

**Kauno technologijos universitetas**

Informatikos fakultetas

Demo užduotys ir bendriniai sąrašai

P175B014 Duomenų struktūrų pirmas laboratorinis darbas

**Projekto autorius**

Gustas Klevinskas

**Akademinė grupė**

IFF-8/7

Kaunas, 2019

Turinys

[Demo0\_Hello 3](#_Toc21564268)

[Demo1\_StringBuilder 3](#_Toc21564269)

[Demo0\_Basic 4](#_Toc21564270)

[Demo1\_Raster 6](#_Toc21564271)

[Demo2\_Transform 7](#_Toc21564272)

[Demo6\_Images 8](#_Toc21564273)

[Papildoma užduotis – kryžiukai nuliukai 8](#_Toc21564274)

[B dalis 9](#_Toc21564275)

[Testuoju savo parašytus parse() ir createString() metodus 9](#_Toc21564276)

[Naujos knygos pridėjimas 10](#_Toc21564277)

[Dar vienos naujos knygos pridėjimas. Šį kartą į 3 vietą 10](#_Toc21564278)

[Testavimas set(int k) metodo koreguojant antrąją knygą 11](#_Toc21564279)

[Testavimas remove(int k) metodo ištrinant 6 elementą (pradedant nuo 1) 11](#_Toc21564280)

[Filtravimas knygų, kurios senesnės nei 1950 12](#_Toc21564281)

[Pirmos individualios užduoties testavimas 12](#_Toc21564282)

[Antros individualios užduoties testavimas 12](#_Toc21564283)

[Trečios individualios užduoties testavimas 13](#_Toc21564284)

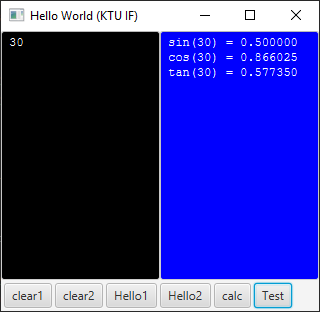
[Greitaveikos testavimas 1 13](#_Toc21564285)

[Greitaveikos testavimas 2 14](#_Toc21564286)

[Išvados 15](#_Toc21564287)

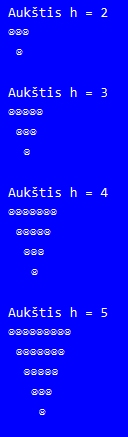
# Demo0\_Hello

Apskaičiuokite sin, cos ir tan reikšmes, kai duoti laipsniai.



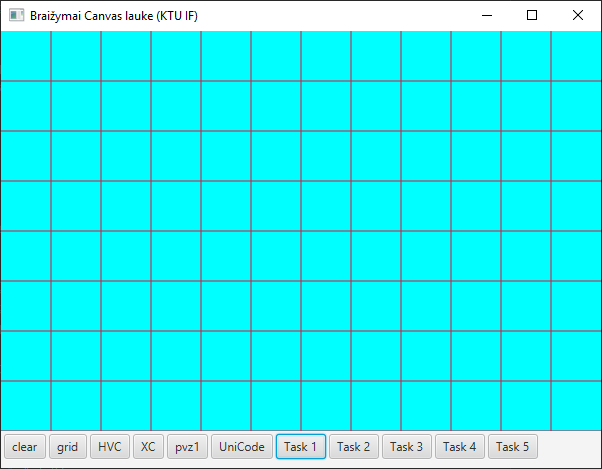
# Demo1\_StringBuilder

Sukurti metodą, kuris pieštų piramidę aukštyn kojom.

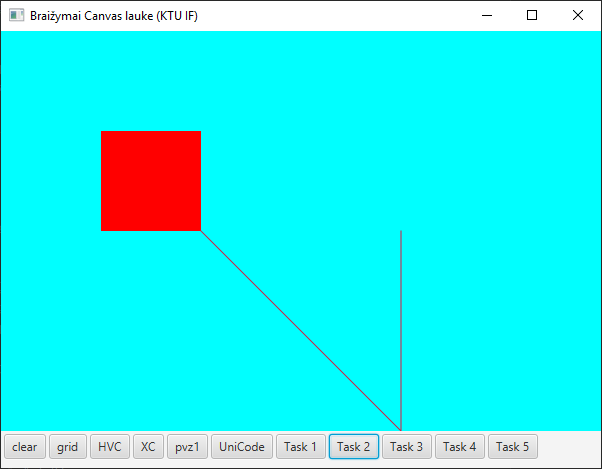


# Demo0\_Basic

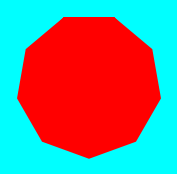
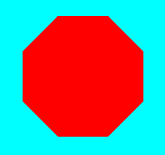
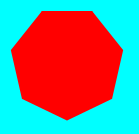
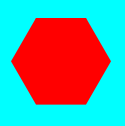
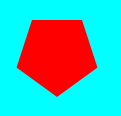
Langeliai.



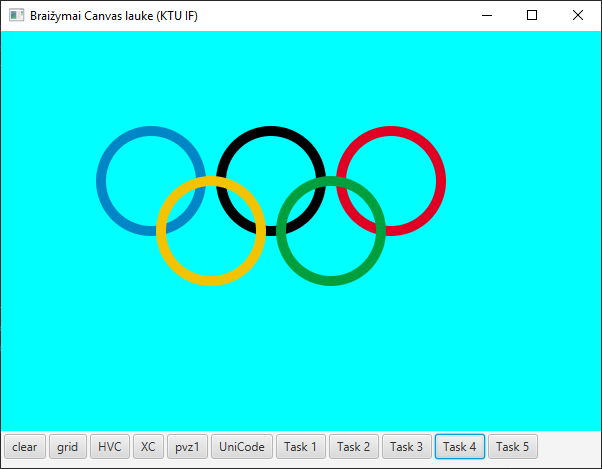
Nubrėžkite daugiakampį ir poliliniją.



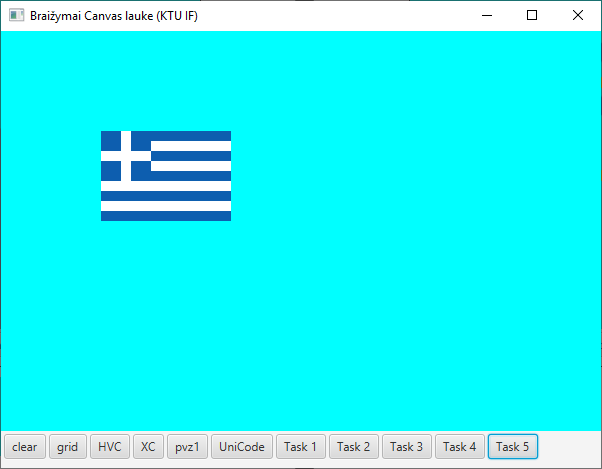
Nubrėžkite 3, 4, 5, ..., 9-kampius.



Nupieškite olimpinius žiedus.

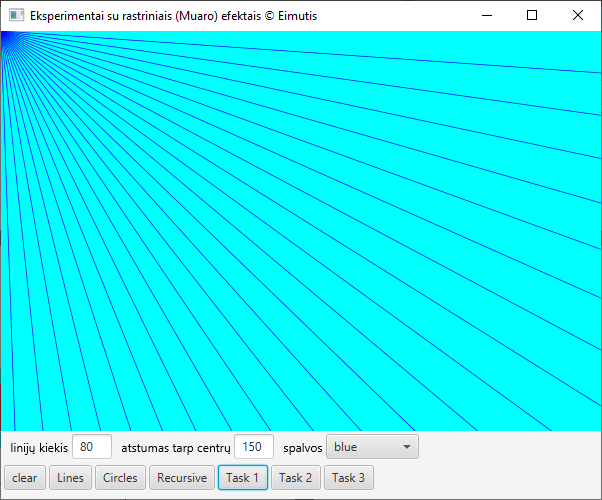


Nupieškite Graikijos vėliavą.

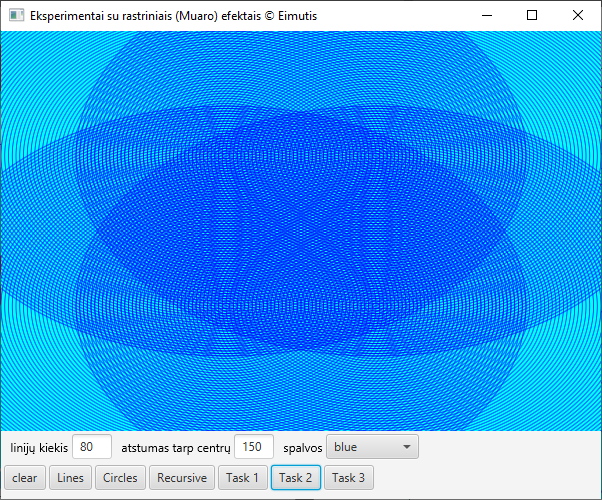


# Demo1\_Raster

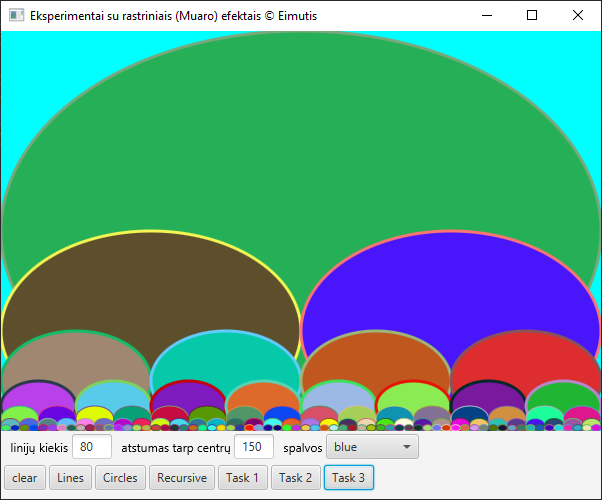
Nubrėžkite linijas iš kampo.



Nubrėžkite apskritimus, kurių centrai būtų keturiuose taškuose.

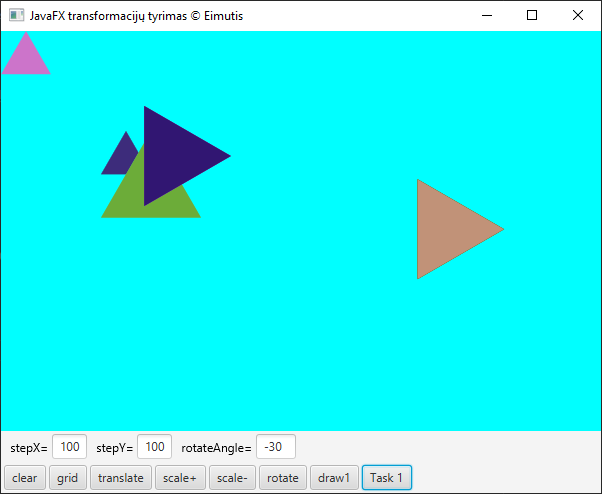


Rekursijos metu gaunami ovalai būtų nukreipti į apačią.



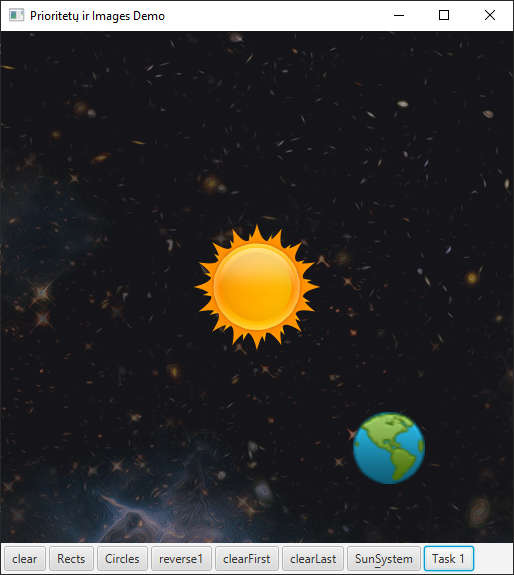
# Demo2\_Transform

Sukurkite savo figūrų rinkinį, kurį transformuosite.



# Demo6\_Images

Sustabdykite Žemės sukimąsi.



# Papildoma užduotis – kryžiukai nuliukai

Suprogramuoti žaidimą kryžiukai nuliukai. Žaidimas vyksta žmogus prieš žmogų. Pasidarius laiminčiajai eilutei ar stulpeliui ar įstrižainei nutraukiamas žaidimas ir pranešama, kas laimėjo. Lygiųjų atveju pranešama, kad lygiosios.

**Klasė TicTacToe**

**Laukai:**

String currentPlayer; // Saugomas dabartinio žaidėjo ženklas

String[] board; // Saugoma visa lenta vienoje eilutėje

**Metodai:**

drawGrid(); // Nupiešia 3x3 rėmelį

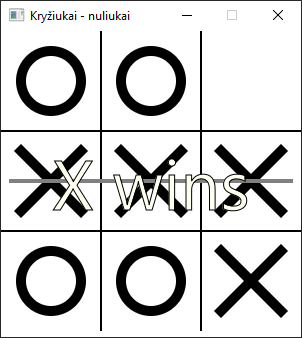
drawSymbol(double x, double y); // Nupiešia O arba X atitinkamai kieno eilė

drawWinningLine(int i); // Nubraukia liniją atitinkamai pagal laiminčiąją kombinaciją

drawText(String text); // Į ekraną išveda duotą tekstą

checkIfDone(); // Tikrina, ar žaidimas baigėsi (laimėjo O/X arba lygiosios)

**Rezultatas:**



# B dalis

2| -- Pradiniai duomenys --

3| Mark Twain - Moby Dick (2019) ☆0.0

4| Maironis - Metai (1894) ☆9.9

5| Antanas Škėma - Balta drobulė (1958) ☆6.9

6| Balys Sruoga - Kupstas (1980) ☆0.0

## Testuoju savo parašytus parse() ir createString() metodus

@Override  
public void parse(String data) {  
 try {  
 Scanner scanner = new Scanner(data);  
 scanner.useDelimiter("-");  
 author = scanner.next();  
 title = scanner.next();  
 publishingYear = scanner.nextInt();  
 rating = scanner.nextFloat();  
 } catch (InputMismatchException e) {  
 Ks.*ern*("Bad data format -> " + data);  
 } catch (NoSuchElementException e) {  
 Ks.*ern*("Missing data -> " + data);  
 }  
}

public String createString() {  
 return String.*format*("%s-%s-%d-%.1f", author, title, publishingYear, rating);  
}

private static void testParse() {  
 for (Object i : *books*.toArray()) {  
 Book parsedBook = new Book(((Book) i).createString());  
  
 Ks.*ou*("Before parsing: ");  
 Ks.*ouf*("%s\n", i);  
 Ks.*ou*("After parsing: ");  
 Ks.*ouf*("%s\n", parsedBook);  
 Ks.*oun*(((Book) i).compareTo(parsedBook) == 0 ? "PASS" : "FAIL");  
 }  
}

7| Before parsing: Mark Twain - Moby Dick (2019) ☆0.0

8| After parsing: Mark Twain - Moby Dick (2019) ☆0.0

9| PASS

10| Before parsing: Maironis - Metai (1894) ☆9.9

11| After parsing: Maironis - Metai (1894) ☆9.9

12| PASS

13| Before parsing: Antanas Škėma - Balta drobulė (1958) ☆6.9

14| After parsing: Antanas Škėma - Balta drobulė (1958) ☆6.9

15| PASS

16| Before parsing: Balys Sruoga - Kupstas (1980) ☆0.0

17| After parsing: Balys Sruoga - Kupstas (1980) ☆0.0

18| PASS

## Naujos knygos pridėjimas

@Override  
public boolean add(int k, E e) {  
 if (e == null)  
 return false;  
  
 if (k < 0 || k >= size)  
 return false;  
  
 if (k == 0) {  
 Node<E> top = first;  
 first = new Node<>(e, top);  
 size++;  
 return true;  
 }  
  
  
 current = first;  
  
 for (int i = 0; i < k - 1; i++)  
 current = current.next;  
  
 Node<E> remainingNodes = current.next;  
 current.next = new Node<>(e, remainingNodes);  
  
 size++;  
 return true;  
}

26| -- Adding a new book --

27| Mark Twain - Moby Dick (2019) ☆0.0

28| add 1 - title (2019) ☆0.0

29| Maironis - Metai (1894) ☆9.9

30| Antanas Škėma - Balta drobulė (1958) ☆6.9

31| Balys Sruoga - Kupstas (1980) ☆0.0

## Dar vienos naujos knygos pridėjimas. Šį kartą į 3 vietą

33| -- Adding a new book --

34| Mark Twain - Moby Dick (2019) ☆0.0

35| add 1 - title (2019) ☆0.0

36| Maironis - Metai (1894) ☆9.9

37| add 3 - title (2019) ☆0.0

38| Antanas Škėma - Balta drobulė (1958) ☆6.9

39| Balys Sruoga - Kupstas (1980) ☆0.0

## Testavimas set(int k) metodo koreguojant antrąją knygą

@Override  
public E set(int k, E e) {  
 current = first;  
  
 for (int i = 0; i < k; i++)  
 current = current.next;  
  
 E old = current.element;  
 current.element = e;  
  
 return old;  
}

41| -- Changing 2nd book --

42| Mark Twain - Moby Dick (2019) ☆0.0

43| set test - null (0) ☆0.0

44| Maironis - Metai (1894) ☆9.9

45| add 3 - title (2019) ☆0.0

46| Antanas Škėma - Balta drobulė (1958) ☆6.9

47| Balys Sruoga - Kupstas (1980) ☆0.0

## Testavimas remove(int k) metodo ištrinant 6 elementą (pradedant nuo 1)

@Override  
public E remove(int k) {  
 if (k < 0 || k >= size)  
 return null;  
  
 if (k == 0) {  
 E old = first.element;  
 first = first.next;  
 size--;  
 return old;  
 }  
  
  
 current = first;  
  
 for (int i = 0; i < k - 1; i++)  
 current = current.next;  
  
 E old = current.next.element;  
 current.next = current.next.next;  
 size--;  
 return old;  
}

49| -- Removing the 6th book --

50| Mark Twain - Moby Dick (2019) ☆0.0

51| set test - null (0) ☆0.0

52| Maironis - Metai (1894) ☆9.9

53| add 3 - title (2019) ☆0.0

54| Antanas Škėma - Balta drobulė (1958) ☆6.9

## Filtravimas knygų, kurios senesnės nei 1950

private static BookList getBooksFromYear(BookList bookList, int year) {  
 BookList newList = new BookList();  
  
 for (Object i : bookList.toArray())  
 if (((Book) i).getPublishingYear() >= year)  
 newList.add((Book) i);  
  
 return newList;  
}

56| -- Books from the year 1950: --

57| Mark Twain - Moby Dick (2019) ☆0.0

58| add 3 - title (2019) ☆0.0

59| Antanas Škėma - Balta drobulė (1958) ☆6.9

## Pirmos individualios užduoties testavimas

public void addLast(E e) {  
 if (e == null)  
 return;  
   
 if (first == null) {  
 first = new Node<>(e, first);  
 last = first;  
 } else {  
 Node<E> e1 = new Node<>(e, null);  
 last.next = e1;  
 last = e1;  
 }  
  
 size++;  
}

61| -- UŽDUOTIS 1 - addLast(E e) --

62| Mark Twain - Moby Dick (2019) ☆0.0

63| set test - null (0) ☆0.0

64| Maironis - Metai (1894) ☆9.9

65| add 3 - title (2019) ☆0.0

66| Antanas Škėma - Balta drobulė (1958) ☆6.9

## Antros individualios užduoties testavimas

public boolean remove(E e) {  
 if (e == null)  
 return false;  
  
 int index = 0;  
 current = first;  
  
 while (current.next != null) {  
 if (current.element.compareTo(e) == 0) {  
 remove(index);  
 return true;  
 }  
  
 current = current.next;  
 index++;  
 }  
  
 return false;  
}

68| -- UŽDUOTIS 2 - remove(E e) --

69| Mark Twain - Moby Dick (2019) ☆0.0

70| set test - null (0) ☆0.0

71| Maironis - Metai (1894) ☆9.9

72| Antanas Škėma - Balta drobulė (1958) ☆6.9

## Trečios individualios užduoties testavimas

public boolean addAll(int index, LinkedList<E> list) {  
 if (index < 0 || index >= size)  
 return false;  
  
 current = first;  
  
 for (int i = 0; i < index - 1; i++)  
 current = current.next;  
  
 list.last.next = current.next; // Append the remaining nodes to the given list  
 current.next = list.first; // Insert the first (and thus all) elements at index  
 size += list.size;  
 return true;  
}

74| -- UŽDUOTIS 3 - addAll(int index, LinkedList list) --

75| Mark Twain - Moby Dick (2019) ☆0.0

76| set test - null (0) ☆0.0

77| New Book - 1 (1995) ☆0.0

78| New Book - 2 (1997) ☆0.0

79| New Book - 3 (1999) ☆0.0

80| Maironis - Metai (1894) ☆9.9

81| Antanas Škėma - Balta drobulė (1958) ☆6.9

## Greitaveikos testavimas 1

private static void greitaveika1() {  
 final int size = 1000000;  
  
 Ks.*ouf*("\n%s\n", "------------------------------");  
 Ks.*ouf*("%s\n", "Greitaveika - UŽDUOTIS 1");  
 Ks.*ouf*("Masyvo dydis - %s\n", size);  
  
 int[] x = new int[size];  
 int[] y = new int[size];  
 double[] num1 = new double[size];  
 double[] num2 = new double[size];  
  
 for (int i = 0; i < size; i++) {  
 x[i] = new Random(Calendar.*getInstance*().getTimeInMillis()).nextInt();  
 y[i] = new Random(Calendar.*getInstance*().getTimeInMillis()).nextInt();  
 }  
  
  
 // Math.sqrt()  
 long begin = Calendar.*getInstance*().getTimeInMillis();  
  
 for (int i = 0; i < size; i++)  
 num1[i] = Math.*sqrt*(x[i] \* x[i] + y[i] \* y[i]);  
 long end = Calendar.*getInstance*().getTimeInMillis();  
  
 Ks.*ouf*("%d ms su Math.sqrt().\n", end - begin);  
  
  
 // Math.hypot()  
 begin = Calendar.*getInstance*().getTimeInMillis();  
  
 for (int i = 0; i < size; i++)  
 num1[i] = Math.*hypot*(x[i], y[i]);  
 end = Calendar.*getInstance*().getTimeInMillis();  
  
 Ks.*ouf*("%d ms su Math.hypot().\n", end - begin);  
}

Greitaveika - UŽDUOTIS 1

Masyvo dydis - 1000000

15 ms su Math.sqrt().

44 ms su Math.hypot().

## Greitaveikos testavimas 2

private static void greitaveika2() {  
 final int size = 100000;  
 final int amountOfOperations = 1000;  
  
 Ks.*ouf*("\n%s\n", "------------------------------");  
 Ks.*ouf*("%s\n", "Greitaveika - UŽDUOTIS 2");  
 Ks.*ouf*("LinkedList dydis - %s\n", size);  
  
 LinkedList<Integer> linkedList = new LinkedList<>();  
 for (int i = 0; i < size; i++)  
 linkedList.add(new Random(Calendar.*getInstance*().getTimeInMillis()).nextInt(100000000));  
  
 int[] indexes = new int[amountOfOperations];  
 for (int i = 0; i < amountOfOperations; i++)  
 indexes[i] = new Random(Calendar.*getInstance*().getTimeInMillis()).nextInt(size);  
  
 long sum = 0;  
 // Repeat operations to get an average  
 for (int i = 0; i < amountOfOperations; i++) {  
 long begin = Calendar.*getInstance*().getTimeInMillis();  
  
 linkedList.indexOf(indexes[i]);  
  
 long end = Calendar.*getInstance*().getTimeInMillis();  
 sum += end - begin;  
 }  
 Ks.*ouf*("%d ms su indexOf(); operacijų kiekis - %d.\n", sum, amountOfOperations);  
  
  
 sum = 0;  
 // Repeat operations to get an average  
 for (int i = 0; i < amountOfOperations; i++) {  
 long begin = Calendar.*getInstance*().getTimeInMillis();  
  
 linkedList.lastIndexOf(indexes[i]);  
  
 long end = Calendar.*getInstance*().getTimeInMillis();  
 sum += end - begin;  
 }  
 Ks.*ouf*("%d ms su lastIndexOf(); operacijų kiekis - %d.\n", sum, amountOfOperations);  
}

Greitaveika - UŽDUOTIS 2

LinkedList dydis - 100000

3710 ms su indexOf(); operacijų kiekis - 1000.

3590 ms su lastIndexOf(); operacijų kiekis - 1000.

# Išvados

Darydamas A dalį visai neblogai apsipratau su JavaFX aplinka, o B dalyje prisiminiau vienkrypčio sąrašo ypatybės.

Daugiausia laiko skyriau A daliai, nes užduotys buvo neaiškiai sužymėtos ir jas buvo sunku rasti bei suprasti, koks jų tikslas.

Žaidimą pavyko atlikti gerai, riktų jame neturėtų būti.