**KAUNO TECHNOLOGIJOS UNIVERSITETAS**

Duomenų struktūros (P175B014)

Laboratorinio darbo Nr.1 ataskaita

Atliko **Martynas Kemežys** gr. IF-8/1

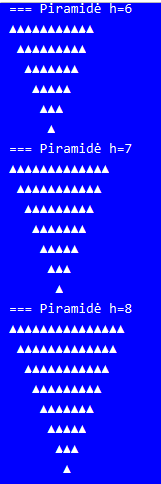
Priėmė lekt. **Giedrius Paulikas**

**Lab1A. Įvado dalis – JAVA FX**

Sudarykite naują metodą funkcijų sin, cos ir tan skaičiavimui:

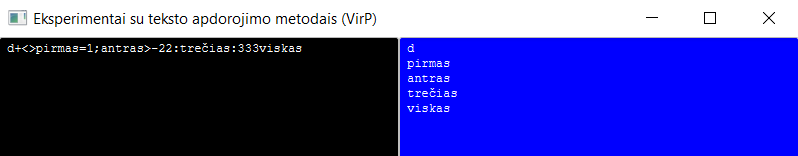


Sudarykite pasirinktinai metodus apverstai ar pasuktai piramidei spausdinti

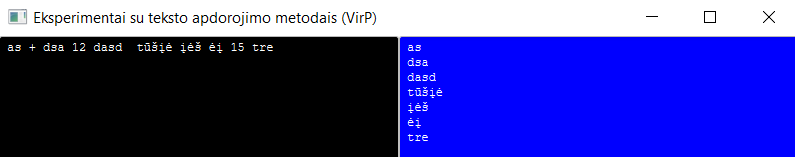


**Pakeiskite šabloną, kad jis atapžintų:**

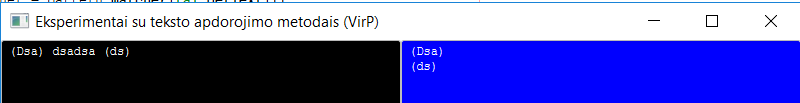
a - žodžius su lotyniškomis raidėmis



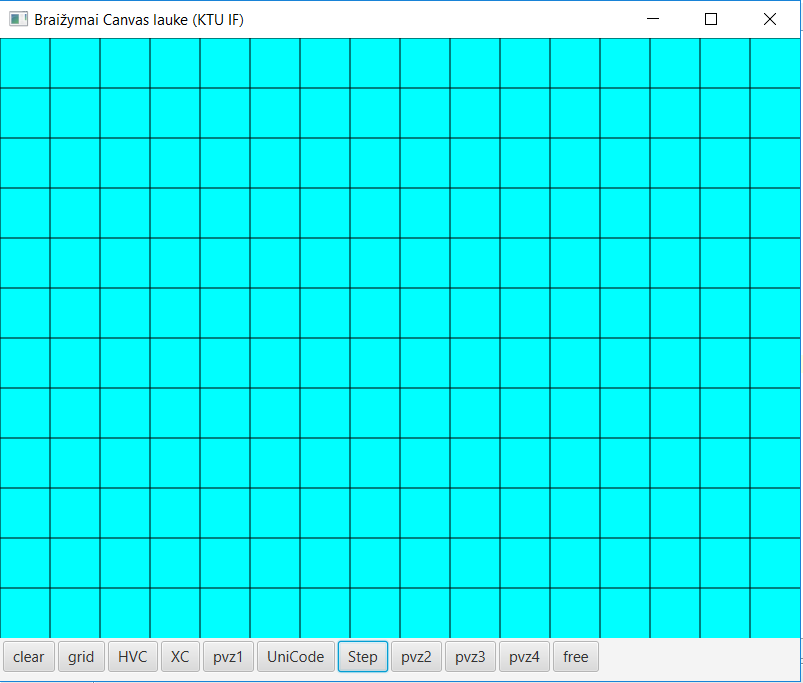
b - žodžius su lietuviškomis raidėmis



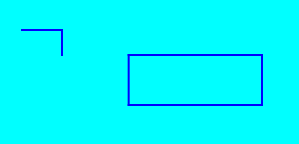
c - informaciją tarp skliaustų ()



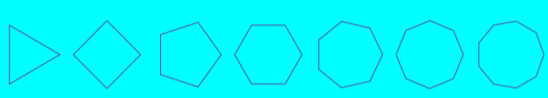
Plonomis linijomis su žingsniu step=50 nubrėžkite tinklelį:



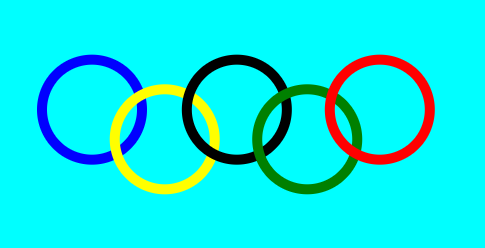
Nubrėžkite polilinijas ir poligonus:



Nubrėžkite taisyklingus 3, 4, 5, ..., 9-kampius:



Nubrėžkite žiedus:



Pasirinktinai nubrėžkite savo tematiką:



public void handle(long now)

{

double t = (now - startNanoTime) / 1\_000\_000\_000.0;

earthView.setX(xc - earth.getWidth()/2 + radius \* Math.cos(t));

earthView.setY(yc - earth.getWidth()/2 + radius \* Math.sin(t));

marsView.setX(xc - mars.getWidth()/2 + radius2 \* Math.cos(t - 2));

marsView.setY(yc - mars.getWidth()/2 + radius2 \* Math.sin(t - 2));

}

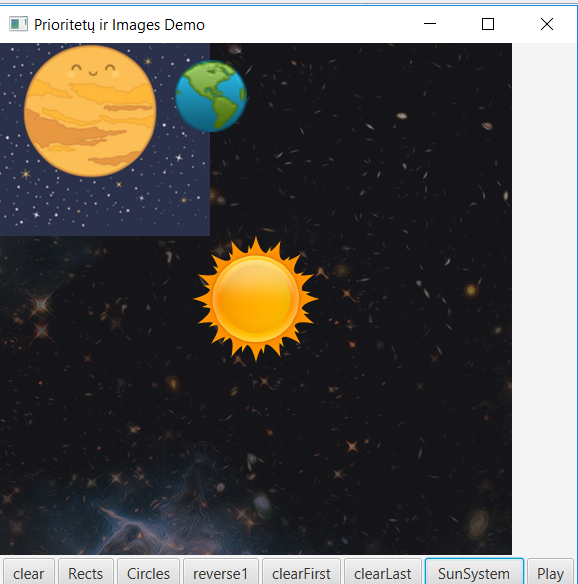
* Paaiškinkite kintamojo t skaičiavime naudojamą konstantą

Nanosekundės konversija į sekundę

* Paskaičiuokite per kiek laiko apskrieja žemė aplink saulę

Per 2\*PI sekundžių, nes 2\*PI radianų yra 360 laipsnių.

* Papildykite erdvę kitais kosminiais ar fantastiniais kūnais



**Lab1B. Susieti bendriniai sąrašai**

**Tiriamas metodas – ListKTU.addLast(E e)**

Pradiniai duomenys:

29| 0: Renault Laguna 1997 50000 1700,0

30| 1: Audi A4 2003 115900 7500,0

31| 2: Renault Megane 2001 20000 3500,0

32| 3: Toyota Corolla 2001 20000 8500,8

33| 4: Renault Laguna 2001 115900 7500,0

34| 5: Renault Megane 1946 365100 9500,0 Netinkami gamybos metai, turi būti [1994:2019]

35| 6: Honda Civic 2007 36400 8500,3

36| 7: Audi A4 2003 115900 7500,0

public boolean addLast(E e) {

if (e == null) {

return false; // nuliniai objektai nepriimami

}

if (first == null) {

first = new Node<>(e, first);

last = first;

} else {

Node<E> e1 = new Node(e, null);

last.next = e1;

last = e1;

}

size++;

return true;

}

Gauti rezultatai:

29| 0: Renault Laguna 1997 50000 1700,0

30| 1: Audi A4 2003 115900 7500,0

31| 2: Renault Megane 2001 20000 3500,0

32| 3: Toyota Corolla 2001 20000 8500,8

33| 4: Renault Laguna 2001 115900 7500,0

34| 5: Renault Megane 1946 365100 9500,0 Netinkami gamybos metai, turi būti [1994:2019]

35| 6: Honda Civic 2007 36400 8500,3

36| 7: Audi A4 2003 115900 7500,0

37| 8: Audi A8 2013 115900 7500,0

Šiuo atveju prie sąrašo buvo pridėtas objektas “Audi A8” ir jis buvo prikabintas į sąrašo pabaigą.

**Tiriamas metodas – ListKTU.removeLastOccurrence(E e)**

Pradiniai duomenys:

29| 0: Renault Laguna 1997 50000 1700,0

30| 1: Audi A4 2003 115900 7500,0

31| 2: Renault Megane 2001 20000 3500,0

32| 3: Toyota Corolla 2001 20000 8500,8

33| 4: Renault Laguna 2001 115900 7500,0

34| 5: Renault Megane 1946 365100 9500,0 Netinkami gamybos metai, turi būti [1994:2019]

35| 6: Honda Civic 2007 36400 8500,3

36| 7: Audi A4 2003 115900 7500,0

public boolean removeLastOccurrence(E e) {

Node<E> temp = first, previous = null;

Node<E> test = null;

if (e == null) {

return false; // nuliniai objektai nepriimami

}

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

test = first.findNode(i);

if(test.element.equals(e) && e != first.element)

{

previous = first.findNode(i-1); //prieš audi

temp = test.next; //kas už audi

}

}

if(previous == null)

{

first = first.next;

size --;

return true;

}

if (temp == null)

{

previous.next = null;

size --;

return true;

}

previous.next = temp;

size--;

return true;

}

Gauti rezultatai:

29| 0: Renault Laguna 1997 50000 1700,0

30| 1: Audi A4 2003 115900 7500,0

31| 2: Renault Megane 2001 20000 3500,0

32| 3: Toyota Corolla 2001 20000 8500,8

33| 4: Renault Laguna 2001 115900 7500,0

34| 5: Renault Megane 1946 365100 9500,0 Netinkami gamybos metai, turi būti [1994:2019]

35| 6: Honda Civic 2007 36400 8500,3

Iš sąrašo buvo ištrintas pirmas pasitaikęs „Audi a4“ objektas.

**Tiriamas metodas – ListKTU.subList(int fromIndex, int toIndex)**

public List<E> subList(int fromIndex, int toIndex)

{

LinkedList list = new LinkedList();

Node<E> test = null;

if(fromIndex > toIndex || toIndex > size) return null;

Ks.oun("=======subList veikimas========");

for (int i = fromIndex; i <= toIndex; i++) {

test = first.findNode(i);

list.add(test.element);

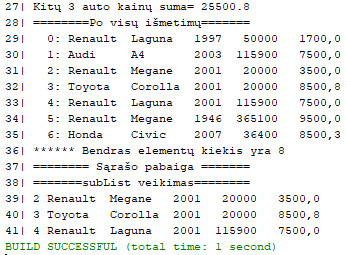
Ks.oun(i + " " +test.element);

}

return list;

}

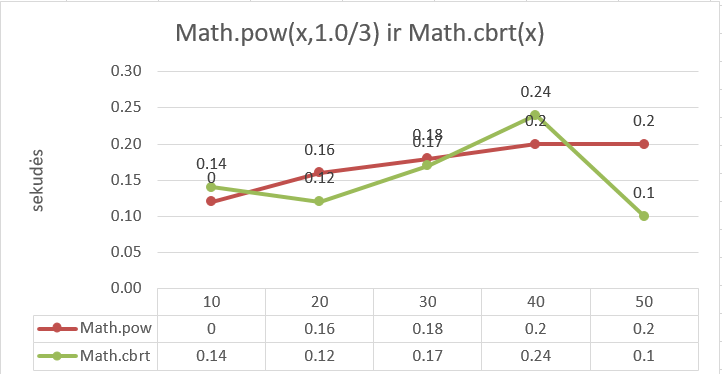
Rezultatai:



Šis metodas pateikia sąrašo objektus, nuo tam tikros pradžios iki tam tikros pabaigos. Šiuo atveju buvo panaudota komanda subList(2,4), ir rezultate buvo pateikta sąrašo 2,3,4 elementai.

**Greitaveikos tyrimas**

Reikia palyginti Math.pow(x,1.0/3) ir Math.cbrt(x) metodų veikimo efektyvumą.



**Išvados**

Labaratorinio darbo metu susidūriau su tam tikrais Java kalbos niuansais, kurie nepasireikšdavo, pavyzdžiui, C# kalboje. Pavyzdys : ref ir out naudojimas metoduose. Java kalboje tai neegzistuoja, tad teko ieškoti kitų sprendimo būdų. Darydamas A dalį visai neblogai apsipratau su JavaFX aplinka, o B dalyje prisiminiau vienkrypčio sąrašo ypatybės. Daugiausia laiko buvo skirta A daliai, mat pati aplinka buvo labai nežinoma, tad reikėjo nemažai mokytis naujų dalykų.

Apytiklė darbo atlikimo trukmė: A dalis – 30 val., B dalis – 8 val.