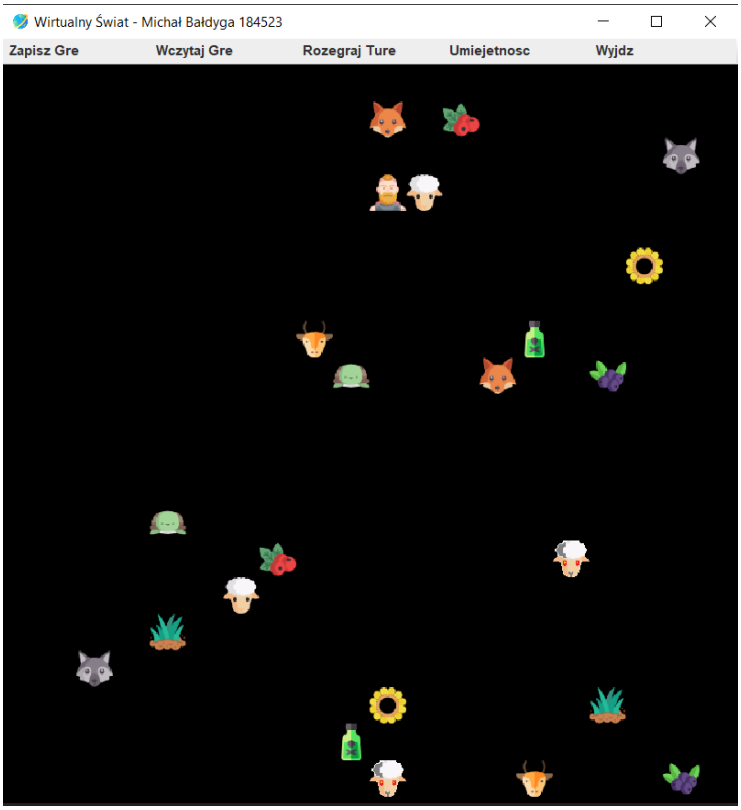
Raport z Projektu 2 – Java

Michał Bałdyga 184523

1. Implementacja świata gry i jego wizualizacji.

```java
public class Swiat {

    // Atrybuty

    private static final int szerokosc = 20;
    private static final int wysokosc = 20;
    private static final int poczatkowaIloscGatunku = 2;
    private final Organizm organizmy[][] = new Organizm[szerokosc][wysokosc];
    private final List<Organizm>  kolejka = new LinkedList<~>();
    private final List<Organizm> martweOrganizmy = new LinkedList<~>();
    private final Komentator komentator = new Komentator();

    // Metody prywatne

    private void sortujKolejke(List<Organizm>  kol) {...}

    private Vector<Polozenie> znajdzPolaGenerowania() {...}

    private void usunOrganizmy(JFrame frame) {...}

    private void wczytajOrganizm(String dane) {...}

    private void stworzOrganizm(String gatunek, int sila, int x, int y) {...}

    private void stworzCzlowieka(int sila, int x, int y, int czasUmiejetnosci, int czasLadowania) {...}

    private void wyczyscSwiat(JFrame frame) {...}

    // Metody publiczne

    public void inicjalizujSwiat() {...}

    public void rysujSwiat(JFrame frame) {...}
```

```java
public void wykonajTure(JFrame frame) {...}

public void generujOrganizmy() {...}

public void dodajOrganizm(Organizm org) {...}

public void dodajDoUsuniecia(Organizm org) {...}

public Czlowiek znajdzCzlowieka() {...}

public void aktywujUmiejetnosc() {...}

public void zapiszStan() {...}

public void wczytajStan(JFrame frame) {...}

public void zakonczGre() {...}

// Getery
public Organizm[][] getOrganizmy() {...}
public Organizm getOrganizm(Polozenie polozenie) {...}
public List<Organizm> getKolejka() {...}
public int getSzerokosc() {...}
public int getWysokosc() {...}
public Komentator getKomentator() {...}

// Setery
public void setOrganizm(Polozenie p, Organizm org) {...}
```



2. Implementacja wszystkich obowiązkowych gatunków zwierząt (wraz z cyber-owcą):

- Abstrakcyjna klasa Organizm:

```java
public abstract class Organizm implements FormaZycia {

    // Wymiary obszaru wyświetlania (label)
    private static final int wysokosc = 32;
    private static final int szerokosc = 32;

    // Atrybuty
    protected int sila;
    protected ImageIcon ikona;
    protected JLabel obszarWyswietlania; // Obszar wyświetlania organizmu na planszy
    protected int inicjatywa;
    protected Polozenie polozenie;
    protected Swiat swiat;
    protected static final int zasiegRuchu = 1;

    // Metody chronione
    protected boolean czyPoleWolne(Polozenie p) { return (swiat.getOrganizm(p) == null); }
    protected Vector<Polozenie> znajdzSasiedniePola(int zasieg) {...}
    protected Vector<Polozenie> znajdzWolnePola(Organizm org) {...}

    // Metody publiczne
    public Organizm(int sila, ImageIcon ikona, int inicjatywa, Polozenie polozenie, Swiat swiat) {...}
    public void rysujOrganizm(JFrame frame) {...}
    public String przygotujDoZapisu() { return (getNazwa() + ";" + getSila() + ";" + getX() + ";" + getY() + "\n"); }
```

```java
// Getery
public int getSila() { return sila; }
public ImageIcon getIkona() { return ikona; }
public int getInicjatywa() { return inicjatywa; }
public Polozenie getPolozenie() { return polozenie; }
public int getX() { return polozenie.x; }
public int getY() { return polozenie.y; }
public JLabel getObszarWyswietlania() { return obszarWyswietlania; }
public int getZasiegRuchu() { return zasiegRuchu; }

// Setery
public void setPolozenie(Polozenie polozenie) { this.polozenie = polozenie; }
public void setSila(int sila) { this.sila = sila; }
public void setInicjatywa(int inicjatywa) { this.inicjatywa = inicjatywa; }
public void setObszarWyswietlania(int x, int y) { obszarWyswietlania.setLocation(x, y);
```

```java
public interface FormaZycia {

    void akcja();
    void kolizja(Organizm atakujacy);
    Organizm stworzPotomka(Polozenie polozenie, Swiat swiat);
    String getNazwa();
}
```

- Abstrakcyjna klasa Zwierzę:

```
public abstract class Zwierze extends Organizm {

    // Metody publiczne

    public Zwierze(int sila, ImageIcon ikona, int inicjatywa, Polozenie polozenie, Swiat swiat) {...}

    @Override
    public void akcja() {...}

    @Override
    public void kolizja(Organizm atakujacy) {...}

    public void zmienPolozenie(Polozenie nowePolozenie) {...}

    // Metody chronione

    protected void sprobujWykonacRuch(Polozenie nowePolozenie) {...}

    protected void rozmnoz(Organizm partner) {...}

    protected void walcz(Organizm atakujacy) {...}

    protected boolean czyOdbilAtak(Organizm atakujacy) { return false; }

    protected boolean czyZrobilUnik() { return false; }

    // Metody prywatne

    private void zrobUnik(Organizm atakujacy) {...}
}
```

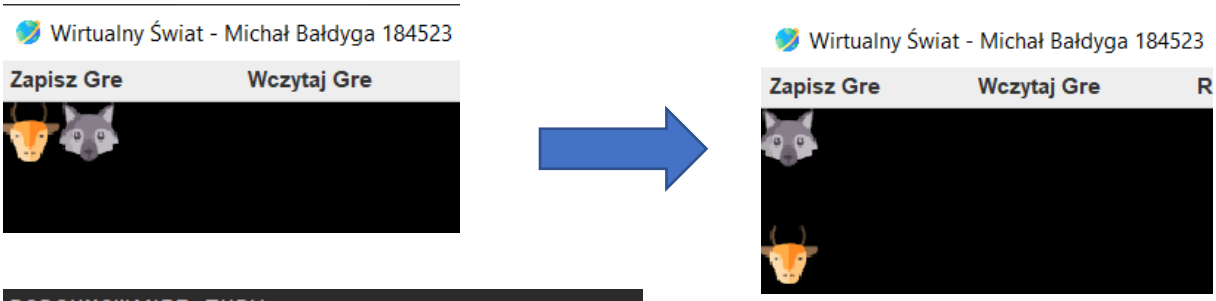
- Klasa Antylopa:

```
public class Antylopa extends Zwierze {

    private static final int sila = 4;
    private static final int inicjatywa = 4;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/antylopa.png");

    public Antylopa(Polozenie polozenie, Swiat swiat) { super(sila, ikona, inicjatywa, polozenie, swiat); }

    @Override
    public void akcja() {...}

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new Antylopa(polozenie, swiat); }

    @Override
    public String getNazwa() { return "Antylopa"; }

    @Override
    public boolean czyZrobilUnik() {...}
}
```

- specyfika metody akcja():

Antylopa zamiast poruszyć się o jedno pole tak jak wszystkie zwierzęta, poruszyła się o dwa pola, ponieważ jej zasięg jest większy.

- specyfika metody kolizja():



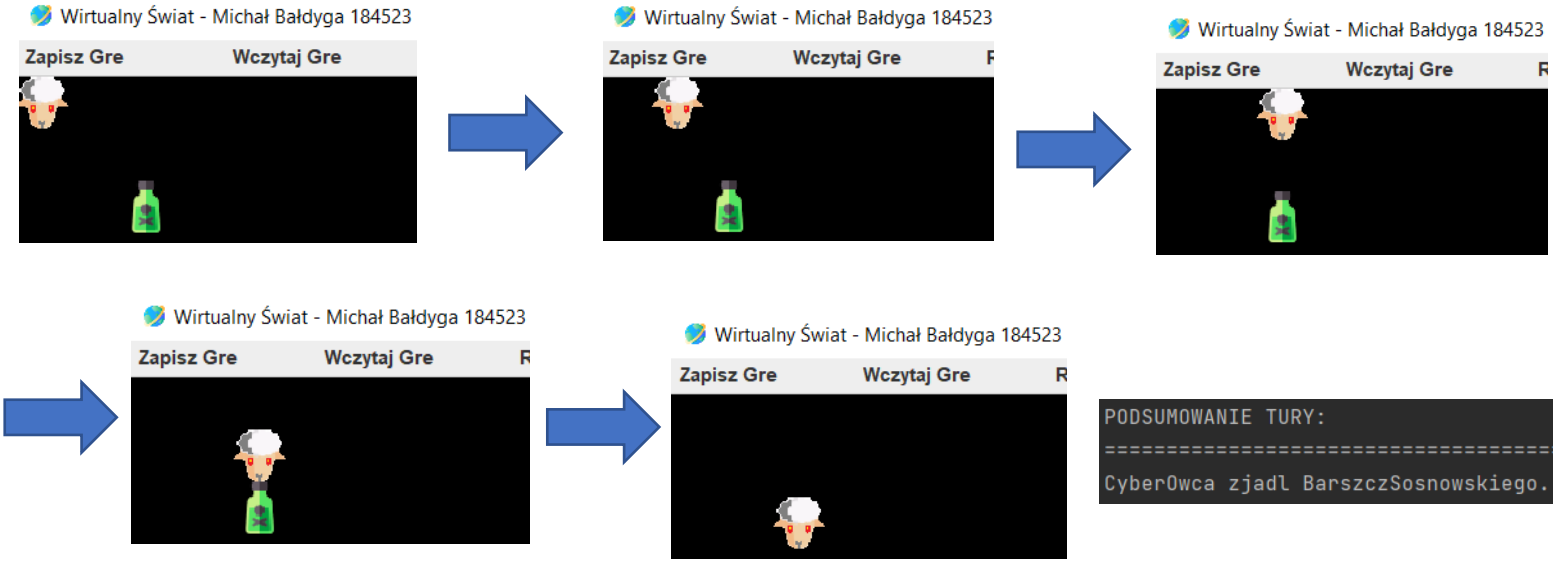

```
PODSUMOWANIE TURY:
==========================================
Wilk atakuje Antylopa. Antylopa robi unik.
```

Antylopa ma 50% szans na wykonanie uniku. Na powyższym przykładzie widzimy, że antylopa robi unik przed atakiem wilka (udaje się w dół), a następnie wykonuje swój ruch (w tym przypadku również w dół).

- Klasa CyberOwca:

```java
public class CyberOwca extends Zwierze {

    private static final int sila = 11;
    private static final int inicjatywa = 4;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/cyberowca.png");

    public CyberOwca(Polozenie polozenie, Swiat swiat) { super(sila, ikona, inicjatywa, polozenie, swiat); }

    @Override
    public void akcja() {...}

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new CyberOwca(polozenie, swiat); }

    @Override
    public String getNazwa() { return "CyberOwca"; }

    private BarszczSosnowskiego szukajBarszczu() {...}

    private int znajdzNajblizszy(Vector<Integer> tab) {...}
}
```

- specyfika metody akcja() i kolizja():



Na powyższym przykładzie widać, że cyberowca kieruje się w stronę barszczu sosnowskiego, a następnie go zjada.

- Klasa Lis:

```java
public class Lis extends Zwierze {

    private static final int sila = 3;
    private static final int inicjatywa = 7;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/lis.png");

    public Lis(Polozenie polozenie, Swiat swiat) { super(sila, ikona, inicjatywa, polozenie, swiat); }

    @Override
    public void akcja() {...}

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new Lis(polozenie, swiat); }

    @Override
    public String getNazwa() { return "Lis"; }
}
```

- specyfika metody akcja():

Na powyższym przykładzie widzimy, że lis spośród czterech sąsiadujących pól wybrał te, które nie było zajmowane przez silniejszy organizm.

- Klasa Owca:

```
public class Owca extends Zwierze {

    private static final int sila = 4;
    private static final int inicjatywa = 4;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/owca.png");

    public Owca(Polozenie polozenie, Swiat swiat) { super(sila, ikona, inicjatywa, polozenie, swiat); }

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new Owca(polozenie, swiat);

    @Override
    public String getNazwa() { return "Owca"; }
}
```

- Klasa Wilk:

```
public class Wilk extends Zwierze {

    private static final int sila = 9;
    private static final int inicjatywa = 5;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/wilk.png");

    public Wilk(Polozenie polozenie, Swiat swiat) { super(sila, ikona, inicjatywa, polozenie, swiat); }

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new Wilk(polozenie, swiat); }

    @Override
    public String getNazwa() { return "Wilk"; }
}
```

- Klasa Żółw:

```
public class Zolw extends Zwierze {

    private static final int sila = 2;
    private static final int inicjatywa = 1;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/zolw.png");

    public Zolw(Polozenie polozenie, Swiat swiat) { super(sila, ikona, inicjatywa, polozenie, swiat); }

    @Override
    public void akcja() {...}

    @Override
    public boolean czyOdbilAtak(Organizm atakujacy) { return (atakujacy.getSila() < 5); }

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new Zolw(polozenie, swiat); }

    @Override
    public String getNazwa() { return "Zolw"; }
}
```

- specyfika metody akcja():


```
PODSUMOWANIE TURY:
==========================
Nie wykryto zadnych zdarzen.
==========================
```

Żółw podczas tury nie wykonał żadnego ruchu, ponieważ w 75% przypadków nie zmienia swojego położenia.

- specyfika metody kolizja():


```
PODSUMOWANIE TURY:
==================================
Owca atakuje Zolw. Zolw odbil atak.
```

Żółw odbił atak owcy, której siła jest < 5 (4). Napastnik wrócił na swoje pole, a następnie żółw wykonał swój ruch.

3. Implementacja wszystkich gatunków roślin.

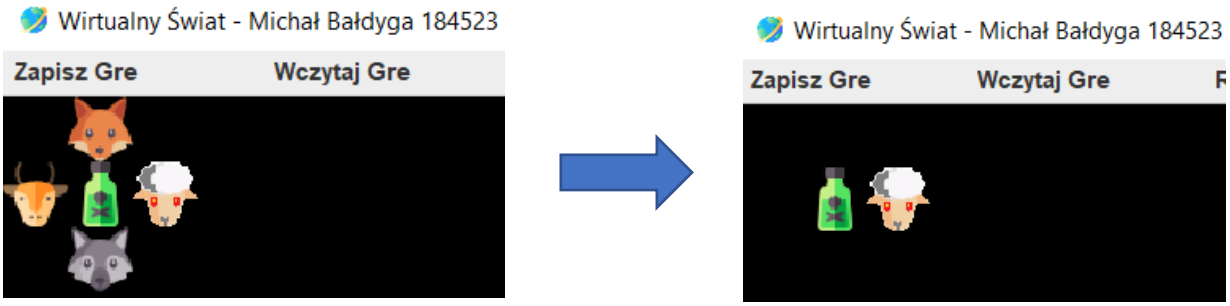
- Klasa abstrakcyjna Roślina:

```java
public abstract class Roslina extends Organizm {

    public Roslina(int sila, ImageIcon ikona, Polozenie polozenie, Swiat swiat) {...}

    @Override
    public void akcja() { zasiej(); }

    @Override
    public void kolizja(Organizm atakujacy) {...}

    protected boolean czySieje() {...}

    protected void zasiej() {...}
}
```

- Klasa BarszczSosnowskiego:

```java
public class BarszczSosnowskiego extends Roslina {

    private static final int sila = 10;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/barszcz.png");

    public BarszczSosnowskiego(Polozenie polozenie, Swiat swiat) { super(sila, ikona, polozenie, swiat); }

    @Override
    public void akcja() {...}

    @Override
    public void kolizja(Organizm atakujacy) {...}

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) {...}

    @Override
    public String getNazwa() { return "BarszczSosnowskiego"; }
}
```

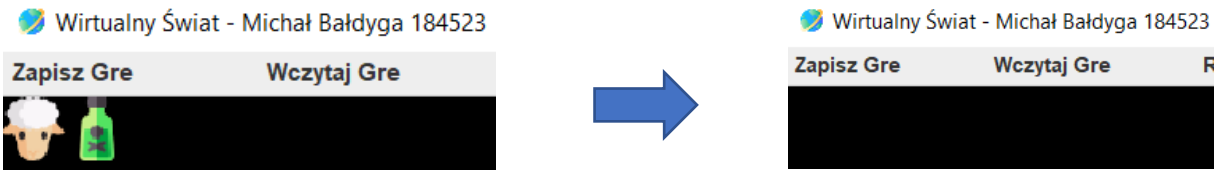
- specyfika metody akcja():




```
PODSUMOWANIE TURY:
====================================
BarszczSosnowskiego zatrul Antylopa.
BarszczSosnowskiego zatrul Lis.
BarszczSosnowskiego zatrul Wilk.
```

Aby zobrazować działanie barszczu, obsadziłem go zwierzętami, którym chwilowo wstrzymałem możliwość poruszania się. Widzimy, że po zakończonej turze, wszystkie zwierzęta sąsiadujące z barszczem (poza cyberowcą) zginęły.

- specyfika metody kolizja():




```
PODSUMOWANIE TURY:
==================================
Owca zjadl BarszczSosnowskiego.
BarszczSosnowskiego zatrul Owca.
```


```
PODSUMOWANIE TURY:
====================================
CyberOwca zjadl BarszczSosnowskiego.
```

Na powyższym przykładzie widzimy, że zwierzę które zjadło barszcz ginie. Jedynie cyber-owca jest odporna.

- Klasa Guarana:

```java
public class Guarana extends Roslina {

    private static final int sila = 0;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/guarana.png");

    public Guarana(Polozenie polozenie, Swiat swiat) { super(sila, ikona, polozenie, swiat); }

    @Override
    public void kolizja(Organizm atakujacy) {...}

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new Guarana(polozenie, swiat); }

    @Override
    public String getNazwa() { return "Guarana"; }
}
```

- specyfika metody kolizja():


```
PODSUMOWANIE TURY:
==========================
Lis zjadl Guarana.
Guarana dodaje sily Lis.
```

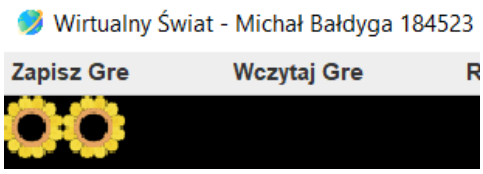
Lis zjada guaraną, co zwiększa jego siłę o 3 punkty.

- Klasa Mlecz:

```java
public class Mlecz extends Roslina {

    private static final int sila = 0;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/mlecz.png");

    public Mlecz(Polozenie polozenie, Swiat swiat) { super(sila, ikona, polozenie, swiat); }

    @Override
    public void akcja() {...}

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new Mlecz(polozenie, swiat); }

    @Override
    public String getNazwa() { return "Mlecz"; }
}
```

- specyfika metody akcja():

```
@Override
public void akcja() {
        int liczbaProbZasiania = 3;
        for (int i = 0; i < liczbaProbZasiania; i++)
            super.zasiej();
}
```



Mlecz podejmuje 3 próby rozprzestrzeniania się w jednej turze.

- Klasa Trawa:

```
public class Trawa extends Roslina {

    private static final int sila = 0;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/trawa.png");

    public Trawa(Polozenie polozenie, Swiat swiat) { super(sila, ikona, polozenie, swiat); }

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new Trawa(polozenie, swiat);

    @Override
    public String getNazwa() { return "Trawa"; }
}
```

- Klasa WilczeJagody:

```
public class WilczeJagody extends Roslina {

    private static final int sila = 99;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/jagody.png");

    public WilczeJagody(Polozenie polozenie, Swiat swiat) { super(sila, ikona, polozenie, swiat); }

    @Override
    public void kolizja(Organizm atakujacy) {...}

    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return new WilczeJagody(polozenie, swiat); }

    @Override
    public String getNazwa() { return "WilczeJagody"; }
}
```

- specyfika metody kolizja():


```
PODSUMOWANIE TURY:
==============================
CyberOwca zjadl WilczeJagody.
WilczeJagody zatrul CyberOwca.
```

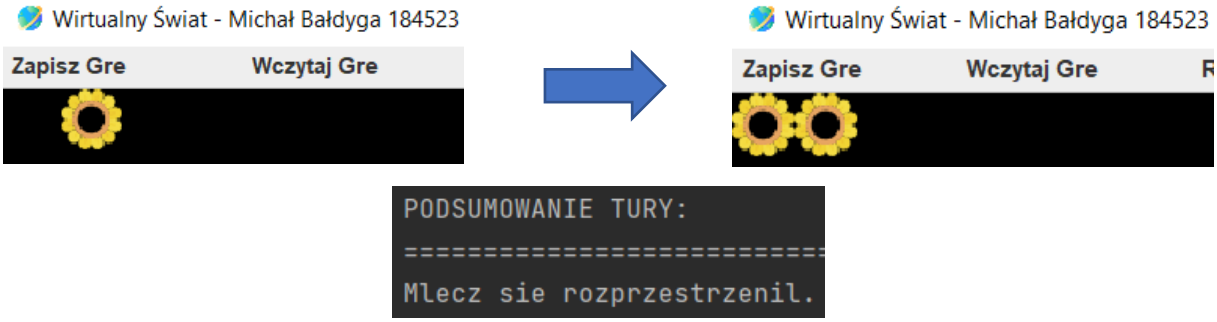
Zwierzę, które zje wilcze jagody ginie.

4. Rozmnażanie się zwierząt i rozprzestrzenianie się roślin.

a) Zwierzęta:

Podany przykład obrazuje rozmnażanie się zwierząt. Jeden z wilków staje na pole, na którym znajduje się osobnik tego samego gatunku. Dochodzi do rozmnożenia, a następnie wilki, które nie poruszyły się w tej turze, wykonują swoje ruchy.

b) Rośliny:

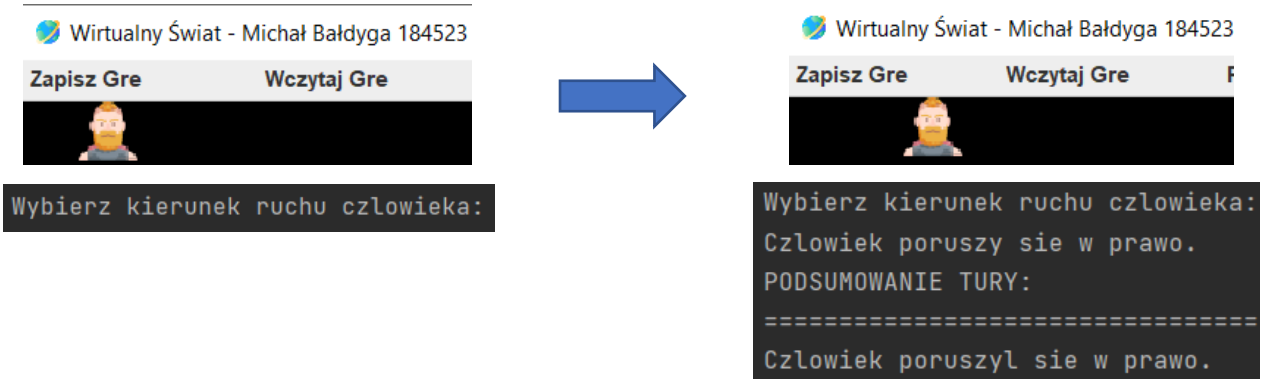


Mlecz sieje nową roślinę na sąsiednim polu.

5. Implementacja Człowieka poruszanego za pomocą strzałek na klawiaturze oraz jego specjalnej umiejętności:

```java
public class Czlowiek extends Zwierze {

    private static final int sila = 5;
    private static final int inicjatywa = 4;
    private static final ImageIcon ikona = new ImageIcon( filename: "images/czlowiek.png");
    private int pozostalyCzasUmiejetnosci;
    private int pozostalyCzasLadowania;
    private String kierunekRuchu;

    public Czlowiek(Polozenie polozenie, Swiat swiat) {...}
    @Override
    public void akcja() {...}
    @Override
    public void kolizja(Organizm atakujacy) {...}
    @Override
    public boolean czyOdbilAtak(Organizm atakujacy) { return (pozostalyCzasUmiejetnosci > 0); }
    @Override
    public Organizm stworzPotomka(Polozenie polozenie, Swiat swiat) { return null; }
    @Override
    public String przygotujDoZapisu() {...}
    public void aktywujUmiejetnosc() {...}
    public void aktualizujUmiejetnosc() {...}

    //Getery
    @Override
    public String getNazwa() { return "Czlowiek"; }
    public String getKierunekRuchu() { return kierunekRuchu; }

    // Setery
    public void setKierunekRuchu(String kierunek) { this.kierunekRuchu = kierunek; }
    public void setPozostalyCzasUmiejetnosci(int czas) { pozostalyCzasUmiejetnosci = czas; }
    public void setPozostalyCzasLadowania(int czas) { pozostalyCzasLadowania = czas; }
}
```

- specyfika metody akcja():

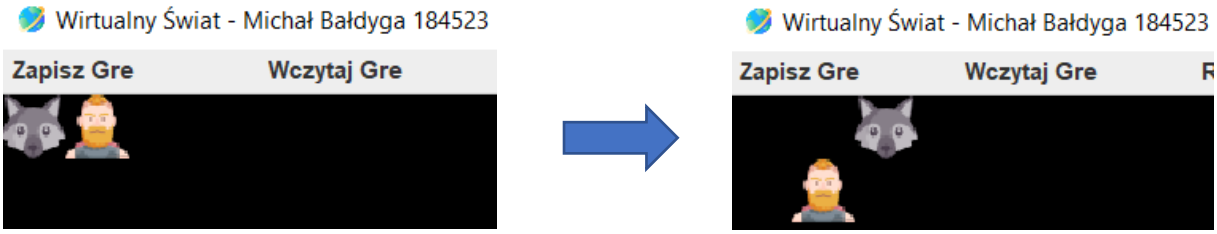



```
Wybierz kierunek ruchu czlowieka:
```

```
Wybierz kierunek ruchu czlowieka:
Czlowiek poruszy sie w prawo.
PODSUMOWANIE TURY:
================================
Czlowiek poruszyl sie w prawo.
```

Na początku tury determiniujemy kierunek ruchu człowiek. Gdy nadejdzie jego kolej postać uda się w wybranym kierunku.

- specyfika metody kolizja() – specjalna umiejętność (Tarcza Alzura):




```
Wybierz kierunek ruchu czlowieka:
Czlowiek poruszy sie w dol.
Umiejetnosc czlowieka zostala aktywowana.
PODSUMOWANIE TURY:
==========================================
Wilk atakuje Czlowiek. Czlowiek odbil atak.
Czlowiek poruszyl sie w dol.
```

Podany przykład obrazuje działanie umiejętności specjalnej (Tarczy Alzura). Na początku tury człowiek aktywuje swoją umiejętność. Zwierzę, które atakuje człowieka, zostaje przesunięte na losowe sąsiednie pole. Następnie człowiek wykonuje swój ruch.

6. Implementacja możliwości zapisania do pliku i wczytania z pliku stanu wirtualnego świata.

Poniżej zamieszczam kod dwóch głównych funkcji służących do zapisywania i odczytywania stanu świata oraz przykładowy plik tekstowy, który przechowuje dany zapis.

```java
public void zapiszStan() {
    try {
        BufferedWriter plik = new BufferedWriter(new FileWriter( fileName: "zapis.txt"));
        for (Organizm org : kolejka)
            plik.write(org.przygotujDoZapisu());
        plik.close();
        System.out.println("Zapisano stan gry.\n");
    } catch(Exception e) {
        System.out.println("Nie udalo sie zapisac stanu gry.\n");
        e.printStackTrace();
    }
}
```

```java
public void wczytajStan(JFrame frame) {
    try {
        File plik = new File( pathname: "zapis.txt");
        Scanner skaner = new Scanner(plik);

        wyczyscSwiat(frame);

        while (skaner.hasNextLine()) {
            String linia = skaner.nextLine();
            wczytajOrganizm(linia);
        }

        skaner.close();
        rysujSwiat(frame);
        System.out.println("Wczytano stan gry.\n");

    } catch (FileNotFoundException e) {
        System.out.println("Nie udalo sie wczytac stanu gry.\n");
        e.printStackTrace();
    }
}
```

```
Lis;3;14;11
Lis;3;17;12
Wilk;9;10;12
Wilk;9;1;15
Czlowiek;5;18;10;0;0
Antylopa;4;5;4
CyberOwca;11;0;0
Owca;4;4;9
Antylopa;4;5;0
CyberOwca;11;19;16
Owca;4;0;6
Zolw;2;6;14
Zolw;2;4;17
BarszczSosnowskiego;10;19;11
Guarana;0;0;3
Mlecz;0;15;8
Trawa;0;8;2
WilczeJagody;99;4;12
BarszczSosnowskiego;10;12;19
Guarana;0;13;13
Mlecz;0;3;15
Trawa;0;6;5
WilczeJagody;99;12;14
```