**Laboratorinis darbas Nr.2: Bendriniai sąrašai**

Atliko: Tadas Laurinaitis, IFF 6/8, 2017 12 18

# 1.1 Darbo užduotis

*Duota:* du paketai **studijosKTU** (su atskiru aprašymu) ir **Labor2** , kuriuose yra pateiktos toliau naudojamos sisteminės klasės ir demo variantas laboratorinio darbo vykdymui.

*Reikia sukurti:* naują paketą **Lab2Pavarde** su klasėmis individualioms užduotims spręsti.

*Tyrimo ir analizės dalis:*

1. Išnagrinėti elementarios klasės **Automobilis** struktūrą, išbandyti jos metodus;
2. Išnagrinėti apjungiančios klasės **AutomobiliuTurgus** struktūrą, išbandyti jo metodus;
3. Išbandyti klasių **ListKTU** ir **ListKTUx** metodus, ištirti sąrašo metodų sąveiką su **KTUable** tipo elementais; naudojamasi klasės **BandymaiAutomobiliųi** pavyzdžiu;
4. Išbandyti greitaveikos tyrimą specialios klasės pagalba;

*Individuali klasių konstravimo dalis:*

1. Pagal duotą **Automobilio** klasės pavyzdį sukurti individualiai pasirinktas elemento klases (4-5 komponentai), tenkinančias **KTUable** interfeisą; programinį kodą rašyti į individualų paketą **Lab2Pavarde**.
2. Patikrinti individualios klasės veikimą testo klasės pagalba;
3. Sudaryti individualių elementų apskaitos klasę, kurioje būtų elementų peržiūra ir jų atranka pagal įvairius kriterijus;
4. Sudaryti elementų apskaitos klasės demonstracinius metodus;
5. Realizuoti ListKTU metodus **add(int k, Data data), set(int k, Data data), remove(int k);**
6. Realizuoti individualiai nurodytus metodus;
7. Atliekamas individualiai nurodytų metodų greitaveikos tyrimas.
8. Sunaudojamos atminties kiekio įvertinimas.

Pateikiami **studento sukurtų metodų** eksperimentai (viso 3-5 variantai)

|  |
| --- |
| ***Tiriamas metodas –* ListKTU.** Equals() |
| public boolean equals(Object o){  System.out.println(o);  if(o == null){  System.out.println("objektas yra null");  return false;  }  if(o.getClass() == this.getClass()){  final ListKTU oLK = (ListKTU)o;  System.out.println("teisinga klase");  if(first.element == oLK.first.element){  System.out.println("lygus");  return true;  }  else{  System.out.println("nelygus: 1. " +first.element +"; 2. " +oLK.first.element);  return false;  }  }  else{  System.out.println("nelygios klases");  return false;  }  } |
| **Gauti rezultatai** |
| lygus |
| **Komentarai:**  Lyginamas dvieju sarasu ekvivalentiskumas |
| ***Tiriamas metodas --* ListKTU.** removeFirstOccurence() |
| public boolean removeFirstOccurence(Object o){  boolean removed = false;  int count = 0;  for (Node<E> elem = first; elem != null; elem = elem.next) {  if(elem.element == o){  Node<E> temp1 = first.findNode(count);  if(count != 0){  Node<E> temp2 = first.findNode(count - 1);  temp2.next = temp1.next;  temp1.next = null;  size--;  return true;  }  else if(count == 0){  first = first.next;  size--;  System.out.println("ascia");  return true;  }  }  else  count++;  }  if(removed = false){  System.out.println("Neistrinta");  }  return removed;  } |
| **Gauti rezultatai** |
| **Komentarai:**  Ištrinamas pirmas sutiktas objektas, lygus nurodytam objektui. |
| ***Tiriamas metodas --* ListKTU.**removeLast() |
| public E removeLast(){  if(size <= 0){  return null;  }  else if(size > 2){  //Node<E> node1 = last;  Node<E> node1 = first.findNode(size-2);  E elem = last.element;  node1.next = null;  size--;  return elem;  }  else{  E elem = last.element;  first = last = null;  size--;  return elem;  }  } |
| **Gauti rezultatai** |
| **Komentarai:**  Pasalinamas ir grazinamas paskutinis elementas. |

# 1.3 Studento sukurtos klasės ir jų metodai

**Sprendimas:** aprašomos sudarytos klasės ir metodai *-> pilni kodai pateikiami kartu su projektu*.

**public class Kate** // Pagal šią klasę kuriamas Kates objektas

**Laukai:**

private String vardas;

private String veisle;

private String lytis;

private double amzius;

private double kaina; private int assists;

**Metodai:**

public String getName() //gražinamas Player vardas

public String getTeam() //gražinama Player komanda

public String getPosition() //gražinama Player pozicija

public String getChampion() //gražinamas Player čempionas

public int getKills() //gražinamas Player kill‘ų skaičius

public int getAssists() //gražinamas Player assist‘ų skaičius

**public class KaciuTurgus**// Šioje klasėje atliekami veiksmai su katemis

**public class KaciuBandymai**// Šioje klasėje atliekami veiksmai su katemis

**public class MetoduTestavimas**// Šioje klasėje atliekami parasytu metodu bandymai.

**Metodai:**

public Kate create(String dataString)

public final void parse(String dataString)

public String validate()

public String toString()

public String getVardas()

public String getLytis()

public String getVeisle()

public double getAmzius()

public double getKaina()

public void setKaina(double kaina)

public int compareTo(Kate a)

**Pradinis duomenų rinkinys:**

Sudaromas metodo create() pagalba

**Gauti rezultatai:**

\*\*\*Klaida 1: Blogas duomenų formatas apie kates -> Morka Egzotas Mot 0.2 6500

\*\*\*Klaida 2: Blogas duomenų formatas apie kates -> Flopas Trumpauodegis Mot 2.3 36000

1| Smikis | Meskenas | Vyr| 3.3| 477.0|

2| Etiopija | Trumpauodegis | Mot| 2.3| 360.0|

3| Morka | Egzotas | Mot| 0.0| 0.0|

4| Flopas | Trumpauodegis | Mot| 0.0| 0.0|

# Išvados

Labai patogu, kad nereikia pačiam rašyti rūšiavimo burbuliuku metodų. Java yra grynai objektinė kalba, todėl kodas atrodo tvarkingesnis, nėra atskirų, išmėtytų funkcijų.

String objektas turi savyje dažnai naudotinus metodus: replace, split, toUpperCase, toLowerCase.

*Apie NetBeans:* Turi gera klaidų taisymą/rodymą. Gali suformatuoti negražiai parašytą kodą. Turi gražią kodo spalvinimo paletę. Automatiškai atnaujina išorėje pakeistą kodą.

Bendros laboratorinio darbo atlikimo laiko sąnaudos buvo 5 valandų.