МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ «БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

ФАКУЛЬТЕТ ЭЛЕКТРОННО-ИНФОРМАЦИОННЫХ СИСТЕМ

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №11

Специальность ПО5

Выполнил:	
А.А. Игнат	юк,
студент гру	⁄ппы ПО-5
Проверил:	
А.А. Крощо	енко,
ст. преп. кафедры ИИТ,	
« »	2022 г

Цель работы: Освоить приемы тестирования кода на примере использования библиотеки JUnit.

Вариант 5.

Задание 1.

12

- Создаете новый класс и скопируйте код класса Sum;
- Создаете тестовый класс SumTest;
- Напишите тест к методу Sum.accum и проверьте его исполнение. Тест должен проверять работоспособность функции accum.
- Очевидно, что, если передать слишком большие значения в Sum.accum, то случится переполнение. Модифицируйте функцию Sum.accum, чтобы она возвращала значение типа long и напишите новый тест, проверяющий корректность работы функции с переполнением. Первый тест должен работать корректно.

```
public class Sum {
   public static int accum(int... values) {
      int result = 0;
      for (int i = 0; i < values.length; i++) {
        result += values[i];
      }
      return result;
   }
}</pre>
```

Рисунок 1.1 - Исходный код класса Sum.

Спецификация ввода: Указанные в тестах значения Спецификация вывода: Результаты тестирования

Код программы и результаты тестирования:

```
SumTest.java X
 src > ● SumTest.java > 😭 SumTest > 🕤 test()
          import static org.junit.Assert.assertEquals;
    3
          import org.junit.Test;
    5
(X)
          public final class SumTest {
    7
               @Test
               public final void test() {
    8
\otimes
                   -assertEquals(Sum.accum(...values: 1, 1, 3, 4, 5), 15; --- Expected [14] but was [15]
  Expected [14] but was [15] test()
    Expected
                               Actual

    X Test run at 5/2/2022, 11:33:59 AM

                              +15
   -14

∨ ⊗ test() Java Test > lab_11_1 > < Default Package > → SumTest

                                                              Expected [14] but was [15]
                                                              java.lang.AssertionError: expected:[14] but was:[15] at SumTest.test(SumTest.java:9)
                                                           Test run at 5/2/2022, 11:33:49 AM
   10
               }
   11
```

Рисунок 1.2 - Тестирование класса Sum (ошибка).

```
TESTI.... ひ ひ ▷ ♠ ▷ ··· ● SumTest.java ×
                                  src > ① SumTest.java > ...
Filter (e.g. text, !exclude, @t... 🔽
                                          import static org.junit.Assert.assertEquals;
                                     1
1/1 tests passed (100%)
                                     2
                                     3
                                          import org.junit.Test;

√ 

✓ lab_11_1 6.0ms

                                     4

∨ ⊘ < Default Package > 6.0ms

                                    5
                                          public final class SumTest {

✓ ✓ SumTest 6.0ms

                                     6

        ✓ test() 6.0ms

                                     7
                                              @Test
                                    8
                                              public final void test() {
                                                   assertEquals(Sum.accum(...values: 1, 2, 3, 4, 5), 15);
                                     9
                                    10
                                    11
                                    12
```

Рисунок 1.3 - Тестирование класса Sum (успех).

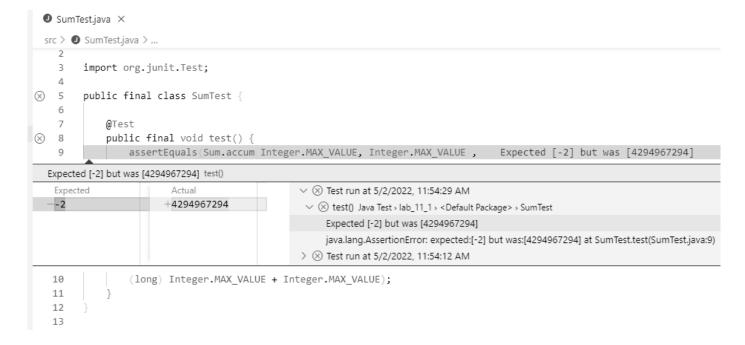


Рисунок 1.4 - Тестирование класса Sum (переполнение).

```
Sum.java X
src > • Sum.java > ...
       public final class Sum {
  1
  2
           public final static long accum(final int... values) {
  3
               long result = 0;
  4
  5
                for (int i = 0; i < values.length; ++i) {
                    result += values[i];
  6
  7
  8
  9
               return result;
 10
 11
 12
```

Рисунок 1.5 - Модифицированный исходный код класса Sum.

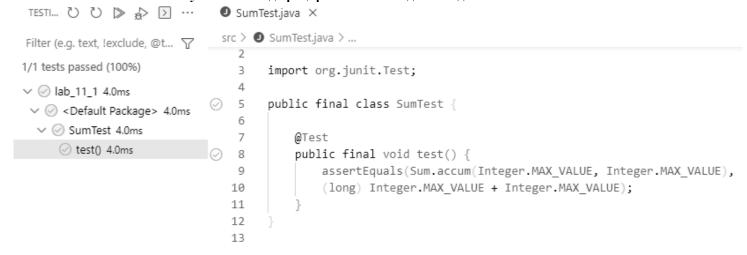


Рисунок 1.6 - Тестирование класса Sum (переполнение отсутствует).

Задание 2.

- Создайте новый проект в рабочей IDE;
- Создайте класс StringUtils, в котором будут находится реализуемые функции;
- Напишите тесты для реализуемых функций.

Написать тесты к методу, а затем реализовать сам метод по заданной спецификации.

Варианты:

5) Реализуйте и протестируйте метод int indexOfDifference(String str1, String str2) который сравнивает две строки и возвращает индекс той позиции, в которой они различаются. Например, indexOfDifference("i am a machine", "i am a robot") должно вернуть 7.

Спецификация метода:

```
indexOfDifference(null, null) = NullPointerException
indexOfDifference("", "") = -1
indexOfDifference("", "abc") = 0
indexOfDifference("abc", "") = 0
indexOfDifference("abc", "abc") = -1
indexOfDifference("ab", "abxyz") = 2
indexOfDifference("abcde", "abxyz") = 2
indexOfDifference("abcde", "xyz") = 0
```

Рисунок 2.1 - Спецификация метода.

Спецификация ввода: Указанные в тестах значения Спецификация вывода: Результаты тестирования

Код программы и результаты тестирования:

● StringUtils.java × src > 1 StringUtils.java > ... public final class StringUtils { 2 public final static int indexOfDifference(final String str1, final String str2) throws NullPointerException { if (str1 == null || str2 == null) 3 throw new NullPointerException(); 4 5 6 if (str1.equals(anObject: "") && str2.equals(anObject: "")) { 8 return -1; 9 10 int i = 0; 11 12 for (; i < str1.length() && i < str2.length(); ++i) {</pre> 13 14 if $(str1.charAt(i) != str2.charAt(i)) {$ 15 return i; 16 17 18 19 return i: 20 21

Рисунок 2.2 - Исходный код файла StringUtils.java.

22

```
TESTING ♂ ♂ ▷ ...
                                 ● StringUtilsTests.java ×
                                 src > ① StringUtilsTests.java > ...
Filter (e.g. text. !exclude. @tag)
                                      import static org.junit.Assert.assertEquals;
                                   1
1/1 tests passed (100%)
                                        import static org.junit.Assert.assertThrows;

√ Ø lab_11_2 5.0ms

                                       import org.junit.Test;

✓ Ø < Default Package > 5.0ms

                                   5
                                        import org.junit.function.ThrowingRunnable;
  ∨ ⊘ StringUtilsTests 5.0ms
     7
                                        public final class StringUtilsTests {
                                   8
                                            @Test
                                            public final void test() {
                                  9
                                                assertThrows(NullPointerException.class, new ThrowingRunnable() {
                                  10
                                   11
                                                    @Override
                                   12
                                                     public void run() throws Throwable {
                                                         StringUtils.indexOfDifference(|str1: null, |str2: null);
                                   13
                                   14
                                   15
                                                 });
                                   16
                                                 assertThrows(NullPointerException.class, new ThrowingRunnable() {
                                   17
                                   18
                                                    @Override
                                   19
                                                     public void run() throws Throwable {
                                                         StringUtils.indexOfDifference(|str1: "i am a machine", str2: null);
                                   20
                                   21
                                   22
                                                 });
                                   23
                                   24
                                                 assertThrows(NullPointerException.class, new ThrowingRunnable() {
                                   25
                                                     @Override
                                                     public void run() throws Throwable {
                                   26
                                   27
                                                         StringUtils.indexOfDifference(str1: null, str2: "i am a robot");
                                   28
                                   29
                                                 });
                                   30
                                                 assertEquals (StringUtils.indexOfDifference ( str1: "", | str2: |""), -1);\\
                                   31
                                                 assertEquals(StringUtils.indexOfDifference(str1: "i am a machine", str2: ""), 0);
                                   32
                                                 assertEquals(StringUtils.indexOfDifference(str1: "", str2: "i am a robot"), 0);
                                   33
                                                 assertEquals(StringUtils.indexOfDifference(|str1: "ab", |str2: "abxyz"), 2);
                                   34
                                   35
                                                 assertEquals(StringUtils.indexOfDifference(str1: "abcde", str2: "abxyz"), 2);
                                                assertEquals(StringUtils.indexOfDifference(str1: "abcde", str2: "xyz"), 0);
                                   36
                                                 assertEquals(StringUtils.indexOfDifference(|str1: "i am a machine", str2: "i am a robot"), 7);
                                   37
                                   38
                                   39
```

Рисунок 2.3 - Исходный код файла StringUtilsTests.java.

Залание 3.

- 1) Импорт проекта. Импортируйте один из проектов по варианту:
- Stack проект содержит реализацию стека на основе связного списка: Stack.java.
- Queue содержит реализацию очереди на основе связного списка: Queue.java.

Разберитесь как реализована ваша структура данных. Каждый проект содержит:

- Клиент для работы со структурой данных и правильности ввода данных реализации (см. метод main()).
 - ТООО-декларации, указывающие на нереализованные методы и функциональность.
 - FIXME-декларации, указывающую на необходимые исправления.
 - Ошибки компиляции (Синтаксические).
 - Баги в коде (!).
 - Метод check() для проверки целостности работы класса.
 - 2) Поиск ошибок.
 - Исправить синтаксические ошибки в коде.
- Разобраться в том, как работает код, подумать о том, как он должен работать и найти допущенные баги.
 - 3) Внутренняя корректность.
 - Разобраться что такое утверждения (assertions) в коде и как они включаются в Java.
 - Заставить ваш класс работать вместе с включенным методом check.
- Выполнить клиент (метод main() класса) передавая данные в структуру используя включенные проверки (assertions).
 - 4) Реализация функциональности.
 - Реализовать пропущенные функции в классе.
- См. документацию перед методом относительно того, что он должен делать и какие исключения выбрасывать.
 - Добавить и реализовать функцию очистки состояния структуры данных.
 - 5) Написание тестов.
 - Все функции вашего класса должны быть покрыты тестами.
 - Использовать фикстуры для инициализации начального состояния объекта.
- Итого, должно быть несколько тестовых классов, в каждом из которых целевая структура данных создается в фикстуре в некотором инициализированном состоянии (пустая, заполненная и тд), а после очищается.
 - Написать тестовый набор, запускающий все тесты.

Спецификация ввода: Пользовательский ввод строковых значений в консоль, указанные в тестах значения

Спецификация вывода: Вывод, хранящихся в стеке, значений в консоль, результаты тестирования

Код программы и результаты тестирования:

```
ひ ひ 

RunAll.java ×

RunAll.java ×

O RunAll.java ×

    TESTING
                                                                                                                                                                    src > stack > • RunAll.java > ...
   Filter (e.g. text, !exclude, @tag)
                                                                                                                                        \nabla
                                                                                                                                                                                                     package stack;
                                                                                                                                                                                1
3/3 tests passed (100%)
                                                                                                                                                                                2
                                                                                                                                                                                3
                                                                                                                                                                                                     import org.junit.runner.RunWith;

✓ Ø lab_11_3 4.0ms

                                                                                                                                                                                4
                                                                                                                                                                                                     import org.junit.runners.Suite;

✓ ✓ stack 4.0ms

                                                                                                                                                                                5
                                                                                                                                                                                                     import org.junit.runners.Suite.SuiteClasses;
         ∨ ⊘ RunAll 4.0ms
                                                                                                                                                                                6
                > ostack.StackClearTest 4.0ms
                                                                                                                                                                                7
                                                                                                                                                                                                    @RunWith(Suite.class)
                > stack.StackPopTest
                                                                                                                                                                                8
                                                                                                                                                                                                    @SuiteClasses({
                > O stack.StackPushTest
                                                                                                                                                                               9
                                                                                                                                                                                                                                          StackClearTest.class,
                                                                                                                                                                           10
                                                                                                                                                                                                                                          StackPopTest.class,

∨ ○ StackClearTest

                                                                                                                                                                            11
                                                                                                                                                                                                                                          StackPushTest.class
                           o test()
                                                                                                                                                                            12

∨ ○ StackPopTest

                                                                                                                                                             2 13
                                                                                                                                                                                                     public class RunAll {
                           test()
                                                                                                                                                                            14
           StackPushTest
                                                                                                                                                                            15
                           o test()
```

Рисунок 3.1 - Исходный код файла RunAll.java.

```
StackClient.java ×
src > stack > ① StackClient.java > ...
  1
       package stack;
  2
  3
       import java.util.Scanner;
  4
       public final class StackClient {
  5
           Run | Debug
           public final static void main(final String[] args) {
  6
  7
               Stack<String> stack = new Stack<String>();
  8
               Scanner scanner = new Scanner(System.in);
  9
               while (scanner.hasNext()) {
 10
 11
                   String item = scanner.next();
 12
 13
                    if (!item.equals(anObject: "-")) {
                        stack.push(item);
 14
                     else if (!stack.isEmpty()) {
 15
                        System.out.println(stack.pop());
 16
 17
 18
 19
               scanner.close();
 20
 21
               System.out.println("Stack size: " + Integer.toString(stack.size()));
 22
               System.out.println("Stack content: " + stack.toString());
 23
 24
 25
```

Рисунок 3.2 - Исходный код файла StackClient.java.

```
● Stack.java ×
src > stack > ① Stack.java > ...
 1 package stack;
      import java.util.NoSuchElementException;
      public final class Stack<Item> {
  6
         private Integer N = 0;
         private Node first = null;
  8
         private final class Node {
  9
 10
             private Item item;
 11
             private Node next;
 12
 13
             public Node(final Item item, final Node next) {
 14
                this.item = item;
                  this.next = next;
 15
 16
 17
 18
        public Stack() {
 19
 20
            assert check();
 22
          public final Boolean isEmpty() {
 23
 24
          return this.N == 0;
 25
 26
 27
         public final Integer size() {
          return this.N;
 28
 29
 30
 31
          public final void push(final Item item) {
            Node oldfirst = this.first;
 32
              first = new Node(item, oldfirst);
 33
 34
              ++this.N;
 35
             assert check();
 36
 37
         public final Item pop() {
 39
 40
            if (this.isEmpty())
 41
                throw new NoSuchElementException();
 42
 43
 44
            Item item = this.first.item;
 45
             this.first = this.first.next;
              --this.N;
 46
 47
 48
             assert check();
             return item;
 50
 51
         public final Item peek() {
           if (this.isEmpty()) {
 53
                 throw new NoSuchElementException();
 54
 56
             Item item = this.first.item;
 57
 58
              assert check();
 60
             return item;
 61
 63
          public final String toString() {
             StringBuilder sb = new StringBuilder();
 64
 65
              for (Node current = this.first; current != null; current = current.next)
 66
                 Item item = current.item;
 67
                  sb.append(item + " ");
 68
 70
              return sb.toString();
 71
 72
 73
```

Рисунок 3.3 - Исходный код файла Stack.java.

Продолжение рисунка 3.3.

public final void clear() {

74

b C d

d

C b

а

Ζ

Stack size: 2 Stack content: y x

PS D:\Documents\Visual Studio Code\Java\lab_11_3> [

```
75
                                 while (!this.isEmpty()) {
                   76
                                     this.pop();
                   77
                   78
                   79
                             private final Boolean check() {
                   80
                                 if (this.N == 0 && this.first != null) {
                   81
                                     return false;
                   82
                   83
                   84
                   85
                                 if (this.N != 0 && this.first == null) {
                   86
                                     return false;
                   87
                   88
                   89
                                 if (this.N > 1 && this.first.next == null) \{
                   90
                                     return false;
                   91
                   92
                   93
                                 Integer checkN = 0;
                   94
                   95
                                 for (Node x = this.first; x != null; x = x.next) {
                   96
                                      ++checkN;
                   97
                   98
                   99
                                 if (this.N != checkN) {
                  100
                                     return false;
                  101
                  102
                                 return true;
                  103
                  104
                  105
                  106
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS D:\Documents\Visual Studio Code\Java\lab 11 3> & 'C:\Program Files\Eclipse Foundation\jdk-11.0.12.7-hotspo
t\bin\java.exe' '@C:\Users\User\AppData\Local\Temp\cp_78sjyxqenhu36qyr22xnqvh8h.argfile' 'stack.StackClient'
```

Рисунок 3.4 - Результат работы кода StackClient.

```
    StackClearTest.java ×

 src > stack > • StackClearTest.java > ...
   1
        package stack;
    2
    3
        import static org.junit.Assert.assertEquals;
    4
        import java.util.ArrayList;
    5
        import java.util.Arrays;
    7
        import org.junit.After;
    9
        import org.junit.Before;
   10
        import org.junit.Test;
   11
> 12
        public final class StackClearTest {
   13
             Stack<String> stack = null;
            ArrayList<String> stackData = null;
   14
   15
   16
            @Before
   17
             public void beforTest() {
   18
                this.stack = new Stack<String>();
                this.stackData = new ArrayList<String>(Arrays.asList(...a: "a", "b", "c", "d", "e"));
   19
   20
                for (final String item : stackData) {
   21
                    this.stack.push(item);
   22
   23
   24
   25
   26
            @After
   27
             public void afterTest() {
   28
                this.stack.clear();
                this.stackData.clear();
   29
   30
   31
   32
            @Test
            public final void test() {
> 33
                this.stack.clear();
   34
   35
                assertEquals(this.stack.isEmpty(), actual: true);
   36
   37
   38
```

Рисунок 3.5 - Исходный код файла StackClearTest.java.

```
    StackPopTest.java ×

 src > stack > ① StackPopTest.java > ...
        package stack;
    1
    2
    3
        import static org.junit.Assert.assertEquals;
    4
    5
        import java.util.ArrayList;
        import java.util.Arrays;
    6
    7
        import java.util.Collections;
    8
    9
        import org.junit.After;
        import org.junit.Before;
   10
        import org.junit.Test;
   11
   12
        public final class StackPopTest {
≥ 13
            Stack<String> stack = null;
   14
   15
            ArrayList<String> stackData = null;
   16
            @Before
   17
            public void beforTest() {
   18
   19
                this.stack = new Stack<String>();
                this.stackData = new ArrayList<String>(Arrays.asList(...a: "a", "b", "c", "d", "e"));
   20
   21
   22
                for (final String item : stackData) {
   23
                     this.stack.push(item);
   24
   25
   26
                Collections.reverse(this.stackData);
   27
   28
            @After
   29
   30
            public void afterTest() {
   31
                this.stack.clear();
                this.stackData.clear();
   32
   33
   34
   35
            @Test
            public final void test() {
> 36
                 for (int i = 0, size = this.stackData.size(); i < size; ++i) {
   37
                     assertEquals(this.stack.isEmpty(), actual: false);
   38
   39
                     assertEquals(this.stack.size(), this.stackData.size() - i);
                     assertEquals(this.stack.peek(), this.stackData.get(i));
   40
   41
                     this.stack.pop();
   42
   43
   44
                 assertEquals(this.stack.isEmpty(), actual: true);
   45
   46
   47
```

Рисунок 3.6 - Исходный код файла StackPopTest.java.

```
    StackPushTest.java ×

 src > stack > • StackPushTest.java > ...
    1
        package stack;
    2
    3
        import static org.junit.Assert.assertEquals;
    4
        import java.util.ArrayList;
    5
    6
        import java.util.Arrays;
        import java.util.Collections;
    2
        import org.junit.After;
    9
        import org.junit.Before;
   10
   11
        import org.junit.Test;
   12
        public final class StackPushTest {
> 13
             Stack<String> stack = null;
   14
   15
             ArrayList<String> stackData = null;
   16
            @Before
   17
   18
             public void beforTest() {
   19
                this.stack = new Stack<String>();
                 this.stackData = new ArrayList<String>(Arrays.asList(...a: "a", "b", "c", "d", "e"));
   20
   21
   22
   23
             @After
   24
             public void afterTest() {
   25
                this.stack.clear();
                this.stackData.clear();
   26
   27
   28
             @Test
   29
> 30
             public final void test() {
                 assertEquals(this.stack.isEmpty(), actual: true);
   31
   32
   33
                 for (int i = 0, size = this.stackData.size(); i < size; ++i) {
                     this.stack.push(this.stackData.get(i));
   34
   35
                     assertEquals(this.stack.isEmpty(), actual: false);
   36
                     assertEquals(this.stack.size(), i + 1);
                     assertEquals(this.stack.peek(), this.stackData.get(i));
   37
   38
   39
                 Collections.reverse(this.stackData);
   40
                 StringBuilder sb = new StringBuilder();
   41
   42
                 for (final String item : stackData) {
   43
   44
                     sb.append(item + " ");
   45
   46
   47
                 assertEquals(this.stack.toString(), sb.toString());
```

Рисунок 3.7 - Исходный код файла StackPushTest.java.

Вывод: Освоил приемы тестирования кода на примере использования библиотеки JUnit.

48 49 50