Федеральное государственное автономное образовательное учреждение
высшего образования
«Национальный исследовательский университет ИТМО»
Факультет Программной Инженерии и Компьютерной Техники

ИТМО

Вариант №5027.5
Лабораторная работа №4
по дисциплине
Программирование

Выполнил Студент группы P3112
Кобелев Роман Павлович
Преподаватель:
Гаврилов Антон Валерьевич.

г. Санкт-Петербург
2022г.

1 Задание

Описание предметной области, по которой должна быть построена объектная модель:

Нужно сказать, что гравитонный телескоп вовсе не похож на обычный оптический телескоп, в который мы можем рассматривать звезды или планеты собственными глазами. Гравитонный телескоп представляет собой сложное устройство, напоминающее телевизор, снабженный большой, расширяющейся к концу трубой, которая легко поворачивается и может быть направлена в любую часть лунного неба. Эта труба, или рупор, представляет собой сплетение тончайших металлических проводов и является антенной, улавливающей волны тяготения, или так называемые гравитоны, распространяющиеся во все стороны от любого космического тела. Как только эта трубчатая, или, как ее иначе называют, рупорная, антенна улавливает волны тяготения, телевизионный экран освещается, и на нем возникает изображение кривой линии. По степени кривизны и по ее положению на экране можно судить о величине наблюдаемого космического тела. Включив гравитонный локатор, можно тут же получить сведения о точном расстоянии до этого тела, а также о скорости его движения. С тех пор как главный гравитонный телескоп давилонской обсерватории был направлен в сторону Земли, астрономам удалось обнаружить несколько мелких космических тел. Не только размеры, но и скорость их движения свидетельствовали о том, что это были обыкновенные метеоры. Вскоре, однако, по соседству с планетой Землей было обнаружено космическое тело, поведение которого показалось астрономам несколько странным. Тело это удалялось от Земли, но скорость его почему-то не уменьшалась, а увеличивалась. Это противоречило законам небесной механики, согласно которым скорость тела, движущегося вблизи планеты, могла увеличиваться только в том случае, если бы тело приближалось к планете. Поскольку же тело не приближалось, а удалялось от Земли, скорость его должна была уменьшаться. Такое ускорение движения могло быть объяснено притяжением какой-нибудь другой крупной планеты, но, поскольку вблизи Земли никакой другой планеты не было, оставалось предположить, что обнаруженное тело приобретало ускорение под влиянием какой-то внутренней, то есть находящейся в нем самом, силы. Источником такой силы мог быть работающий реактивный двигатель, и если это так, то обнаруженное космическое тело было не что иное, как космическая ракета. Продолжив свои наблюдения, давилонские астрономы убедились, что завладевший их вниманием космический предмет постепенно приобрел скорость, достаточную для того, чтоб со временем выйти из сферы земного притяжения. Рассчитав траекторию, то есть линию полета этого перемещающегося в межпланетном пространстве тела, астрономы убедились, что оно направляется к Луне. Об этом немедленно сообщили господину Спрутсу. Господин Спрутс отдал распоряжение продолжать астрономические наблюдения, после чего позвонил по телефону главному полицейскому комиссару Ржиглю и сказал, что ожидается прибытие космического корабля с коротышками на борту, с которыми необходимо как можно скорей разделаться, поскольку они намерены сеять повсюду гигантские семена и подстрекать бедняков к неповиновению богачам. Главный полицейский комиссар Ржигль сказал, что все необходимые меры будут приняты, но просил сообщить о времени ожидаемого прибытия космического корабля на Луну, о месте предполагаемой высадки космонавтов и об их примерном количестве. Между тем лунные астрономы продолжали свои наблюдения и вскоре заметили, что космическое тело вышло из сферы притяжения Земли. Полет его, однако, был не совсем точен, и одно время казалось, что оно пролетит мимо Луны, но вскоре было замечено, что оно несколько замедлило свой полет и совершило небольшой поворот, в результате чего курс его стал более точным. Такой маневр в космосе могло совершить только управляемое тело, и у давилонских астрономов не оставалось больше сомнений в том, что они имеют дело с космической ракетой, а не с какой-нибудь случайной кометой или метеором. Теперь космическая ракета была уже в непосредственной близости от Луны, и по показаниям гравитонных приборов можно было довольно точно определить ее вес и объем. Сопоставив полученные цифровые материалы и произведя некоторые расчеты, давилонские астрономы пришли к заключению, что в ракете могло помещаться от десяти до двадцати, а может быть, даже и до тридцати пассажиров. Пока невозможно было указать примерное место посадки космического корабля, так как, приблизившись на достаточное расстояние, он не пошел на посадку, а начал круговой облет Луны. Астрономы тотчас догадались, что прилетевшие космонавты решили выбрать наиболее удобное место для посадки и поэтому перешли на орбитальный, то есть круговой, полет. Догадка лунных астрономов была верна. Знайка, Фуксия и Селедочка заранее условились, что не станут производить посадку до тех пор, пока не обнаружат на лунной поверхности космического корабля, на котором прилетели Незнайка и Пончик. Они знали, что корабль этот следует искать в районе лунного моря Ясности, но им все же понадобилось совершить вокруг Луны не менее двух десятков витков, прежде чем удалось обнаружить ракету, одиноко торчавшую на берегу окаменевшего моря.
Программа должна удовлетворять следующим требованиям:

1. В программе должны быть реализованы 2 собственных класса исключений (checked и unchecked), а

также обработка исключений этих классов.

2. В программу необходимо добавить использование локальных, анонимных и вложенных классов (static и non-static).

Порядок выполнения работы:

1. Доработать объектную модель приложения.
2. Перерисовать диаграмму классов в соответствии с внесёнными в модель изменениями.
3. Согласовать с преподавателем изменения, внесённые в модель.
4. Модифицировать программу в соответствии с внесёнными в модель изменениями.

2 Диаграмма классов объектной модели

3 Исходный код программы

inappropriateVelocity.java

```java
package exceptions;

public class inappropriateVelocity extends RuntimeException{
   public inappropriateVelocity(){
      super("Нельзя лететьстакойскоростью, дурак.\Штрафn -
         ↪ увасотбираетсякошкадевочка-!");
   }
}
```

NegativeNumberOfPeopleException.java

```java
package exceptions;

public class NegativeNumberOfPeopleException extends Exception{
   public NegativeNumberOfPeopleException(String message){
      super(message);
   }
}
```

ThingInSpace.java

```java
package Interfaces;

import Objects.Direction;
import Objects.SizeOfGravityWaves;

public interface ThingInSpace {
   int getAccelerationInSpace();
   Direction getDirection();
   void setDirection(Direction newDirection);
   void setAccelerationInSpace(int newAcceleration);
   SizeOfGravityWaves getSize();
}
```

Alive.java

```java
package Interfaces;

import Live.Mood;

public interface Alive {
   void dead();
   Mood getMood();
   void setMood(Mood newMood);
   void listen();

}
```

NotAlive.java

```java
package Interfaces;

import Objects.Materials;

```

```java
public interface NotAlive {
   Materials getMaterial();
}
```

StateNameInterface.java

```java
package Interfaces;

import Others.State;

public interface StateNameInterface {
   String getName();
   State getState();
   void setState(State newState);
}
```

Astronomers.java

```java
package Live;
import Others.State;
import Objects.*;

public class Astronomers extends Human {
   public Astronomers(String name, Mood mood, State state, String nativeLanguage){
      super(name, mood, state, nativeLanguage);
   }

   @Override
   public void gotIntelligence() {
      System.out.println(getName()+"имееет интеллект");
   }

   public boolean checkObj(Spaceship spc){
      return (spc.getState()==State.FLY);

   }
   public boolean checkMultiplicityAndUsing(Telescope tls){
      return (tls.getState()==State.USE && tls.getMultiplicity() ⩾8);
   }

   public String getDestination(Spaceship spc, Telescope tls){
      if (checkObj(spc) && checkMultiplicityAndUsing(tls)){
         int[] a = spc.getCoordinates();
         int Velocity = (a[1] - a[0])/(a[2]*60);

         if (Velocity >1600){
            setMood(Mood.PANIC);
            setState(State.REPORT);
            return "MOON";
         }
         else
         {
            setMood(Mood.CALM);
            return "Something else";
         }
      }
      else{
```

```
40          if (!checkMultiplicityAndUsing(tls)){
41             System.out.println(getName() + ": Возьмидругойтелескоп!");
42          }
43          else{
44             if (!checkObj(spc)){
45                System.out.println(getName() + ": Тамничегонет,
                  ↪ чтотыхочешьпросмотреть?");
46             }
47          }
48
49          return "";
50       }
51
52    }
53
54
55
56
57 }
```

Astronomers4.java

```
1  package Live;
2
3
4  import Objects.*;
5  import Others.State;
6
7
8  public class Astronomers4 extends Astronomers{
9     private boolean suspicions = false;
10    public Astronomers4(String name, Mood mood, State state, String nativeLanguage){
11       super(name, mood, state, nativeLanguage);
12    }
13
14    public boolean checkObj(Spaceship4 spc){
15       return (spc.getState()==State.FLY);
16
17    }
18    public boolean checkObj(Meteorite mt){
19       return (mt.getState()==State.FLY);
20
21    }
22    public boolean checkMultiplicityAndUsing(Telescope4 tls){
23       return (tls.getState()==State.USE && tls.getMultiplicity() ⩾8);
24    }
25
26    public boolean isScreenOn(Screen scr){
27       return(scr.getState()==State.USE);
28    }
29
30    public boolean isGravityLocatorOn(GravityLocator gl){
31       return(gl.getState()==State.USE);
32    }
33
34
35    public String exploreObject(Spaceship4 spc, Telescope4 tls, Screen scr,
```

```
        ↪ GravityLocator gl){
36      if (checkObj(spc) && checkMultiplicityAndUsing(tls) && isScreenOn(scr) &&
            ↪ isGravityLocatorOn(gl)){
37          int Velocity = gl.getVelocity(spc);
38          if (scr.wavesIntoLine(tls, spc)== SizeOfGravityWaves.BIG && spc.
                ↪ getDirection()==Direction.FROMPLANET && Velocity >1600){
39              setState(State.TALK);
40              System.out.println(getName() + ": 
                    ↪ Этопротиворечитзаконамнебесноймеханики,
                    ↪ согласнокоторымскоростьтела, движущегосявблизипланеты,
                    ↪ моглаувеличиватьсятольковтомслучае,
                    ↪ еслибытелоприближалоськпланете");
41              System.out.println(getName()+": Вэтойштукеполюбомуестьдвигатель!");
42              suspicions = true;
43              setMood(Mood.PANIC);
44              setState(State.REPORT);
45              return "MOON";
46          }
47          else
48          {
49              return "Something else";
50          }
51      }
52      else{
53          setState(State.TALK);
54          System.out.println(getName()+": Мынеможемисследоватьлунноенебо,
                ↪ унаснедостаточнооборудованиядляэтого!");
55          setState(State.SIT);
56          return "Nothing";
57      }
58
59  }
60  public String exploreObject(Meteorite mt, Telescope4 tls, Screen scr,
        ↪ GravityLocator gl){
61      int Velocity = gl.getVelocity(mt);
62      if (checkObj(mt) && checkMultiplicityAndUsing(tls) && isScreenOn(scr) &&
            ↪ isGravityLocatorOn(gl)){
63          if (scr.wavesIntoLine(tls, mt)== SizeOfGravityWaves.BIG && mt.getDirection
                ↪ ()==Direction.FROMPLANET && Velocity >1600){
64              setState(State.TALK);
65              System.out.println(getName() + ": 
                    ↪ Этопротиворечитзаконамнебесноймеханики,
                    ↪ согласнокоторымскоростьтела, движущегосявблизипланеты,
                    ↪ моглаувеличиватьсятольковтомслучае,
                    ↪ еслибытелоприближалоськпланете");
66              System.out.println(getName()+": Вэтойштукеполюбомуестьдвигатель!");
67              suspicions = true;
68              setMood(Mood.PANIC);
69              setState(State.REPORT);
70              return "MOON";
71          }
72          else
73          {
74              setMood(Mood.CALM);
75              setState(State.TALK);
76              System.out.println(getName()+": Этообычныйметеорит");
```

```
77              setState(State.OBSERVE);
78              setMood(Mood.CONCENTRATED);
79              return "METEORITE";
80          }
81       }
82       else
83       {
84          setState(State.TALK);
85          System.out.println(getName()+": Мынеможемисследоватьлунноенебо,
                ↪ унаснедостаточнооборудованиядляэтого!");
86          setState(State.SIT);
87          return "Nothing";
88       }
89
90    }
91
92    public void awarness(Spaceship4 spc){
93       if (suspicions && spc.rotationInSpace && spc.getAccelerationInSpace()<0){
94          setState(State.TALK);
95          System.out.println(getName()+": Всёсходится! Этоточноракета!
                ↪ Никакихсомнений, этоона!");
96          setState(State.OBSERVE);
97       }
98    }
99    public void saidAboutNumberOfPeople(Spaceship4 spc){
100      System.out.println(getName()+" : Number of people on the board is "+ spc.
             ↪ getNumberOfPeople());
101   }
102}
```

Human.java

```
1  package Live;
2
3  import Objects.*;
4  import Others.State;
5
6  import java.util.concurrent.TimeUnit;
7
8  public abstract class Human extends LiveCreature {
9     private final String nativeLanguage;
10    private final String Name;
11    private State state;
12    public Human(String name, Mood mood, State state, String language){
13       super(name, mood, state);
14       this.Name = name;
15       this.nativeLanguage = language;
16    }
17    @Override
18    public void listen(){
19       setState(State.LISTEN);
20       try {
21          TimeUnit.SECONDS.sleep(1);
22       } catch (InterruptedException ie) {
23          Thread.currentThread().interrupt();
24       }
25    }
```

```
26      @Override
27      public void eat(String nameOfMeal){
28          setState(State.EAT);
29          System.out.println(getName() + ": Сейчасяем"+nameOfMeal);
30
31      }
32      public void speak(){
33          setState(State.TALK);
34          System.out.println(this.Name+" speaking "+this.nativeLanguage);
35          setState(State.STAND);
36      }
37      public void setStateToSomething(Objects object, State newState){
38          object.setState(newState);
39      }
40      public abstract void gotIntelligence();
41
42
43
44  }
```

LiveCreature.java

```
1   package Live;
2
3   import Others.State;
4   import Interfaces.Alive;
5   import Interfaces.StateNameInterface;
6
7   import java.util.concurrent.TimeUnit;
8
9
10  public abstract class LiveCreature implements Alive, StateNameInterface{
11      private final String Name;
12      private Mood Mood;
13      private State State;
14      public LiveCreature(String name, Mood mood, State state){
15          this.Name = name;
16          this.Mood = mood;
17          this.State = state;
18      }
19
20      public final State getState(){
21          return this.State;
22      }
23      public final Mood getMood(){
24          return this.Mood;
25      }
26      public final String getName(){
27          return this.Name;
28      }
29
30      public abstract void eat(String nameOfMeal);
31      public abstract void listen();
32      public void setState(State newState){
33          if (this.getState() == newState){
34              System.out.println(this.getName()+" is already "+ this.getState().toString
                    ↪ ());
```

```
35        }
36        else{
37            this.State = newState;
38            System.out.println(this.getName()+" is "+ this.getState().toString()+"
                  ↪ right now");
39        }
40    }
41    public final void setMood(Mood newMood){
42        if (this.getMood() == newMood){
43            System.out.println(this.getName()+" is already "+ this.getMood().toString
                  ↪ ());
44        }
45        else{
46            this.Mood = newMood;
47            System.out.println(this.getName()+" is "+ this.getMood().toString()+"
                  ↪ right now");
48        }
49    }
50
51    public void dead(){
52        this.State = State.DEAD;
53    }
54 }
```

Mood.java

```
1  package Live;
2
3  public enum Mood {
4     HAPPY("happy"),
5     SAD("sad"),
6     ANGRY("angry"),
7     PANIC("panic"),
8     CALM("calm"),
9     CONCENTRATED("concentrated");
10
11    private final String Status;
12    Mood(String status) {
13       Status = status;
14    }
15    @Override
16    public final String toString() {
17       return this.Status;
18    }
19 }
```

PeopleInSpaceship.java

```
1  package Live;
2
3  import Objects.Spaceship4;
4  import Others.State;
5
6  public class PeopleInSpaceship extends Human{
7     private Spaceship4 Spaceship;
8     public boolean AgreeOnLandingPlace = false;
9     public PeopleInSpaceship(String names, Mood mood, State state, String language,
          ↪ Spaceship4 spaceship){
```

```
10        super(names, mood, state, language);
11        this.Spaceship = spaceship;
12    }
13    @Override
14    public void gotIntelligence() {
15        System.out.println(getName()+"имееет интеллект");
16    }
17
18    public void AgreeOnLandingPlace(){
19        System.out.println(getName() + ": Мынепосадим" + this.Spaceship.getName()+"
              ↪ Поканенайдёмкорабль НезнайкииПончика!");
20        AgreeOnLandingPlace = true;
21    }
22 }
```

Policeman.java

```
1  package Live;
2
3  import Others.State;
4  import java.util.concurrent.TimeUnit;
5
6  public class Policeman extends Human{
7     public Policeman(String name, Mood mood, State state, String nativeLanguage){
8        super(name, mood, state, nativeLanguage);
9     }
10    @Override
11    public void gotIntelligence() {
12       System.out.println(getName()+"имееет интеллект");
13    }
14    public void arrest(Human human){
15       human.setState(State.AREST);
16       System.out.println(getName() + ": Япоймал"+human.getName()+ "!");
17    }
18    public void patrolling(){
19       setState(State.WALK);
20       try {
21          TimeUnit.SECONDS.sleep(1);
22       } catch (InterruptedException ie) {
23          Thread.currentThread().interrupt();
24       }
25       setState(State.STAND);
26    }
27
28 }
```

Rjigl.java

```
1  package Live;
2  import Others.State;
3
4
5
6  public class Rjigl extends Policeman{
7     public Rjigl(String name, Mood mood, State state, String nativeLanguage) {
8        super(name, mood, state, nativeLanguage);
9     }
```

```
10      @Override
11      public void gotIntelligence() {
12          System.out.println(getName()+"имееет интеллект");
13      }
14      public void getCall(String themeOfCall){
15          if (themeOfCall == "SPRUTS"){
16              listen();
17              setState(State.TALK);
18              System.out.println(getName() + ": Дайтевсюинформациюобэтомобъекте");
19          }
20          else{
21              System.out.println(getName() + ": Ничегонепроизошло!");
22          }
23      }
24  }
```

Spruts.java

```
1   package Live;
2
3   import Others.State;
4
5   public class Spruts extends Human {
6       public Spruts(String name, Mood mood, State state, String nativeLanguage){
7           super(name, mood, state, nativeLanguage);
8       }
9       @Override
10      public void gotIntelligence() {
11          System.out.println(getName()+"не имееетинтеллект");
12      }
13      public String getCall(String themeOfCall){
14          if (themeOfCall == "MOON"){
15              listen();
16              setMood(Mood.ANGRY);
17              setState(State.TALK);
18              System.out.println(getName() +": Онихотятнамтогосамогосделать,
                    ↪ АЛООПОЛИСМЕНСДЕЛАЙЧТОНИБУДЬ-!");
19              return "SPRUTS";
20
21          }
22          else{
23              System.out.println(getName()+ ": Ничегонепроизошла, аденьг,
                    ↪ какивластисталотолькобольше!");
24              return "Не надозвноитькомиссару, наэтонетпричин";
25          }
26      }
27
28  }
```

Direction.java

```
1   package Objects;
2
3   public enum Direction {
4       TOWARDSPLANET("towards the planet"),
5       FROMPLANET("from the planet");
6
```

```
7     private final String name;
8     Direction(String name) {
9        this.name = name;
10    }
11    @Override
12    public String toString() {
13       return this.name;
14    }
15 }
```

GravityLocator.java

```
1  package Objects;
2
3  import Live.Astronomers4;
4  import exceptions.inappropriateVelocity;
5  import Others.State;
6
7
8  public class GravityLocator extends Objects{
9     public GravityLocator (String name, Materials material, State state,
          ↪ PlanetaryGravity planetaryGravity){
10       super(name, material, state, planetaryGravity);
11    }
12
13    @Override
14    public void beAstroneersProperty(){
15       System.out.println(getName()+" являетсясобственностьюлунныхастрономов");
16    }
17    public void checkVelocity(int[] a) throws inappropriateVelocity{
18       if ((a[1] - a[0])/(a[2]*60) < 0 || (a[1] - a[0])/(a[2]*60)> 2400){
19          throw new inappropriateVelocity();
20       }
21    }
22    public int getVelocity(Meteorite mt){
23       int[] a = mt.getCoordinates();
24       checkVelocity(a);
25       return (a[1] - a[0])/(a[2]*60);
26    }
27    public int getVelocity(Spaceship4 spc) {
28       int[] a = spc.getCoordinates();
29       checkVelocity(a);
30       return (a[1] - a[0])/(a[2]*60);
31    }
32    public void getVolume(Spaceship4 spc, Astronomers4 ast){
33       System.out.println(ast.getName()+": Volume of "+spc.getName()+ " is "+spc.
            ↪ getSize().toString());
34    }
35    public void getWeight(Spaceship4 spc, Astronomers4 ast){
36
37       System.out.println(ast.getName()+": Weight of "+spc.getName()+" is "+Integer.
            ↪ toString(spc.getNumberOfPeople()*5+200));
38    }
39
40    @Override
41    public String toString(){
42       return "Этот "+getName()+" сделаниз"+getMaterial().toString()+".
```

```
        ↪ Оннужендлявычисленияскоростьобъекта, весаиобъёма";
43  }
44
45  @Override
46  public int hashCode(){
47      return this.getMaterial().hashCode() + this.getName().hashCode() + this.
        ↪ getState().hashCode();
48  }
49  @Override
50  public boolean equals(Object obj) {
51      if (this.hashCode() == obj.hashCode()) return true;
52      if (this.hashCode() ≠ obj.hashCode()) return false;
53      return this.toString() == obj.toString();
54  }
55 }
```

Materials.java

```
1  package Objects;
2
3  public enum Materials {
4      METAL("metal"),
5      PLASTIC("plastic"),
6      SPACESTONE("spacestone"),
7      GLASS("glass");
8
9
10     private final String name;
11     Materials(String name) {
12         this.name = name;
13     }
14     @Override
15     public String toString() {
16         return this.name;
17     }
18 }
```

Meteorite.java

```
1  package Objects;
2
3  import Interfaces.ThingInSpace;
4  import Others.State;
5
6  public class Meteorite extends Objects implements ThingInSpace {
7      private SizeOfGravityWaves Size;
8      public int x_now;
9      public int x_SomeminAgo;
10     public int Minute;
11     public Direction Direction;
12     private int AccelerationInSpace;
13     public boolean rotationInSpace = false;
14     public Meteorite(String name, Materials material, State state,
           ↪ SizeOfGravityWaves size, PlanetaryGravity planetaryGravity, Direction
           ↪ direction, int minute, int x_someminAgo, int x_now, int
           ↪ accelerationInSpace){
15         super(name, material, state, planetaryGravity);
```

```
16          this.Size = size;
17          this.x_now = x_now;
18          this.x_SomeminAgo = x_someminAgo;
19          this.Minute = minute;
20          this.Direction = direction;
21          this.AccelerationInSpace = accelerationInSpace;
22      }
23      @Override
24      public void beAstroneersProperty(){
25          System.out.println(getName()+" неявляетсясобственностьюлунныхастрономов");
26      }
27      public int[] getCoordinates(){
28          int[] coordinates = {this.x_SomeminAgo, this.x_now, this.Minute};
29          return coordinates;
30      }
31      public int getAccelerationInSpace(){
32          return this.AccelerationInSpace;
33      }
34      public Direction getDirection(){
35          return this.Direction;
36      }
37      public final SizeOfGravityWaves getSize(){
38          return this.Size;
39      }
40      public void setDirection(Direction newDirection){
41          if (this.getDirection() == newDirection){
42              System.out.println(this.getName()+" is already flying"+ this.getDirection
                    ↪ ().toString());
43          }
44          else{
45              this.Direction = newDirection;
46              System.out.println(this.getName()+" is flying"+ this.getDirection().
                    ↪ toString()+" right now");
47              this.rotationInSpace=true;
48          }
49      }
50      public void setAccelerationInSpace(int newAcceleration){
51          if (this.getAccelerationInSpace() == newAcceleration){
52              System.out.println(this.getName()+" is already have"+ this.
                    ↪ getAccelerationInSpace() + " acceleration");
53          }
54          else{
55              this.AccelerationInSpace = newAcceleration;
56              System.out.println(this.getName()+" is having"+ this.
                    ↪ getAccelerationInSpace()+" acceleration right now");
57          }
58      }
59
60      @Override
61      public String toString(){
62          return "Это "+getName()+", сделаниз"+getMaterial().toString()+".\Онn
                ↪ летитускорением"+ getAccelerationInSpace()+
63                  " в"+ getDirection().toString()+" направлении.
                        ↪ Онespecкaетволнытяготенияразмера" + getSize().toString();
64      }
65
```

```
66     @Override
67     public int hashCode(){
68        return this.getMaterial().hashCode() + this.getName().hashCode() + this.
          ↪ getState().hashCode() +
69           this.getSize().hashCode() + this.getDirection().hashCode();
70     }
71     @Override
72     public boolean equals(Object obj) {
73        if (this.hashCode() == obj.hashCode()) return true;
74        if (this.hashCode() ≠ obj.hashCode()) return false;
75        return this.toString() == obj.toString();
76     }
77  }
```

Objects.java

```
1  package Objects;
2
3  import Others.State;
4  import Interfaces.NotAlive;
5  import Interfaces.StateNameInterface;
6
7  public abstract class Objects implements NotAlive, StateNameInterface {
8     private final String Name;
9     private final Materials Material;
10    private State State;
11    private PlanetaryGravity PlanetaryGravity;
12    public Objects(String name, Materials material, State state, PlanetaryGravity
         ↪ planetaryGravity){
13       this.Name = name;
14       this.Material = material;
15       this.State = state;
16       this.PlanetaryGravity = planetaryGravity;
17    }
18
19    public abstract void beAstroneersProperty();
20    public final State getState(){
21       return this.State;
22    }
23    public final String getName(){
24       return this.Name;
25    }
26    public final Materials getMaterial() {
27       return this.Material;
28    }
29    public PlanetaryGravity getPlanetaryGravity(){
30       return this.PlanetaryGravity;
31    }
32    public void setState(State newState){
33       if (this.getState() == newState){
34          System.out.println(this.getName()+" is already "+ this.getState().toString
              ↪ ());
35       }
36       else{
37          this.State = newState;
38          System.out.println(this.getName()+" is "+ this.getState().toString()+"
              ↪ right now");
```

```
39        }
40     }
41     public void setPlanetaryGravity(PlanetaryGravity newPlanetaryGravity){
42        if (this.getPlanetaryGravity() == newPlanetaryGravity){
43           System.out.println(this.getName()+" is already "+ this.getPlanetaryGravity
                 ↪ ().toString()+ " gravity");
44        }
45        else{
46           this.PlanetaryGravity = newPlanetaryGravity;
47           System.out.println(this.getName()+" is in "+ this.getPlanetaryGravity().
                 ↪ toString()+" gravity right now");
48        }
49     }
50  }
```

PartsOfMoon.java

```
1  package Objects;
2
3  public enum PartsOfMoon {
4     SHORE("the shore of the fossilized sea"),
5     CRATOR("crator"),
6     PEEK("peek of the mountain"),
7     NOTINMOON("not in the moon");
8
9     private final String name;
10    PartsOfMoon(String name) {
11       this.name = name;
12    }
13    @Override
14    public String toString() {
15       return this.name;
16    }
17 }
```

PlanetaryGravity.java

```
1  package Objects;
2
3  public enum PlanetaryGravity {
4     NONE("none"),
5     EARTH("earth"),
6     MOON("moon");
7
8     private final String name;
9     PlanetaryGravity(String name) {
10       this.name = name;
11    }
12    @Override
13    public String toString() {
14       return this.name;
15    }
16 }
```

Screen.java

```
1  package Objects;
```

```

import Others.State;

public class Screen extends Objects{
    public Screen (String name, Materials material, State state, PlanetaryGravity
        ↪ planetaryGravity){
        super(name, material, state, planetaryGravity);
    }
    @Override
    public void beAstroneersProperty(){
        System.out.println(getName()+" являетсясобственностьюлунныхастрономов");
    }
    public SizeOfGravityWaves wavesIntoLine(Telescope4 tls, Spaceship4 spc){
        return tls.getWaves(spc);
    }
    public SizeOfGravityWaves wavesIntoLine(Telescope4 tls, Meteorite mt){
        return tls.getWaves(mt);
    }
    @Override
    public String toString(){
        return "Этот "+getName()+" сделаниз"+getMaterial().toString()+".
            ↪ Оннужендлявыведениякривойлинии";
    }
    @Override
    public int hashCode(){
        return this.getMaterial().hashCode() + this.getName().hashCode() + this.
            ↪ getState().hashCode();
    }
    @Override
    public boolean equals(Object obj) {
        if (this.hashCode() == obj.hashCode()) return true;
        if (this.hashCode() ≠ obj.hashCode()) return false;
        return this.toString() == obj.toString();
    }

}
```

SizeOfGravityWaves.java

```
package Objects;

public enum SizeOfGravityWaves {
    BIG("big"),
    MEDIUM("medium"),
    SMALL("small"),
    NOTENOUGH("not enough");

    private final String name;
    SizeOfGravityWaves(String name) {
        this.name = name;
    }
    @Override
    public String toString() {
        return this.name;
    }
}
```

Spaceship.java

```java
package Objects;

import Others.State;

public class Spaceship extends Objects{
    private final int x_now;
    private final int x_SomeminAgo;
    private final int Minute;
    public Spaceship(String name, Materials material, State state, PlanetaryGravity
        ↪ planetaryGravity, int minute, int x_someminAgo, int x_now){
        super(name, material, state, planetaryGravity);
        this.x_now = x_now;
        this.x_SomeminAgo = x_someminAgo;
        this.Minute = minute;
    }
    @Override
    public void beAstroneersProperty(){
        System.out.println(getName()+" неявляетсясобственностьюлунныхастрономов");
    }
    public int[] getCoordinates(){
        int[] coordinates = {this.x_SomeminAgo, this.x_now, this.Minute};
        return coordinates;
    }

    @Override
    public String toString(){
        return "Это "+getName()+", сделаниз"+getMaterial().toString();
    }
    @Override
    public int hashCode() {
        return this.getCoordinates().hashCode() + getMaterial().hashCode();
    }

    @Override
    public boolean equals(Object obj) {
        if (this.hashCode() == obj.hashCode()) return true;
        if (this.hashCode() ≠ obj.hashCode()) return false;
        return this.toString() == obj.toString();
    }
}
```

Spaceship4.java

```java
package Objects;

import Interfaces.ThingInSpace;
import Others.State;
import exceptions.NegativeNumberOfPeopleException;
public class Spaceship4 extends Spaceship implements ThingInSpace {
    private SizeOfGravityWaves Size;
    private Direction Direction;
    private int NumberOfPeople;
    private int AccelerationInSpace;
    private PartsOfMoon PartOfMoon;

    public boolean rotationInSpace = false;
```

```
14
15      public Spaceship4(String name, Materials material, State state, int
           ↪ numberOfPeople,SizeOfGravityWaves size, PlanetaryGravity planetaryGravity,
           ↪  PartsOfMoon partOfMoon, Direction direction, int minute, int x_someminAgo
           ↪ , int x_now, int acceleration) throws NegativeNumberOfPeopleException{
16         super(name, material, state, planetaryGravity,minute, x_someminAgo, x_now);
17         this.Size = size;
18         this.Direction = direction;
19         this.PartOfMoon=partOfMoon;
20         if (numberOfPeople <0){
21            throw new NegativeNumberOfPeopleException("Корабль неполетитбезлюдей!");
22         }
23         this.NumberOfPeople = numberOfPeople;
24         this.AccelerationInSpace = acceleration;
25      }
26
27      public int getNumberOfPeople(){
28         return this.NumberOfPeople;
29      }
30      public int getAccelerationInSpace(){
31         return this.AccelerationInSpace;
32      }
33      public PartsOfMoon getPartOfMoon(){
34         return this.PartOfMoon;
35      }
36      public void setPartOfMoon(PartsOfMoon newPartOfMoon){
37         if (this.getPartOfMoon() == newPartOfMoon){
38            System.out.println(this.getName()+" is already at "+ this.getPartOfMoon().
                 ↪ toString());
39         }
40         else{
41            this.PartOfMoon = newPartOfMoon;
42            System.out.println(this.getName()+" is at "+ this.getPartOfMoon().toString
                 ↪ ()+" right now");
43         }
44      }
45      public Direction getDirection(){
46         return this.Direction;
47      }
48      public void setDirection(Direction newDirection){
49         if (this.getDirection() == newDirection){
50            System.out.println(this.getName()+" is already flying "+ this.getDirection
                 ↪ ().toString());
51         }
52         else{
53            this.Direction = newDirection;
54            System.out.println(this.getName()+" is flying "+ this.getDirection().
                 ↪ toString()+" right now");
55            this.rotationInSpace=true;
56         }
57      }
58      public void setAccelerationInSpace(int newAcceleration){
59         if (this.getAccelerationInSpace() == newAcceleration){
60            System.out.println(this.getName()+" is already have "+ this.
                 ↪ getAccelerationInSpace() + " acceleration");
61         }
```

```
62          else{
63              this.AccelerationInSpace = newAcceleration;
64              System.out.println(this.getName()+" is having "+ this.
                    ↪ getAccelerationInSpace()+" acceleration right now");
65          }
66      }
67      public void seekingWhileLooping(int numberOfLoops, Spaceship4 spc){
68          System.out.println("Пока "+getName()+" искали"+ spc.getName()+" онисделали"+
                ↪ numberOfLoops+" витковвокругЛуны!");
69          setPartOfMoon(spc.getPartOfMoon());
70          System.out.println(getName()+": Мынашли"+spc.getName());
71
72      }
73      public final SizeOfGravityWaves getSize(){
74          return this.Size;
75      }
76
77      @Override
78      public String toString(){
79          return "Это "+getName()+", сделаниз"+getMaterial().toString()+".\Онn
                ↪ летитускорением"+ getAccelerationInSpace()+
80                  " в"+ getDirection().toString()+" направлении.
                        ↪ Ониспускаетволнытяготенияразмера" + getSize().toString();
81      }
82
83      @Override
84      public int hashCode(){
85          return this.getMaterial().hashCode() + this.getName().hashCode() + this.
                ↪ getState().hashCode() +
86              this.getSize().hashCode() + this.getDirection().hashCode();
87      }
88      @Override
89      public boolean equals(Object obj) {
90          if (this.hashCode() == obj.hashCode()) return true;
91          if (this.hashCode() ≠ obj.hashCode()) return false;
92          return this.toString() == obj.toString();
93      }
94  }
```

Telescope.java

```
1   package Objects;
2
3   import Others.State;
4
5   public class Telescope extends Objects{
6       private final int multiplicity;
7       public Telescope(String name, Materials material, State state, PlanetaryGravity
            ↪ planetaryGravity, int Multiplicity){
8           super(name, material, state, planetaryGravity);
9           this.multiplicity = Multiplicity;
10      }
11      @Override
12      public void beAstroneersProperty(){
13          System.out.println(getName()+" являетсясобственностьюлунныхастрономов");
14      }
15      public final int getMultiplicity(){
```

```
16          return this.multiplicity;
17      }
18
19      @Override
20      public String toString(){
21          return "Этот "+getName()+" сделаниз"+getMaterial().toString()+".
                ↪ Оннужендлянаблюдениязаобъектамивкосмосе.\n"+
22                  "Он имееткратность" + getMultiplicity() + ".";
23      }
24
25      @Override
26      public int hashCode(){
27          return this.getMaterial().hashCode() + this.getName().hashCode() + this.
                ↪ getState().hashCode();
28      }
29      @Override
30      public boolean equals(Object obj) {
31          if (this.hashCode() == obj.hashCode()) return true;
32          if (this.hashCode() ≠ obj.hashCode()) return false;
33          return (this.toString() == obj.toString());
34      }
35  }
```

Telescope4.java

```
1   package Objects;
2
3   import Others.State;
4
5   public class Telescope4 extends Telescope{
6       private Angles Angles;
7       public Telescope4(String name, Materials material, State state, PlanetaryGravity
            ↪  planetaryGravity, int multiplicity, Angles angles){
8           super(name, material, state, planetaryGravity, multiplicity);
9           this.Angles = angles;
10      }
11
12      public enum Angles{
13          CENTER("center"),
14          LEFT("left"),
15          RIGHT("right"),
16          UP("up"),
17          DOWN("down");
18          private final String name;
19          Angles(String name) {
20              this.name = name;
21          }
22          @Override
23          public String toString() {
24              return this.name;
25          }
26      }
27      public Angles getAngles(){
28          return this.Angles;
29      }
30      public SizeOfGravityWaves getWaves(Meteorite mt){
31          return mt.getSize();
```

```
32      }
33      public SizeOfGravityWaves getWaves(Spaceship4 spc){
34          return spc.getSize();
35      }
36
37      @Override
38      public String toString(){
39          return "Этот "+getName()+" сделаниз"+getMaterial().toString()+".
            ↪ Оннужендлянаблюдениязаобъектамивкосмосе.\n"+
40              "Он имееткратность" + getMultiplicity() + "и направленв" + getAngles().
                ↪ toString() + " сторону";
41      }
42
43      @Override
44      public int hashCode(){
45          return this.getMaterial().hashCode() + this.getName().hashCode() + this.
            ↪ getState().hashCode() +
46              this.getAngles().hashCode();
47      }
48      @Override
49      public boolean equals(Object obj) {
50          if (this == obj) return true;
51          if (obj == null) return false;
52          return obj instanceof Telescope4 && this.hashCode() == obj.hashCode();
53      }
54  }
```

State.java

```
1   package Others;
2
3   public enum State {
4       STAND("standing"),
5       DEAD("dead"),
6       LISTEN("listening"),
7       REPORT("reporting"),
8       FLY("flying"),
9       SEEK("seeking"),
10      TALK("talking"),
11      WALK("walking"),
12      OBSERVE("observing"),
13      SIT("sitting"),
14      AREST("arrested"),
15      USE("in use"),
16      EAT("eating");
17      private final String Status;
18      State(String status) {
19          Status = status;
20      }
21      @Override
22      public final String toString() {
23          return this.Status;
24      }
25  }
```

Main1.java

```
import Live.*;
import Objects.*;
import Others.*;

public class Main1 {
   public static void main(String[] args) throws Throwable{
      Telescope4 telescope = new Telescope4("Телескоп", Materials.PLASTIC, State.STAND, PlanetaryGravity.MOON, 10, Telescope4.Angles.CENTER);
      Screen screen = new Screen("Экран", Materials.GLASS, State.STAND, PlanetaryGravity.MOON);
      GravityLocator gravityLocator = new GravityLocator("Гравитационный локатор", Materials.METAL, State.STAND, PlanetaryGravity.MOON);
      Astronomers4 astronomers = new Astronomers4("Астрономы", Mood.CONCENTRATED, State.OBSERVE, "moon language");
      Meteorite meteorite1 = new Meteorite("Метеор 1", Materials.SPACESTONE, State.FLY, SizeOfGravityWaves.SMALL, PlanetaryGravity.NONE, Direction.FROMPLANET, 10, 1, 500000, 0);
      Meteorite meteorite2 = new Meteorite("Метеор 2", Materials.SPACESTONE, State.FLY, SizeOfGravityWaves.MEDIUM, PlanetaryGravity.NONE, Direction.FROMPLANET, 10, 1, 200000, -1);
      Spaceship4 spaceship = new Spaceship4("Корабль Знайки", Materials.METAL, State.FLY, 25, SizeOfGravityWaves.BIG, PlanetaryGravity.EARTH, PartsOfMoon.NOTINMOON, Direction.FROMPLANET, 10, 1, 1000000, 0);
      Spaceship4 spaceship1 = new Spaceship4("Корабль НезнайкииПончика", Materials.METAL, State.STAND, 2, SizeOfGravityWaves.BIG, PlanetaryGravity.MOON, PartsOfMoon.SHORE, Direction.TOWARDSPLANET, 10, 1, 1, 0);
      Spruts spruts = new Spruts("Спрутс", Mood.HAPPY, State.SIT, "moonLanguage");
      Rjigl rjigl = new Rjigl("Комиссар Ржигль", Mood.CALM, State.WALK, "moonLanguage");
      PeopleInSpaceship peopleInSpaceship = new PeopleInSpaceship("Знайка, ФуксияиСеледочка", Mood.HAPPY, State.WALK, "earth language", spaceship) ;

      peopleInSpaceship.AgreeOnLandingPlace();
      astronomers.setStateToSomething(telescope, State.USE);
      astronomers.setStateToSomething(screen, State.USE);
      astronomers.setStateToSomething(gravityLocator, State.USE);
      astronomers.exploreObject(meteorite1, telescope, screen, gravityLocator);
      astronomers.exploreObject(meteorite2, telescope, screen, gravityLocator);
      if (telescope.getAngles()== Telescope4.Angles.CENTER){
         // СТАРАЯИСТОРИЯ
         String checkObjectInSpace = astronomers.exploreObject(spaceship, telescope , screen, gravityLocator);
         String checkCallToSpruts = spruts.getCall(checkObjectInSpace);
         rjigl.getCall(checkCallToSpruts);
// СТАРАЯИСТОРИЯЗАКОНЧИЛАСЬ
         spaceship.setPlanetaryGravity(PlanetaryGravity.NONE);


         spaceship.setAccelerationInSpace(-1);
         spaceship.setDirection(Direction.TOWARDSPLANET);
         if (spaceship.rotationInSpace && spaceship.getAccelerationInSpace()<0){
            astronomers.awarness(spaceship);
            spaceship.setPlanetaryGravity(PlanetaryGravity.MOON);
            spaceship.setAccelerationInSpace(0);
            gravityLocator.getVolume(spaceship, astronomers);
```

```
41              gravityLocator.getWeight(spaceship, astronomers);
42              astronomers.saidAboutNumberOfPeople(spaceship);
43              if (peopleInSpaceship.AgreeOnLandingPlace){
44                  spaceship.seekingWhileLooping(24, spaceship1);
45                  System.out.println("Конец!");
46              }
47              else {
48                  System.out.println(spaceship.getName()+": Акудамысядем?
                        ↪ Мыэтогонеобговаривали!");
49              }
50          }
51          else{
52              System.out.println(spaceship.getName()+" пролетелмимопланеты\Конецn!!")
                    ↪ ;
53          }
54      }
55      else{
56          System.out.println(astronomers.getName()+"
                ↪ смотреливдругуюсторонуинеувиделикорабля");
57          spaceship.setPlanetaryGravity(PlanetaryGravity.NONE);
58
59          spaceship.setAccelerationInSpace(-1);
60          spaceship.setDirection(Direction.TOWARDSPLANET);
61          spaceship.setPlanetaryGravity(PlanetaryGravity.MOON);
62          spaceship.setAccelerationInSpace(0);
63          if (peopleInSpaceship.AgreeOnLandingPlace){
64              spaceship.seekingWhileLooping(24, spaceship1);
65              System.out.println("Конец!");
66          }
67          else {
68              System.out.println(spaceship.getName()+": Акудамысядем?
                    ↪ Мыэтогонеобговаривали!");
69          }
70      }
71
72  }
73
74 }
```

4 Результат работы программы

Знайка, Фуксия и Селедочка: Мы не посадим Корабль Знайки Пока не найдём корабль Незнайки и Пончика!
Телескоп is in use right now
Экран is in use right now
Гравитационный локатор is in use right now
Астрономы is calm right now
Астрономы is talking right now
Астрономы: Это обычный метеорит
Астрономы is observing right now
Астрономы is concentrated right now
Астрономы is calm right now
Астрономы is talking right now
Астрономы: Это обычный метеорит
Астрономы is observing right now
Астрономы is concentrated right now

Астрономы is talking right now
Астрономы: Это противоречит законам небесной механики, согласно которым скорость тела, движущегося вблизи планеты, могла увеличиваться только в том случае, если бы тело приближалось к планете
Астрономы: В этой штуке полюбому есть двигатель!
Астрономы is panic right now
Астрономы is reporting right now
Спрутс is listening right now
Спрутс is angry right now
Спрутс is talking right now
Спрутс: Они хотят нам того самого сделать, АЛОО ПОЛИСМЕН СДЕЛАЙ ЧТО-НИБУДЬ!
Комиссар Ржигль is listening right now
Комиссар Ржигль is talking right now
Комиссар Ржигль: Дайте всю информацию об этом объекте
Корабль Знайки is in none gravity right now
Корабль Знайки is having -1 acceleration right now
Корабль Знайки is flying towards the planet right now
Астрономы is talking right now
Астрономы: Всё сходится! Это точно ракета! Никаких сомнений, это она!
Астрономы is observing right now
Корабль Знайки is in moon gravity right now
Корабль Знайки is having 0 acceleration right now
Астрономы: Volume of Корабль Знайки is big
Астрономы: Weight of Корабль Знайки is 325
Астрономы : Number of people on the board is 25
Пока Корабль Знайки искали Корабль Незнайки и Пончика они сделали 24 витков вокруг Луны!
Корабль Знайки is at the shore of the fossilized sea right now
Корабль Знайки: Мы нашли Корабль Незнайки и Пончика
Конец!

5 Вывод

Я познакомился с тремя типами исключений и обработкой этих исключений. также ознакомился с вложенными, локальными и анонимными классами.