ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| acc. |  |  |  | Д.А.Кочин |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2 |
| «Функциональное тестирование методом черного ящика» |
| по курсу: Управление качеством программного обеспечения |
|  |
|  |

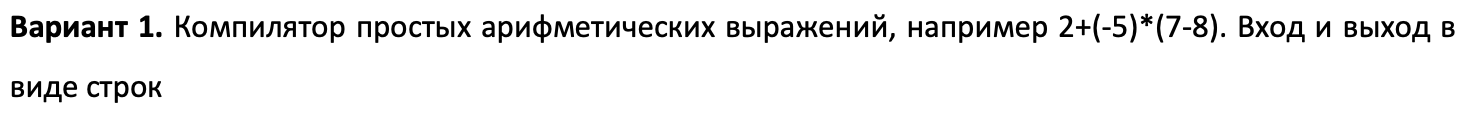
РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 4831 |  | 08.10.2020 |  | К.А.Корнющенков |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2020

1. **Задание на лабораторную работу**

В рамках лабораторной работы необходимо произвести функциональное тестирование кода, оценить его покрытие и качество тестов



1. **Код программы**
2. package com.company;  
     
   import java.util.LinkedList;  
   import java.util.Scanner;  
     
   class Pair<T,U> {  
    public final T t;  
    public final U u;  
     
    public Pair(T t, U u) {  
    this.t= t;  
    this.u= u;  
    }  
   }  
     
   public class Main {  
     
    public static boolean isNumeric(char str) {  
    try {  
    Double.*parseDouble*(String.*valueOf*(str));  
    return true;  
    } catch(NumberFormatException e){  
    return false;  
    }  
    }  
     
    public static Boolean checkData(String data){  
    for (int i=0;i<data.length();i++){  
    if ((i ==0 || i == data.length() - 1) && !*isNumeric*(data.charAt(i))) {  
    return false;  
    }  
    if (*isNumeric*(data.charAt(i)) | String.*valueOf*(data.charAt(i)).equals("+")  
    | String.*valueOf*(data.charAt(i)).equals("\*") | String.*valueOf*(data.charAt(i)).equals("/") | String.*valueOf*(data.charAt(i)).equals("-")){  
    continue;  
    } else {  
    return false;  
    }  
    }  
    return true;  
    }  
     
    public static Pair<LinkedList<Integer>, LinkedList<String>> parseData(String data){  
    LinkedList<Integer> number = new LinkedList<Integer>();  
    LinkedList<String> action = new LinkedList<String>();  
    String timeNumber = "";  
    for (int i=0;i<data.length();i++){  
    if (!*isNumeric*(data.charAt(i))){  
    action.add(String.*valueOf*(data.charAt(i)));  
    if (timeNumber != "") {  
    number.add(Integer.*valueOf*(timeNumber));  
    timeNumber = "";  
    }  
    } else{  
    timeNumber += String.*valueOf*(data.charAt(i));  
    }  
    }  
    if (timeNumber != "") {  
    number.add(Integer.*valueOf*(timeNumber));  
    timeNumber = "";  
    }  
    return new Pair(number,action);  
    }  
     
    public static Pair<LinkedList<Integer>, LinkedList<String>> hardAction(LinkedList<Integer> number, LinkedList<String> action){  
    Integer i = 0;  
    Boolean check = true;  
    while (check){  
    if (action.get(i).equals("\*")) {  
    number.set(i,number.get(i)\*number.get(i+1));  
    number.remove(i+1);  
    action.remove(i+1-1);  
    i = 0;  
    }else if (action.get(i).equals("/")) {  
    number.set(i,number.get(i)/number.get(i+1));  
    number.remove(i+1);  
    action.remove(i+1-1);  
    i = 0;  
    }  
    i += 1;  
    if (action.size() <= i){  
    break;  
    }  
    }  
    return new Pair(number,action);  
    }  
     
    public static String simpleAction(LinkedList<Integer> number, LinkedList<String> action){  
    Integer i = 0;  
     
    while (!action.isEmpty()){  
    if (action.get(i).equals("+")) {  
    number.set(i,number.get(i)+number.get(i+1));  
    number.remove(i+1);  
    action.remove(i+1-1);  
    i = 0;  
    }else if (action.get(i).equals("-")) {  
    number.set(i,number.get(i)-number.get(i+1));  
    number.remove(i+1);  
    action.remove(i+1-1);  
    i = 0;  
    }  
    }  
    return String.*valueOf*(number.get(0));  
    }  
     
    public static String action(String data){  
     
    if (*checkData*(data)){  
    Pair<LinkedList<Integer>, LinkedList<String>> parse = *parseData*(data);  
    parse = *hardAction*(parse.t,parse.u);  
    return *simpleAction*(parse.t,parse.u);  
    }else {  
    return "Ошибка";  
    }  
    }  
     
    public static void main(String[] args) {  
    Scanner in = new Scanner(System.*in*);  
    System.*out*.print("Введите уравнение: ");  
    String data = in.nextLine();  
    System.*out*.println(*action*(data));  
    }  
   }
3. **Ошибки в требованиях к спецификации**
4. В требовании номер один необходимо уточнить тип кофеварки и на чем основан принцип работы устройства (неполнота)
5. В данном требовании необходимо уточнить весь функционал электрокофеварка, чтобы полноценно описать и спроектировать устройство прибора (неполнота)
6. В данном требовании всё правильно
7. В данном требование необходимо какая кнопка должна включаться и какая крышка должна быть закрыть (неполнота)
8. Данное требование противоречит само себе. Непонятно куда необходимо наливать воду, если крышка закрыта (некорректность)
9. Необходимо уточнить с помощью чего надо делать управление электрокофеварки(экран на самом устройстве или какие-то стороннее приложение для отслеживания данных) (HTCPCP - протокол для управления, слежения и диагностики приборов для приготовления кофе.) (неполнота)
10. Требование номер семь невозможно проверить, необходимо согласовать дизайн с заказчиком в том смысле, что сперва будет утвержден дизайн и только после этого будет продолжена реализация. (непроверяемость)
11. **Спецификация на тесты**

Функция checkData(string)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя теста | Описание сценария | Входные данные | Выходные данные |
| Test\_checkData\_1 | Проверка входной строки на корректность нашего алфавита | 3+4\*5\*10 | true |
| Test\_checkData\_2 | Проверка входной строки на корректность нашего алфавита | +5+10 | false |
| Test\_checkData\_3 | Проверка входной строки на корректность нашего алфавита | 5+x+10 | False |

Функция action (string)

|  |  |  |  |
| --- | --- | --- | --- |
| Имя теста | Описание сценария | Входные данные | Выходные данные |
| action \_1 | Преобразование входных данных в строку, которое содержит чисто или ошибку | 5+5+1-2 | 9 |
| action \_2 | Преобразование входных данных в строку, которое содержит чисто или ошибку | +5+5+1-2 | Ошибка |

1. **Вывод**

В ходе выполнения лабораторной работы были получены навыки написания тестов методом черного ящика.