|  |  |  |  |
| --- | --- | --- | --- |
|  | | для прик эмбл |  |
| *МИНОБРНАУКИ РОССИИ* | | | |
| ***Федеральное государственное бюджетное образовательное учреждение***  ***высшего профессионального образования***  *"Московский государственный университет информационных технологий, радиотехники и* *электроники"*  *(МИРЭА МГУПИ)* | | | |
| Институт Информационных Технологий | | | |
| Кафедра МОСИТ | | | |

|  |
| --- |
| Лабораторная работа №1  По дисциплине: Объектно ориентированное программирование |
| Выполнила студентка группы: ИКБО-02-13  Силаева А.Э. |
| *2017 год* |
|  |
|  |
|  |

Задание 1

Цель: Ввести n строк с консоли. Вывести на консоль те строки, длина которых меньше средней, также их длины.

Блок схема

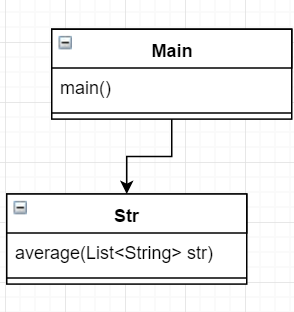
Нахождение средней длины



Основная функция Main()



Проект-схемы иерархии классов задачи. Описание классов и методов.



Класс Main - основной класс выполнении программы

Метод main() – выполнение основной задачи программы, а именно вывод в консоль те строки, длина которых меньше средней, также их длины.

Класс Str – класс,в конструкторе которого, создается пустая переменная типа String, и реализован метод average();

Метод average(List<String> str) – нахождение средней длины строк

Программный код реализации задачи.

**import** com.sun.org.apache.xpath.internal.functions.FuncFalse;  
  
**import** java.io.IOException;  
**import** java.util.ArrayList;  
**import** java.util.List;  
**import** java.util.Scanner;  
*/\*Первая лабораторная \*/  
/\*Ввести n строк с консоли. Вывести на консоль те строки, длина которых меньше сред-ней, также их длины.\*/***class** Str { *// класс в котором создается пустая строка* **private final** String **st**;  
 *// Класс Меин* Str()  
 {  
 **st** = **""**;  
 }  
 Str(String st)  
 {  
 **this**.**st** = st;  
 }  
 String get(String st)  
 {  
 **return** st;  
 }  
 **public static int** average(List<String> st) *// Функция для определение самой большой и самой маленькой строки и нахождение среднего* {  
 **int** minimun = st.get(0).length();  
 **int** maximum = st.get(0).length();  
 **int** av = 0;*// среднее значение* **for** (String s : st) *// прохождение списка через цикл* {  
  
 **if** (s.length() < minimun && s.length() <= 15 && !s.equalsIgnoreCase(**"empty"**)) {  
 minimun = s.length();*// длина элемента списка* }  
 **if** (s.length() > maximum && s.length() <= 15 ) {  
 maximum = s.length();  
  
 }  
  
 }  
  
 av = (minimun + maximum) / 2;*// нахождение среднего* **return** av;  
 }  
}  
  
**public class** Main { *// Класс Меин* **public static void** main(String[] args) **throws** IOException {  
  
 **int** av = 0;  
 **boolean** f = **false**;  
  
 System.***out***.print(**"Введите количество строк "**);  
 **try** {  
 Scanner n = **new** Scanner(System.***in***);*// создаём объект класса Scanner* **int** number = n.nextInt(); *//считывает целое число с потока ввода и сохраняем в переменную* **if** (number > 15) {  
 System.***out***.print(**"Введено больше n чем может быть"**);  
 System.*exit*(0);  
 }  
 **if** (number < 1) {  
 System.***out***.println(**"Введено число 0 или меньше 0"**);  
 }  
 Scanner in = **new** Scanner(System.***in***);*// создаём объект класса Scanner* List<String> strings = **new** ArrayList<String>(); *//создает пустой список из строк* Str stt = **new** Str();  
 String[] a;  
 String str1;  
 System.***out***.println(**"Вводите по одной строке через enter"**);  
 System.***out***.println(**"Если вводить все одной строкой, то по пробелу будут отделены строки"**);  
 *// System.out.println("Ытобы ввести пустую строку введите ");* **for** (**int** i = 1; i <= number; i++) {  
 System.***out***.println(**"Введите строку "** + i);  
 a = in.nextLine().split(**" "**);  
 str1 = a[0];  
  
 **if** (str1.equalsIgnoreCase(**"empty"**)) {  
 System.***out***.println(**"Данная строка является пустой и обрабатываться не будет"**);  
 **continue**;  
 }  
 **if** (a.**length** > 1) {  
 System.***out***.println(**"Введнная строка после пробела обрабатываться не будет"**);  
 }  
 **if** (str1.length() > 15) {  
 System.***out***.println(**"Данная ,больше 15 и обрабатываться не будет"**);  
 **continue**;  
 }  
 strings.add(stt.get(str1));*// добавление в список строк* }  
 av = stt.*average*(strings);*// функция по нхождению среднеего числа строк* System.***out***.println(**"Вывод слов длина, которых меньше средней, средняя длина "** + av);  
  
 **for** (String s : strings) {  
 **if** (s.length() < av) {  
 f = **true**;  
 System.***out***.println(**"Строка "** + s + **" Длина "** + s.length()); *// Вывод* }  
  
  
 }  
 System.***out***.println();  
 **if** (f == **false**) {  
 System.***out***.println(**"Таких строк нет"**);  
 }  
 }  
 **catch** (Exception e){  
 System.***out***.println(**"Вы ввели недопустимый символ"**);  
 }  
  
 }  
}

План тестирования по задачи.

|  |  |  |  |
| --- | --- | --- | --- |
| N – количество строк | Ввод строк и Вывод сообщений если строка неправильная | Вывод средней длины | Вывод строк |
| 5 | Привет, няня, как,дела, вявя | 4 | Строка как Длина 3 |
| 8 | Привет, няня, как, дела, вявя, ны, ла, ряр, мяу | 4 | Строка как Длина 3  Строка ны Длина 2  Строка ла Длина 2  Строка ряр Длина 3  Строка мяу Длина 3 |
| 3 | Я, Я, Я | 1 | Таких строк нет |
| 4 | 1234, 21, 12, 32 | 3 | Строка 21 Длина 2  Строка 12 Длина 2  Строка 32 Длина 2 |
| 16 | NULL | 0 | Введено больше n чем может быть |
| 8 | Строка номер 1  empty  Данная строка является пустой и обрабатываться не будет  Строка номер 2  wertyuiop[asdfghjkl;xcvbnm,.  Данная ,больше 15 и обрабатываться не будет  Привет Как ты У меня все | 3 | Вывод слов длина, которых меньше средней, средняя длина 3  Строка ты Длина 2  Строка У Длина 1 |
| 5 | Введите строку 1  Как же грустно все  Введенная строка после пробела обрабатываться не будет  Введите строку 2  Эх  Введите строку 3  Ничего  Введите строку 4  никогда  Введите строку 5  не работает  Введнный строка после пробела обрабатываться не будет |  | Вывод слов длина, которых меньше средней, средняя длина 4  Строка Как Длина 3  Строка Эх Длина 2  Строка не Длина 2 |
| 0 | Null | Null | Введено количество строк 0 или меньше 0 |
| adasdas | Null | Null | Введены не допустимые символы |

Скриншоты работы программы.

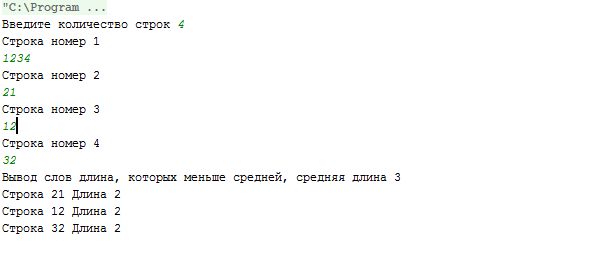


Рис. 1

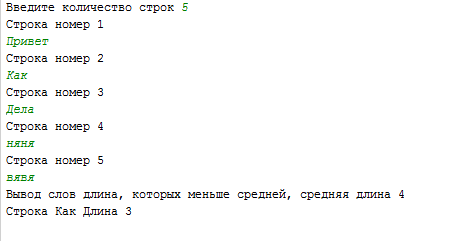


Рис. 2

Вывод: были получены навыки в разработке консольных java-приложений

Задание 2

Цель: Написать регулярное выражение, определяющее является ли заданная строка правильным MAC-адресом.:

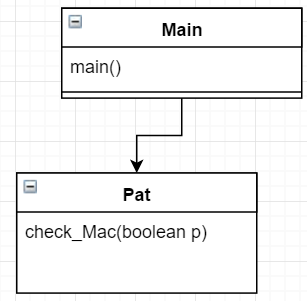
1. пример правильных выражений: aE:dC:cA:56:76:54.
2. пример неправильных выражений: 01:23:45:67:89:Az.

Блок-схема задачи

Функция main()

D:\Загрузки\Untitled Diagram (1).png

Проект-схемы иерархии классов задачи. Описание классов и методов.



Класс Main - основной класс выполнении программы

Метод main() – выполняет основную задачу

Класс Pat – создание переменной для сохранения регулярного выражения

Метод check\_Mac(boolean p) – проверка строки по совпадению регулярных выражений

Программный код реализации задачи.

**import** java.util.Scanner;  
**import** java.util.regex.Matcher;  
**import** java.util.regex.Pattern;  
*/\*3. Написать регулярное выражение, определяющее является ли заданная строка  
правиль-ным MAC-адресом. – пример правильных выражений: aE:dC:cA:56:76:54.  
– пример неправильных выражений: 01:23:45:67:89:Az. \*/***class** Pat {  
 Pattern **p1**;  
 Pat()  
 {  
 **p1** = Pattern.*compile*(**"([a-f]+[A-F]+:){3}+([0-9]{2}+:){2}+[0-9]{2}"**); */\*регулярное выражение\*/* }  
  
 **boolean** check\_mac(String str)  
 {  
 Matcher m = **this**.**p1**.matcher(str);  
 **return** m.matches();  
 }  
 }  
  
  
**public class** Main {  
  
 **public static void** main(String[] args) {  
*// Ввод, мас адрес не введен* Pat p = **new** Pat();  
 System.***out***.print(**"Введите Мас адрес "**);  
 Scanner n = **new** Scanner(System.***in***);*// создаём объект класса Scanner* String str = n.next(); *//считывает целое число с потока ввода и сохраняем в переменную* **if** (str.length() == 0)  
 {  
 System.***out***.println(**"Введена пустая строка"**);  
 }  
 **else** {  
 **boolean** b = p.check\_mac(str);  
 String m = b ? **"Введен правильный MAC адрес: "** + str : **"Введен не правильный MAC адрес: "** + str;  
 System.***out***.println(m);  
 }  
 */\*re+ Одно или более совпадений предыдущего выражения.  
 re{n} Соответствует ровно N Количество вхождений предшествующего выражения.  
 (?: re) Группы регулярных выражений, не помнящая найденный текст.  
 [...] Соответствует любой одиночный символ в скобках  
 \* \*/* }  
}

План тестирования по задачи.

|  |  |
| --- | --- |
| Ввод MAС адреса | Вывод проверки МАС адреса |
| aE:dC:cA:56:76:54 | Введен правильный MAC адрес: aE:dC:cA:56:76:54 |
| 01:23:45:67:89:Az | Введен не правильный MAC адрес: 01:23:45:67:89:Az |
| 01:23:45:67:89:Az234sf234 | Введен не правильный MAC адрес: 01:23:45:67:89:Az234sf234 |
| edrtfgyhjkl;edrtfgyuhjikol | Введена слишком большая строка |
| empty | Введена пустая строка |

Снимок экрана, отражающего работу программы по задаче.

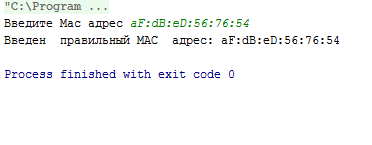


Рис. 1

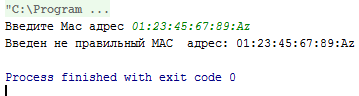


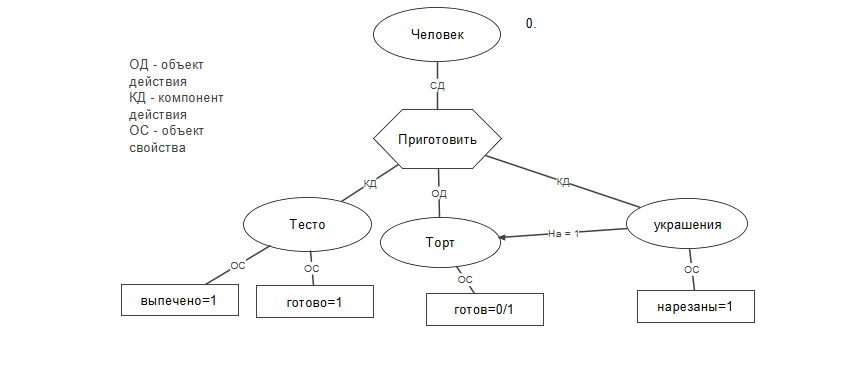
Рис. 2

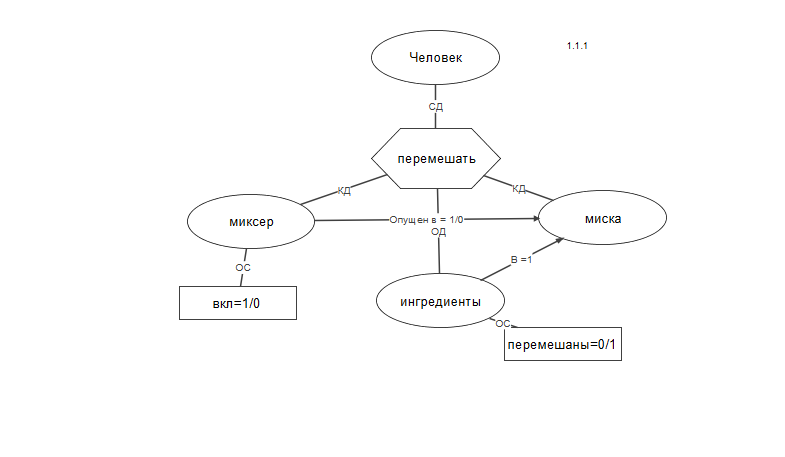
Вывод: Был получен опыт по написанию регулярных выражений на языке java.

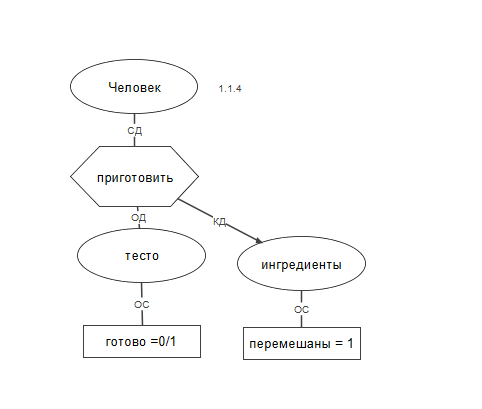
Задание

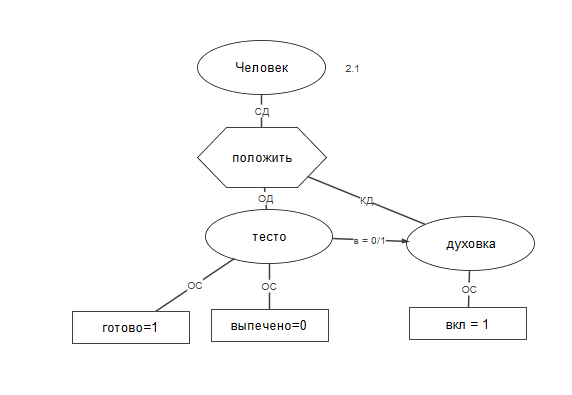
Цель: запрограммировать концептуальную модель приготовления торта.

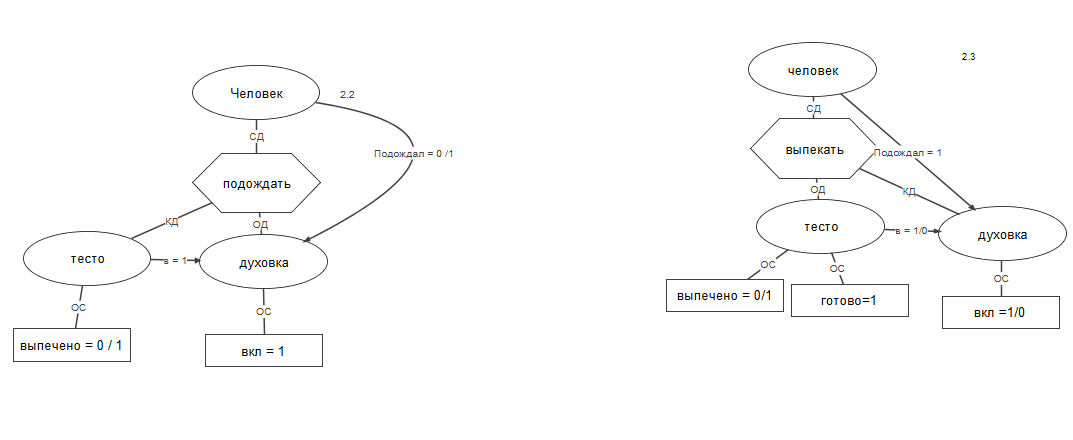
Блок-схема задачи

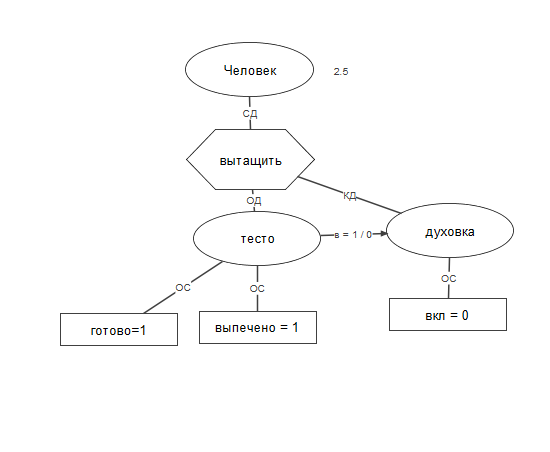


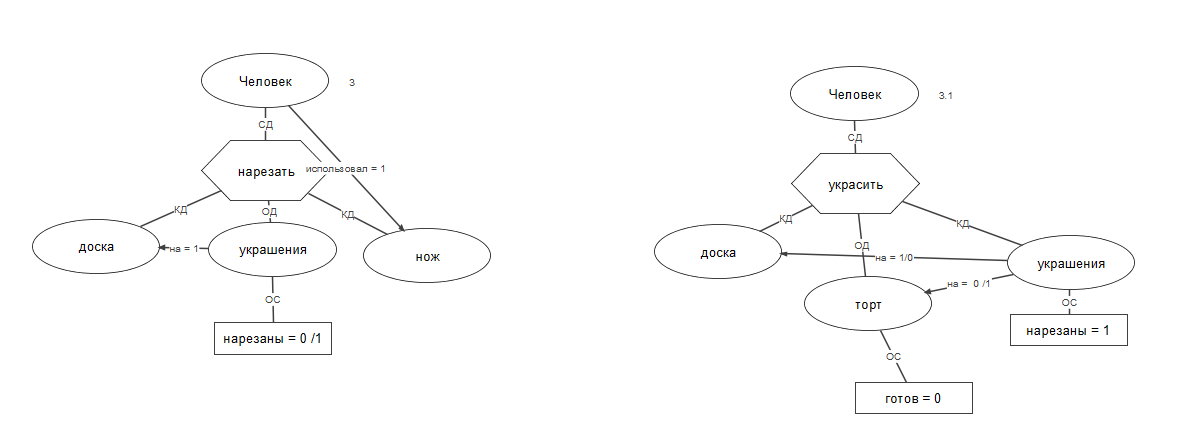




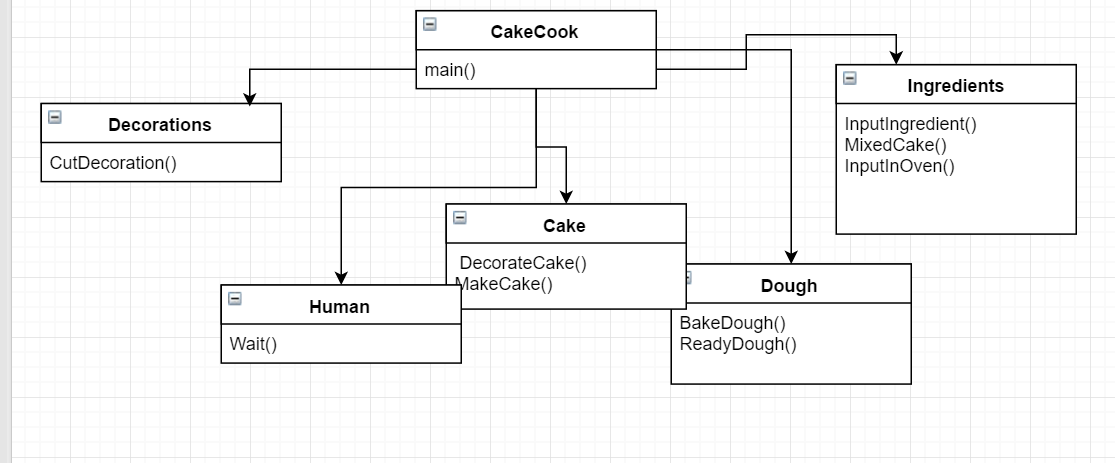








Проект-схемы иерархии классов задачи. Описание классов и методов.



Класс CakeCook - основной класс выполнении программы

Метод main() – выполняет основную задачу готовит торт и выводит сообщения об этом

Cake – класс в котором выполняются методы по готовке торта

DecorateCake() – метод по украшению торта

MakeCake()- метод готовке торта

Ingredients – класс по готовке ингредиентов

InputIngredient() – метод положить ингредиенты в миску

MixedCake() – метод перемешка ингредиентов

InputInOven()- метод положить в духовку

Dough – класс по выпеканию теста

BakeDough()- метод по выпекании теста

ReadyDough()- метод по готовке теста  
Decorates – класс для нарезки украшений

CutDecoration() – метод по нарезке украшений

Human – класс для того чтобы подождать когда все приготовиться

Wait() – метод подождать

Программный код реализации задачи.

**package** com.company;  
  
**class** Cake {  
 **boolean ready** = **false**;  
 **boolean decorate** = **false**;  
  
 **public void** DecorateCake() { *// Украшение торта* System.***out***.println(**"Decorated cake"**);  
 **decorate** = **true**;  
 }  
  
 **public void** MakeCake() {*// приготовление торта* System.***out***.println(**"Cake is ready "**);  
 **ready** = **true**;  
 }  
}  
  
**class** Decorates {  
 **boolean cut** = **false**;  
  
 **public void** CutDecoration() { *// Нарезать украшения* System.***out***.println(**"Decorates is cut"**);  
 **cut** = **true**;  
 }  
}  
  
**class** Ingredients {  
 **boolean in\_plate** = **false**;  
 **boolean mix** = **false**;  
 **boolean in\_oven** = **false**;  
  
 **public void** InputIngredient() { *// положить ингредиенты в тарелку* System.***out***.println(**"Ingredients in plate"**);  
 **in\_plate** = **true**;  
 }  
  
 **public void** MixedCake() {*// Мешать ингредиенты* System.***out***.println(**"Ingredients is mixed"**);  
 **mix** = **true**;  
 }  
  
 **public void** InputInOven() { *// Полодить перемешанные ингредиенты в духовку* System.***out***.println(**"Mixed Ingredients in oven"**);  
 **in\_oven** = **true**;  
 }  
  
}  
**class** Human{  
 **boolean wait** = **false**;  
 **public void** Wait() { *// Подождать духовку* System.***out***.println(**"Human waited"**);  
 **wait** = **true**;  
 }  
}  
  
**class** Dough {  
  
 **boolean bake** = **false**;  
 **boolean ready** = **false**;  
  
  
  
 **public void** BakeDough() { *// Тесто выпекается* System.***out***.println(**"Dough is baked"**);  
 **bake** = **true**;  
 }  
  
 **public void** ReadyDough() { *// тесто готовится* System.***out***.println(**"Dough is ready"**);  
 **ready** = **true**;  
 }  
}  
  
**public class** Main {  
  
 **public static void** main(String[] args) {  
 *// write your code here  
 //System.out.print("LOL");* Cake cake = **new** Cake();  
 Ingredients ingredient = **new** Ingredients();  
 Dough dough = **new** Dough();  
 Decorates decorates = **new** Decorates();  
 Human human = **new** Human();  
 **if** (ingredient.**in\_plate** == **false**)  
 {  
 System.***out***.println(**"Ingredients not in plate"**);  
 ingredient.InputIngredient();  
 }  
 **if** (ingredient.**mix** == **false**)  
 {  
 System.***out***.println(**"Ingredients not mixed"**);  
 ingredient.MixedCake();  
 }  
 **if** (ingredient.**in\_oven** == **false**)  
 {  
 System.***out***.println(**"Mixed Ingredients not in oven"**);  
 ingredient.InputInOven();  
  
 }  
 **if** (human.**wait** == **false**)  
 {  
 System.***out***.println(**"Human not wait"**);  
 human.Wait();  
 }  
 **if** (dough.**bake** == **false**)  
 {  
 System.***out***.println(**"Dough not baked"**);  
 dough.BakeDough();  
 }  
 **if** (dough.**ready** == **false**)  
 {  
 System.***out***.println(**"Dough is not ready"**);  
 dough.ReadyDough();  
 }  
 **if** (decorates.**cut** == **false**)  
 {  
 System.***out***.println(**"Decorates is not cut"**);  
 decorates.CutDecoration();  
 }  
 **if** (cake.**decorate** == **false**)  
 {  
 System.***out***.println(**"Cake is not decorated"**);  
 cake.DecorateCake();  
 }  
 **if** (cake.**decorate** == **true** && decorates.**cut** ==**true**)  
 {  
  
 cake.MakeCake();  
 }  
  
  
 }  
}

План тестирования по задачи.

|  |  |
| --- | --- |
| Начальные параметры | Вывод |
| Если есть ингредиенты | Ingredients not in plate  Ingredients in plate  Ingredients not mixed  Ingredients is mixed  Mixed Ingredients not in oven  Mixed Ingredients in oven  Human not wait  Human waited  Dough not baked  Dough is baked  Dough is not ready  Dough is ready  Decorates is not cut  Decorates is cut  Cake is not decorated  Decorated cake  Cake is ready |
| Если нет ингредиентов | Ingredients not here  Ingredients not in plate  Ingredients not mixed  Mixed Ingredients not in oven  Human not wait  Dough not baked  Dough is not ready  Decorates is not cut  Cake is not decorated  Cake is not ready |

Снимок экрана, отражающего работу программы по задаче.

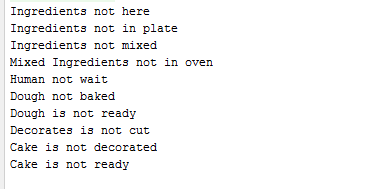


Рис. 1

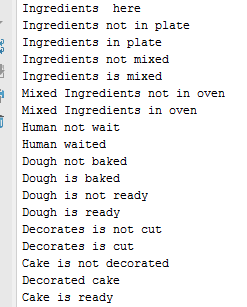


Рис. 2

Вывод: был получен опыт по программировании иерархии классов концептуальной модели по приготовлению торта c помощью языка программирования java.