Выполняется метод setPerson(), чтобы записать в поля текущего объекта списка данные из текстовых полей формы.
* Присвоим новое значение переменной currentInd, которая хранит номер текущего объекта, значение. Значение берется из компонента jSpinnerCrn
* Проверяем значение переменной currentInd на ее допустимость:
  + номер текущего объекта не может быть больше или равен количеству элементов в списке
  + номер текущего объекта не может быть меньше нуля
* Выполняется метод getPerson(), чтобы скопировать данных из текущего объекта списка в поля формы.

Все механизмы по работе со списком экземпляров класса Class2 теперь готовы, осталось реализовать код, при помощи которого мы покажем выполнение методов этих экземпляров.

## Реализация выполнения методов обработки данных Class1/ Class2

Код, обеспечивающий выполнение функции-метода 1 обработки данных класса-родителя связываем с обработчиком события для кнопки "***Увеличить оклад***" ( JButtonIncSalary).

private void **jButtonIncSalaryActionPerformed**(java.awt.event.ActionEvent evt) {

setPerson();

for (int i = 0; i < personList.size(); i++) {

personList.get(i).incSalary();

}

getPerson();

}

Здесь выполняется следующее:

* Выполняется метод setPerson для копирования в поля текущего объекта данных из текстовых полей формы.
* В цикле для каждого элемента списка выполняем метод incSalary (Увеличить оклад)
* Выполняется метод getPerson, чтобы показать измененный оклад

Выполнение функции-метода 2 обработки данных класса-родителя связываем с обработчиком события для кнопки ***"Иван в инженеры***" (JButtonPosition2Engieer)

private void **jButtonPosition2EngineerActionPerformed**(java.awt.event.ActionEvent evt) {

for (int i = 0; i < personList.size(); i++) {

if (personList.get(i).name.toUpperCase().startsWith("ИВАН")) {

personList.get(i).position2Engieer();

}

}

getPerson();

}

Особенностью здесь является то, что вызов метода position2Engieer выполняется только тогда, когда фамилия работника начинается с сочетания букв «иван». Чтобы результат не зависел от регистра написания фамилии, преобразуем поле name к верхнему регистру.

Функция-метод обработки данных объекта класса-потомка выполняется при помощи обработчика события для кнопки "***Увеличить оклад по рейтингу***" (JbuttonChngRating). В цикле для каждого элемента списка выполняем метод chngSalary (Изменить оклад работника в зависимости от его рейтинга)

private void **jButtonChngRatingActionPerformed**(java.awt.event.ActionEvent evt) {

setPerson();

for (int i = 0; i < personList.size(); i++) {

personList.get(i).chngSalary();

}

getPerson();

}

## Тестирование приложения

Приведем последовательность действий по тестированию спроектированного приложения.

E:\Univer\Программирование и основы алгоритмизации Java\Методичка к лабораторным работам\Самостоятельная работа Java\вып-ть.bmpЧтобы запустить приложение, нажмите по проекту правой кнопкой мыши и выберите "***Запустить проект*** " из контекстного меню или нажмите на кнопку

Создаем и помещаем в список последовательно три экземпляра «Работник» (Class2), т.е . нажимаем на кнопку "***Работник***", после чего заполняем все поля на форме. Поля были заполнены так:

1й работник

* Фамилия - Иванов
* Должность - Техник
* Оклад- 10000
* Рейтинг - 70

2й работник

* Фамилия - Петров
* Должность - Техник
* Оклад- 10000
* Рейтинг – 80

3й работник

* Фамилия - Сидоров
* Должность - Техник
* Оклад- 10000
* Рейтинг - 90

Просмотреть все поля всех объектов можно пролистывая список при помощи счетчика jSpinnerCrn.

Теперь протестируем методы спроектированного класса.

Метод incSalary – нажимаем на кнопку " ***Увеличить оклад***" - сразу после этого оклад у всех работников увеличится на15%, т.е. мы увидим что оклад после применения метода incSalary будет 11500.

Метод position2Engieer– нажимаем на кнопку "***Иван в инженеры***". Метод position2Engieer будет выполнен для 1-го работника с фамилией Иванов. После выполнения position2Engieer в поле «Должность» будет записано «Инженер»

Метод chngSalary – нажимаем на кнопку ***"Увеличить оклад по рейтингу".*** Метод chngSalary будет выполнен для всех объектов из списка. Оклад будет изменен в зависимости от поля «Рейтинг». В результате объекты в списке будут иметь следующий вид.

1й работник

* Фамилия - Иванов
* Должность - Инженер
* Оклад- 13800
* Рейтинг - 70

2й работник

* Фамилия - Петров
* Должность - Техник
* Оклад- 16100
* Рейтинг – 80

3й работник

* Фамилия - Сидоров
* Должность - Техник
* Оклад- 18400
* Рейтинг - 90

## Этап 2. Использование таблицы (JTable)

В существующем приложении для отображения или изменения полей объектов, помещенных в список, используются текстовые поля, при этом пользователю в каждый момент доступен только один объект (мы называем его текущим). Удобно иметь возможность работы сразу с несколькими объектами, такую возможность предоставляет компонент JTable.

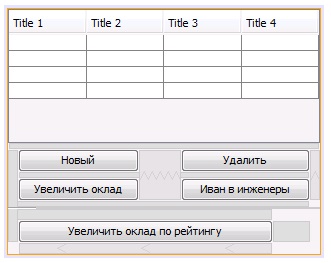
Разработчики *JTable* создали его на основе принципа «модель-вид-контроллер». Другими словами, это разделило *JTable* на части таким образом, что одна часть отвечает за способы отображения информации, другая за внешний вид, третья за получение данных.

Для JTable данные поставляются классом, который называется модель (TableModel). TableModel — это интерфейс, который мы должны реализовать сами в своей собственной модели для JTable. Создадим свою собственную TableModel для JTable.

Задача следующая. У нас имеется список экземпляров класса «Работник» (Class2). Необходимо отобразить (а также обеспечить изменение полей и выполнение методов) данный список в JTable при помощи TableModel.

Итак последовательность действий следующая:

1. Создаем новое приложение на основе существующего;
2. Удаляем все поля, через которые происходило взаимодействие с полями класса (JtextFieldName, JtextFieldPosition, JtextFieldSalary) и обозначающие их метки.
3. Удаляем компонент JSpinner. Теперь во фрагментах текста, которые ссылаются на удаленные компоненты, будут выявлены ошибки.
4. Удаляем все методы связанные с удаленными компонентами, в частности getPerson, setPerson..
5. Удаляем все ссылки на удаленные методы – теперь ошибок в программе быть не должно
6. Размещаем на форме компонент JTable, так как это показано на рисунке. (Имя таблицы компонента будет JTable1). Теперь нужно будет написать код, определяющий поведение таблицы, т.е. как информация об объектах отображается в строках таблицы и как объекты получают данные из таблицы.



1. К определению поля personList добавляем модификатор static, это нужно для того чтобы можно обращаться к статическому полю (в нашем случае personList) используя имя класса (а не объекта).

static ArrayList<Class2> *personList*;

Обращение к полю personList потребуется , чтобы модель для таблицы имела доступ к списку объектов.

1. К списку импортированных классов добавляем DefaultTableModel

import javax.swing.table.DefaultTableModel;

1. Определим класс MyTableModel, являющийся наследником DefaultTableModel

class **MyTableModel** extends DefaultTableModel {

};

Теперь нужно дописать все нужные нам методы MyTableModel. Эти методы были изначально определены в родительском классе DefaultTableModel.

1. Определяем конструктор класса, при создании экземпляра MyTableModel добавим в таблицу столбцы, при этом заполним заголовок для каждого столбца.

// Добавим столбцы и заполним их шапку

public **MyTableModel**() {

addColumn("Фамилия");

addColumn("Должность");

addColumn("Оклад");

addColumn("Рейтинг");

}

1. Определим метод getRowCount(), который задает количество строк таблицы. В нашем случае количество строк равно количеству элементов в списке personList.

//кол строк

public int **getRowCount**() {

return Example11\_30JFrame.*personList*.size();

}

1. Определим метод Object getValueAt(int row, int column), который вернет содержимое ячейки таблицы для каждой заданной строки и столбца. Т.е. Метод getValueAt отвечает за то, какие данные в каких ячейках JTable будут показываться. Данные для отображения извлекаем из полей объектов, хранящихся в списке personList.

//данные таблицы

public Object **getValueAt**(int row, int column){

switch (column) {

case 0:

return Example11\_30JFrame.*personList*.get(row).name;

case 1:

return Example11\_30JFrame.*personList*.get(row).position;

case 2:

return Example11\_30JFrame.*personList*.get(row).salary;

case 3:

return Example11\_30JFrame.*personList*.get(row).rating;

}

return "";

}

1. Определим метод, обеспечивающий запись данных из таблицы в объект.

// Изменение элемента

public void **setValueAt**(Object o, int i, int i1) {

switch (i1) {

case 0: {

Example11\_30JFrame.*personList*.get(i).name=(String)o;

break; }

case 1: {

Example11\_30JFrame.*personList*.get(i).position=(String)o;

break; }

case 2: {

try {

Example11\_30JFrame.*personList*.get(i).salary = new Integer((String)o);

}

catch (Exception e) {

Example11\_30JFrame.*personList*.get(i).salary = 0; }

break; }

case 3:

try {

Example11\_30JFrame.*personList*.get(i).rating = new Integer((String)o); }

catch (Exception e) {

Example11\_30JFrame.*personList*.get(i).rating=0; }

}

}

В методе происходит преобразование объекта ( Object) o в строку для записи поля name или в целое число для заполнения полей position, salary или rating . При преобразовании в целое число используется блок try , чтобы обработать исключительную ситуацию, возникающую при попытке выполнить преобразование того, что не является числом, используем блок catch. Внутри этого блока соответствующее поле устанавливаем в нуль.

Приведем полный текст класса MyTableModel:

class **MyTableModel** extends DefaultTableModel {

// Добавим столбцы и заполним их шапку

public **MyTableModel**() {

addColumn("Фамилия");

addColumn("Должность");

addColumn("Оклад");

addColumn("Рейтинг");

}

//кол строк

public int **getRowCount**() {

return Example11\_30JFrame.*personList*.size();

}

//данные таблицы

public Object **getValueAt**(int row, int column){

switch (column) {

case 0:

return Example11\_30JFrame.*personList*.get(row).name;

case 1:

return Example11\_30JFrame.*personList*.get(row).position;

case 2:

return Example11\_30JFrame.*personList*.get(row).salary;

case 3:

return Example11\_30JFrame.*personList*.get(row).rating;

}

return "";

}

// Изменение элемента

public void **setValueAt**(Object o, int i, int i1) {

switch (i1) {

case 0: {

Example11\_30JFrame.*personList*.get(i).name=(String)o;

break; }

case 1: {

Example11\_30JFrame.*personList*.get(i).position=(String)o;

break; }

case 2: {

try {

Example11\_30JFrame.*personList*.get(i).salary = new Integer((String)o);

}

catch (Exception e) {

Example11\_30JFrame.*personList*.get(i).salary = 0; }

break; }

case 3:

try {

Example11\_30JFrame.*personList*.get(i).rating = new Integer((String)o); }

catch (Exception e) {

Example11\_30JFrame.*personList*.get(i).rating=0; }

}

}

}

1. В конструкторе Example11\_30JFrame создадим экземпляр класса MyTableModel

model = new MyTableModel()

1. Назначим созданную нами модель в качестве модели для таблицы jTable1

jTable1.setModel(model);

1. Теперь нужно изменить реакцию на нажатие кнопок. В обработчики для реакции на нажатие кнопок, которые создают или удаляют объекты класса, вставляем строку для обновления таблицы. Метод jTable1.getSelectedRow() возвращает номер выделенной строки таблицы.

private void **jButtonNewPersActionPerformed**(java.awt.event.ActionEvent evt) {

*personList*.add(new Class2());

jTable1.updateUI(); //обновить таблицу

}

private void **jButtonDelPersActionPerformed**(java.awt.event.ActionEvent evt) {

try {

*personList*.remove(jTable1.getSelectedRow());

jTable1.updateUI();} //обновить таблицу }

catch (Exception e) {}

}

Изменяем реализацию выполнения методов обработки данных Class1/ Class2:

Код, обеспечивающий выполнение функции-метода 1 обработки данных класса-родителя для кнопки "Увеличить оклад" ( JButtonIncSalary):

private void **jButtonIncSalaryActionPerformed**(java.awt.event.ActionEvent evt) {

//метод для текущего объекта

*personList*.get(jTable1.getSelectedRow()).incSalary();

jTable1.updateUI(); //обновить таблицу

}

Обработчик события для кнопки ***"Иван в инженеры***" (JButtonPosition2Engieer):

private void **jButtonPosition2EngineerActionPerformed**(java.awt.event.ActionEvent evt) {

//метод для всех объектов из списка

for (int i = 0; i < *personList*.size(); i++) {

if (*personList*.get(i).name.toUpperCase().startsWith("ИВАН")) {

*personList*.get(i).Position2Engieer();

}

}

jTable1.updateUI(); //обновить таблицу

}

Обработчик события для кнопки "***Увеличить оклад по рейтингу***" (JbuttonChngRating):

private void **jButtonChngRatingActionPerformed**(java.awt.event.ActionEvent evt) {

//метод для всех объектов из списка

for (int i = 0; i < *personList*.size(); i++) {

*personList*.get(i).chngSalary();

}

jTable1.updateUI(); //обновить таблицу

}

## Приложение

**Задания для выполнения самостоятельной работы**

Все задания взяты из Сборника задач по программированию. – Одесса: ОНАС им. А.С. Попова, 2011

11. Элементы объектно-ориентированного программирования

11.1. Cоздание данных типа «класс»

***Средний уровень***

Задание: создать класс с полями, указанными в индивидуальном задании (табл. 11.2, столб 2).

Реализовать в классе методы:

* конструктор по умолчанию;
* конструктор перезагрузки с параметрами;
* деструктор для освобождения памяти (с сообщением об уничтожении объекта);
* функции обработки данных (1 и 2), указанные в индивидуальном задании (табл. 11.2, столбцы 3 и 4);
* функцию формирования строки информации об объекте.

Создать проект для демонстрации работы: сформировать объекты со значениями-константами и с введенными значениями полей объекта из компонентов Edit. Выводить результаты в компонент Memo.

Таблица 11.2 – Варианты индивидуальных заданий.

|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **вар** | **Класс-родитель и**  **его поля** | **Функция-метод 1**  **обработки данных** | **Функция-метод 2 обра-**  **ботки данных** |
| 1 | Дата (три числа):  день, месяц, год | Определить, является ли год высокосным (кратным 4) | Увеличить дату на 5 дней |
| 2 | Дата (три числа):  день, месяц, год | Увеличить год на 1 | Уменьшить дату на 2 дня |
| 3 | Дата (три числа):  день, месяц, год | Определить, совпадают ли номер месяца и число дня | Увеличить дату на один  месяц |
| 4 | Время (три числа):  часы, минуты,  секунды | Вычислить количество секунд в указанном времени | Увеличить время на 5секунд |
| 5 | Время (три числа):  часы, минуты,  секунды | Вычислить количество полных минут в указанном времени | Уменьшить время на 10  минут |
| 6 | Время (три числа):  часы, минуты,  секунды | Определить количество минут до полуночи (24:00:00) | Увеличить время 100 минут |
| 7 | Координаты  изображения  прямоугольника:  x1, y1, x2, y2 | Вычислить площадь прямоугольника в пикселях | Изобразить прямоугольник на форме(Image) с толщиной линии  2 пикселя |
| 8 | Координаты  изображения  прямоугольника:  x1, y1, x2, y2 | Вычислить длину диагонали прямоугольника в пикселях | Изобразить прямоугольник и его диагональ на форме (Image) |
| 9 | Координаты  изображения  прямоугольника:  x1, y1, x2, y2 | Определить, является ли прямоугольник квадратом | Изобразить прямоугольник на форме (Image), закрашенный  зеленым цветом |
| 10 | Правильная дробь:  числитель, знаменатель | Выразить значение дроби в процентах | Найти сумму цифр значения знаменателя |
| 11 | Комплексное число: действительная (a1) и мнимая (b1) части числа | Вычислить модуль комплексного числа | Найти комплексное число, обратное заданному |
| 12 | Комплексное число: действительная (a1) и мнимая (b1) части числа | Вычислить произведение комплексного числа на число, вводимое пользователем | Вычислить аргумент комплексного числа в градусах |
| 13 | Книга: название, количество страниц, цена | Вычислить среднюю стоимость одной страницы | Увеличить цену книги в два раза, если название начинается со слова «Программирование» |
| 14 | Книга: название, автор, год издания | Вычислить, сколько лет книге | Количество дней, прошедших после года издания книги |
| 15 | Работник: фамилия, оклад, дата поступления на работу (три числа:  день, месяц, год) | Вычислить стаж работы работника на данном предприятии | Сколько дней прошло после года поступления на работу |
| 16 | Работник: фамилия, оклад, год рождения | Вычислить возраст работника | Сколько календарных дней до исполнения работнику 50 лет |
| 17 | Вектор на плоскости: координаты вектора на плоскости (x1, y1, x2, y2) | Вычислить длину вектора | Изобразить линию вектора на форме (Image) с толщиной линии 2 пикселя |
| 18 | Вектор на плоскости: координаты вектора на плоскости (x1, y1, x2, y2) | Вычислить координаты середины вектора | Равен ли угол наклона вектора 45 градусов? |
| 19 | Вектор на плоскости: координаты вектора на плоскости (x1, y1, x2, y2) | Вычислить координаты вектора, удвоенной длины | Вычислить площадь прямоугольного треугольника, образованного вектором и прямыми, параллельн. осям Оx, Оy |
| 20 | Цилиндр: диаметр основания, высота | Вычислить объем цилиндра | Изобразить круг заданного диаметра на форме (Image), закрашенный красным цветом |
| 21 | Параллелепипед:  длины сторон | Вычислить объем параллелепипеда | Вычислить длину наибольшей диагонали параллелепипеда |
| 22 | Параллелепипед:  длины сторон | Вычислить площадь поверхности | Вычислить сумму длин всех ребер параллелепипеда. |
| 23 | Четыре целых  числа: a, b ,c, d | Вычислить среднеарифметическое чисел | Определить максимальное из чисел |
| 24 | Три вещественных  числа x, y, z | Вычислить среднее  геометрическое чисел | Определите, сколько цифр содержит сумма заданных трех чисел. |
| 25 | Товар: наименование, цена, год выпуска | Определить, сколько лет назад был выпущен товар | Увеличить цену товара на 20%, если в наименовании товара есть слово «TV». |
| 26 | Товар: наименование, цена в *рублях*, изготовитель | Пересчитать цену товара в долларах | Увеличить цену товара в долларах, если название товара содержит слово «Toyota». |
| 27 | Эллипс: координаты эллипса: x1, y1, x2, y2 | Определить, является ли эллипс окружностью? | Изобразить эллипс на форме (Image) зеленым цветом . |
| 28 | Книга: название,  количество страниц | Увеличить количество страниц на 10 | Уменьшить цену в два раза, если количество страниц больше 100  (после увеличения) |
| 29 | Комната: длина, ширина, высота (в метрах) | Площадь стен (вместе  с окнами и дверьми) | Площадь стен без окна (размер 2×15 м) и двери (размер 2 ×8 м). |
| 30 | Работник: фамилия, должность, оклад | Увеличить оклад на 15% (каждому работнику) | Работникам, у которых фамилия начинается с сочетания букв «Иван», присвоить должность  «инженер». |

11.2. Наследование классов

***Средний уровень***

Задание:

Для класса, созданного в предыдущем задании (по вариантам табл. 11.2) создать класс-потомок с полями, указанными в индивидуальном задании (табл. 11.5, столб 2).

Реализовать в классе-потомке методы:

* конструктор;
* функцию обработки данных, указанную в индивидуальном задании (табл.11.5, столб 3);
* функцию формирования строки информации об объекте.

Создать проект для демонстрации работы: ввод и вывод информации об объектах: классе-родителе и классе-потомке.

Таблица 11.5 – Варианты индивидуальных заданий

|  |  |  |  |
| --- | --- | --- | --- |
| **№**  **вар.** | **Класс-родитель и его поля (из табл. 11.2)** | **Класс-потомок и егополя (поля класса-родителя выделены курсивом)** | **Функция-метод обработки данных объекта класса-потомка** |
| 1 | Дата (три числа):  день, месяц, год | Человек: ФИО, телефон, дата рождения, | Количество дней до дня очередного рождения |
| 2 | Дата (три числа):  день, месяц, год | Работник: ФИО, дата поступления на предприятие | Количество лет работы на предприятии |
| 3 | Дата (три числа):  день, месяц, год | Лекарство: наименование, дата выпуска, фирма | Сколько прошло дней от  изготовления лекарства |
| 4 | Время (три числа):  часы, минуты, секунды | Данные по движению поезда: номер поезда, направление, время отправления | Количество минут до отправления поезда с указанным номером и введенное время |
| 5 | Время (три числа):  часы, минуты, секунды | Абонент мобильной связи: *фамилия*, оператор, текущее время | Определить, является ли время льготным для абонента (время от 0 до 8 часов) |
| 6 | Время (три числа):  часы, минуты,  секунды | Данные по занятию:  дисциплина, время начала, аудитория | Какая дисциплина по расписанию начинается в указанное время |
| 7 | Координаты  изображения  прямоугольника:  x1, y1, x2, y2 | Изображение конверта (прямоугольник с линиями диагоналей):  координаты прямоугольника, цвет линий | Площадь верхнего (наддиагонального) треугольника в пикселях |
| 8 | Координаты изображения прямоугольника:  x1, y1, x2, y2 | Изображение прямоугольника с вписанным в его центр кругом: координаты  прямоугольника, радиус круга R (R< x2– x1, R< y2 – y1) | Площадь фигуры между прямоугольником и кругом |
| 9 | Координаты  изображения  прямоугольника:  x1, y1, x2, y2 | Изображение закрашенного прямоугольника с текстом: координаты прямоугольника, заданный текст,цвет закрашивания | Произведение периметра и длины диагонали прямоугольника в пикселях |
| 10 | Правильная дробь: числитель, знаменатель | Смешанная дробь:  целая часть, числитель и знаменатель | Представить смешанную дробь в виде десятичного вещественного числа |
| 11 | Комплексное число: действительная (a1) и мнимая (b1) части числа | Комплексное число: действительная и мнимая часть числа, цвет и толщина линии для отображения комплексного числа | Изобразить геометрическое представление комплексного числа на форме (оси изобразить черным цветом, а само число указанной толщиной и цветом) |
| 12 | Комплексное число: действительная и мнимая часть числа | Комплексное сопротивление: сопротивление резистора  (действительная часть), значение индуктивности (*мнимая часть*), угловая частота | Вычислить модуль и аргумент комплексной проводимости участка цепи «резистор - индуктивность». |
| 13 | Книга: название, количество страниц, цена | Книга: название, количество страниц, цена, скидка в процентах | Стоимость книги с учетом скидки. |
| 14 | Книга: название, автор, год издания | Книга:  название, автор, год издания, цена | Уменьшить стоимость книги на 20%, если книге больше 5 лет. |
| 15 | Работник: фамилия, оклад, дата поступления на работу | Работник предприятия: фамилия, оклад, дата поступления на работу, дата рождения | Определить, сколько лет нужно работать работнику до 60 лет, а если ему больше 60, то сколько лет он работает после 60 лет. |
| 16 | Работник: фамилия, оклад, дата рождения  (три числа: день, месяц, год) | Работник фирмы:  фамилия, оклад, дата рождения, должность | Увеличить оклад работникам старше 30 лет (на текущую дату) с должностью программист на 20%. |
| 17 | Вектор на плоскости: координаты  вектора на плоскости (x1, y1, x2, y2) | Вектор и точка на плоскости: координаты вектора ( x1, y1,x2, y2); координаты точки–x3, y3 | Определить площадь треугольника, образованного вектором и точкой. |
| 18 | Вектор на плоскости: координаты вектора на плоскости (x1, y1, x2, y2) | Два вектора с общим началом ( x1, y1) на плоскости: координаты первого вектора –x1, y1, x2, y2; координаты второго вектора– x1, y1, x3, y3 | Определить координаты вектора суммы двух векторов. |
| 19 | Вектор на плоскости: координаты вектора на плоскости (x1, y1, x2, y2) | Два параллельных вектора на плоскости одинаковой длины: координаты первого  вектора – x1, y1, x2, y2; второй вектор смещен вправо по оси Ох на величину а, второй – вниз по оси Оу на b | Определить площадь параллелограмма, образованного этими векторами и линиями, соединяющих их начала и концы. |
| 20 | Цилиндр: диаметр основания, высота | Изолированный провод: диаметр, длина, удельный вес; толщина изоляции и её удельный вес | Определить вес изолированного провода. |
| 21 | Параллелепипед:  длины сторон | Металлический брус:  ширина, высота, длина, удельный вес | Определить вес металлического бруса. |
| 22 | Параллелепипед:  длины сторон | Балка с прямоугольным сечением: ширина, высота, длина, удельный вес; количество равных частей, на которое её распилят | Площадь поверхности одной части распиленной балки и её вес. |
| 23 | Четыре целых числа:  a, b ,c, d | Пять чисел: четыре целых числа (a,d,c,d) и число х | Вычислить сумму квадратов разности каждого из четырех чисел и числа х. |
| 24 | Три вещественных числа: x,y,z | Два набора чисел:  три вещественных числа x,y,z и три вещественных числа a, b ,c | Определить скалярное произведение двух наборов чисел. |
| 25 | Товар: наименование, цена, год выпуска | Товар:  наименование, цена, год выпуска, дата поступления товара | Количество дней после года выпуска товара до текущего дня. |
| 26 | Товар: наименование, цена в рублях, изготовитель | Товар: наименование, цена в рублях, изготовитель, год выпуска, скидка в процентах | Изменить стоимость товара с учетом скидки для товаров, изготовленных фирмой более двух лет назад. |
| 27 | Эллипс: координаты эллипса: x1, y1, x2, y2 | Дуга эллипса: координаты изображения эллипса x1, y1, x2, y2, координаты концов дуги x3, y3, x4, y4 | Построить изображение дуги эллипса на форме (Image) синим цветом, толщиной линии 2 пикселя. |
| 28 | Книга: название, количество страниц, цена | Изданная книга:название, количество страниц, цена, автор книги, дата поступления в типографию | Сколько дней книга находилась в типографии |
| 29 | Комната: длина, ширина, высота (в метрах) | Помещение для офиса: Состоит из комнаты с (длиной, шириной, высотой)  и коридора с (длиной, шириной, высотой) | Определить количество краски, необходимое для покраски стен и потолка помещений офиса (в каждой комнате одно окно размером 2×15 м). .Дополнительно задается расход краски на 1 м2 |
| 30 | Работник: фамилия, должность, оклад | Работник предприятия:  фамилия, должность, оклад, рейтинг (в 100-бальной системе) | Увеличить оклад работников на 20%, если их рейтинг от 60 до 75 балов, на 40%, если их рейтинг от 75 до 90 балов, на 60%, если их рейтинг от 90 до 100 балов. |

.