Ročníkový projekt - Výškomapa

Gymnázium Arabská

Felix Navrátil

Vyučující: Mgr. Jan Lána

Předmět programování

20. dubna 2023

1 Anotace

Tento ročníkový projekt se zabývá problematikou generování nebo konverzí mapy. Jsou zde vysvětleny problémy, mezi které patří generování semirealistického terénu, generování terénu kolem uživatelem vybraných bodů, konverzí mapy z internetu do mé aplikace nebo pohled ze strany. Také se zde píše o tom jak jsem problémy vyřešil nebo nevyřešil.

2 Anotation

This year's project encompasses my implementation of terrain height generation, generation of terrain around user-selected points, side view rendering, and height map transformation from the internet. I will now share how I addressed or did not address the challenges associated with these tasks.

Obsah

1 Anotace **2**

2 Anotation **2**

3 Úvod **4**

4 Rozložení aplikace **5**

4.1 Okno na generování mapy 5

4.2 Okno na konvertování mapy 5

4.3 Nastavení 6

5 Generování terénu **6**

5.1 Základní generování terénu 7

5.2 Generování terénu kolem uživatelem vybraných bodů 10

6 Pohled za strany **15**

7 Přeměna výškomapy **17**

8 Závěr **23**

3 Úvod

Jako ročníkový projekt jsem si vybral výškomapu, protože mi přišlo zajímavé, jak vyřešit generování semirealistického výškového terénu. Programoval jsem v javaFX a jako IDE jsem používal Intellij idea. Při programování jsem narazil na mnoho problému, mezi které patří například, jakou výšku mám jednotlivým bodům přiřadit, jak generovat terén kolem bodů přidaných uživatelem nebo celkový design aplikace. Aplikace bude tedy umět vytvořit mapu výškového terénu, vytvořit pohled ze strany, vzít mapu z internetu a vyrobit velmi podobnou mapu a také generovat terén kolem uživatelem vybraných bodů.

4 Rozložení aplikace

Jakmile spustíme aplikaci tak se nám zobrazí okno na kterém jsou 3 tlačítka. První z nich nás přesměruje do okna, které je určeno na generování terénu. Druhé z nich nás odkáže na okno ke konvertování mapy a třetí nám zobrazí nastavení.

4.1 Okno na generování mapy

Jakmile se ocitneme v okně určeném ke generování mapy zobrazí, se nám kromě tlačítka, které nás pošle zpět na domovskou obrazovku, tři další tlačítka(Generuj mapu, Vytvoř mapu, Návod). Tlačítko *Generuj mapu* vytvoří další okno, na kterém se zobrazí vygenerovaná mapa. *Návod* je tlačítko které, když najedete myší, zobrazí uprostřed okna návod jak vytvořit mapu. A *vytvoř mapu* je tlačítko které jakmile je stisknuto, vytváří nové okno určené ke tvorbě mapy.

4.2 Okno na konvertování mapy

V tomto okně jsou 4 tlačítka (Návod, Zde berte obrázky, Konvertuj mapu a Zpět). *Návod* opět zobrazí návod k použití, tlačítko *Zde berte obrázky* nás odkáže na webovou stránku, ze které se budou brát obrázky, *Konvertuj mapu* nám zobrazí scénu, která slouží k samotné přeměně obrázku na mapu. A jako poslední se zde nachází tlačítko *Zpět*, které odkáže uživatele na domovskou obrazovku.

4.3 Nastavení

Zde je možné nastavit základní hodnoty: velikost bodu, schodek (maximální rozdíl výšek dvou bodů vedle sebe), maximální a minimální výška, kterou mohou body mít.

5 Generování terénu

Jako první jsem vyrobil třídu *Výška*, která určuje výšku a barvu bodu, ke kterému je připsána. Nižší místa jsou znázorněny zelenou barvou a vyšší mají barvy do červena, růžova až bíla. Potom jsem napsal třídu *Bod* s konstruktorem, který vyrobí čtverec s připsanou výškou a barvou, jenž následně umístí na obrazovku podle hodnot v konstruktoru. Kromě obrazovky je bod také přidán do 2D pole bodů, které má rozměry *800/bodSize*, na příslušné souřadnice.

```
public Bod(int coordX, int coordY, Vyska vyska1) {
    this.vyska1 = vyska1;
    this.coordX = coordX;
    this.coordY = coordY;
    this.vyska = vyska1.getvalueOfVyska();
    setWidth(bodSize);
    setHeight(bodSize);
    setFill(vyska1.getBarva2());
    setLayoutX(coordX);
    setLayoutY(coordY);
}
```

5.1 Základní generování terénu

Nejdříve si vytvoříme 2D pole bodů s názvem *mapa* a základní výšku.

```
Bod[][] mapa =
new Bod[velikostMapy/bodSize][velikostMapy/bodSize];
Vyska zakladniVyska = new Vyska(MIN + schodek);
```

Následně vytvořím 2 for-cykly, které postupně projdou celou mapou. V něm nejprve zjistíme průměr výšek nejbližších (existujících) sousedů. Jestliže se nacházíme na prvním místě v poli, tedy na souřadnicích [0,0] nemáme žádné sousedy, a tak si vytvoříme bod, ten bude mít výšku
zakladniVyska.

Jestli jsme v prvním řádku, a ne na prvním místě v poli, tak máme jednoho souseda a to vlevo. Průměr se tedy rovná výšce souseda vlevo.

```
vlevo = mapa[i][j-1].getVyska().getvalueOfVyska();
prumer = vlevo;
```

Další možnost je, že se nacházíme v posledním sloupci pole a tedy máme jen 2 sousedy (nahoře a nahoře vlevo). Průměr se tím pádem rovná součtu jejich výšek dělen dvěma.

```
nahoreVlevo=mapa[i-1][j-1].getVyska().getvalueOfVyska();
nahore = mapa[i-1][j].getVyska().getvalueOfVyska();
prumer = (nahore + nahoreVlevo)/2;
```

Třetí možnost je, že se budeme nacházet v prvním sloupci v pole. Zde máme také pouze 2 sousedy(nahore a nahore vpravo). Takže se průměr rovná následujícímu.

```
nahore = mapa[i-1][j].getVyska().getvalueOfVyska();
nahoreVpravo = mapa[i-1][j+1].getVyska().getvalueOfVyska();
prumer = (nahore + nahoreVpravo)/2;
```

A poslední možnost je ta, že žádná z těchto podmínek není splněna. V tomto případě máme 3 sousedy (Nahoře, nahoře vlevo a nahoře vpravo). A průměr se tedy rovná součtu výšek sousedů dělen třema.

```
nahoreVlevo=mapa[i-1][j-1].getVyska().getvalueOfVyska();
nahore = mapa[i-1][j].getVyska().getvalueOfVyska();
nahoreVpravo = mapa[i-1][j+1].getVyska().getvalueOfVyska();
prumer = (nahore + nahoreVlevo + nahoreVpravo)/3;
```

Dále je vytvořen interval od *min* do *max*, který zkontroluje jestli *min* není menší než minimální povolená výška *MIN* nebo *max* není větší než maximální povolená výška *MAX*

```
max = prumer + schodek;
min = prumer - schodek;


    if (min<MIN){
    min = MIN;
}else if(max>MAX){
    max = MAX;
}
```

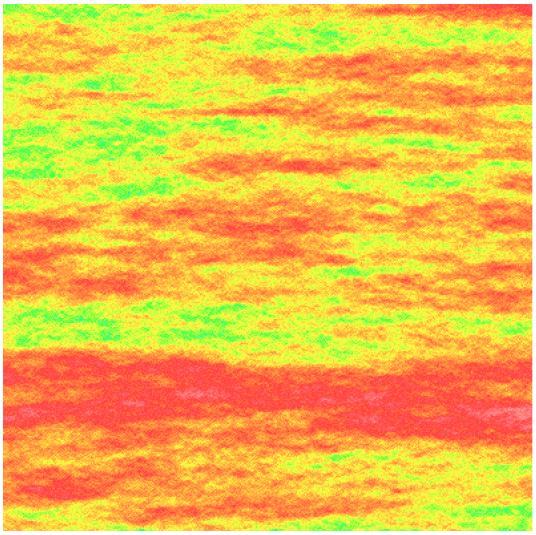
Potom se zavolá metoda *vytvorVysku*, která vybere z intervalu od *min* do *max* náhodné číslo.

```
public int vytvorVysku(int max, int min){
    Random random = new Random();
    return random.nextInt(min,max);
}
```

A jako poslední se vytvoří nový *Bod* s příslušnou výškou a souřadnicemi.

```
vyska = vytvorVysku(max,min);
Vyska v = new Vyska(vyska);
Bod bod = new Bod(i*bodSize, j*bodSize, v);
mapa[i][j]=bod;
```

Výsledná mapa vypadá takto. Zelená a modrá barva značí oblasti s nízkou elevací, místa kde je výška o něco vyšší jsou znázorněna žlutou barvou. A oblasti s červenou a bílou barvou ukazují body s nejvyšší elevací.



Obrázek č. 1 - Vygenerovana mapa

Zdroj: Vlastní

5.2 Generování terénu kolem uživatelem vybraných bodů

Vyrobil jsem *souradnicovyCtverec*, to je průhledný čtverec o velikosti 800X800px, který je vložen na místo, kde bude budoucí mapa. Jakmile uživatel klikne na čtverec, zjistím souřadnice kliknutí a pomocí metody *getKordX/Y*, která vypadá následovně

```
private int x;
public int getKordX(double dX){
    this.x = (int)dX/bodSize;
    return x;
}
```

(analogická metoda k souřadnicím Y je také přítomna) zjistím souřadnice v poli bodů *mapa*. Dále je přítomen *textField* na zadávání výšky. Do něj je

možné napsat libovolnou výšku od minimální do maximální povolené výšky. To je možné díky metodě *isInputValid*.

```
    private boolean isInputValid(String input) {
    try {
        int value = Integer.parseInt(input);
        return value >= MIN && value <= MAX;
    } catch (NumberFormatException e) {
        return false;
    }
}
```

Jakmile uživatel klikne na *souradnicovyCtverec*, zvolí výšku a klikne na tlačítko *Zvolit vysku*, vytvoří se bod s příslušnými souřadnicemi a výškou a ten je zapsán do *poleVybranychBodu*. Dále je možné vybrat čtyři další body nebo kliknout na tlačítko *Generuj Mapu*. Až uživatel klikne na tlačítko *Generuj Mapu*, tak se vygeneruje ,,základní mapa"(mapa generována podle pravidel základního generování terénu viz. kapitola 5.1). Potom pomocí metody *vytvorMapuSMezerama*, která pomocí tohoto for-cyklu změní všechny body, jejichž souřadnice x a y o 10 menší nebo o 10 větší než souřadnice vybraného bodu a změní je na null.

```
for (int p = 0; p < bodyNaMape.length; p++) {
     if (bodyNaMape[p]!=null){

            for (int i = bodyNaMape[p].getCoordY()-10;
            i < bodyNaMape[p].getCoordY()+10; i++) {
```

```
                for (int j = bodyNaMape[p].getCoordX()-10;
                j < bodyNaMape[p].getCoordX()+10; j++) {

                mapa[j][i] = null;
            }
        }
    }
}
```

Dále se zavolá metoda *vyplnMezery2*. Jako první zjistí, o kolik se musí výška bodů okolo vybraného bodu změnit, aby až se body potkají, se jejich výšky o moc nelišily. K tomu je vytvořeno 8 polí celých čísel: *zmenaNahore, zmenaDole, zmenaNalevo, zmenaNapravo, prumerNahore, prumerDole, prumerNalevo, prumerNapravo*. Pole s ,,průměry"zaplním následovně (analogicky to samé k ostatním průměrům):

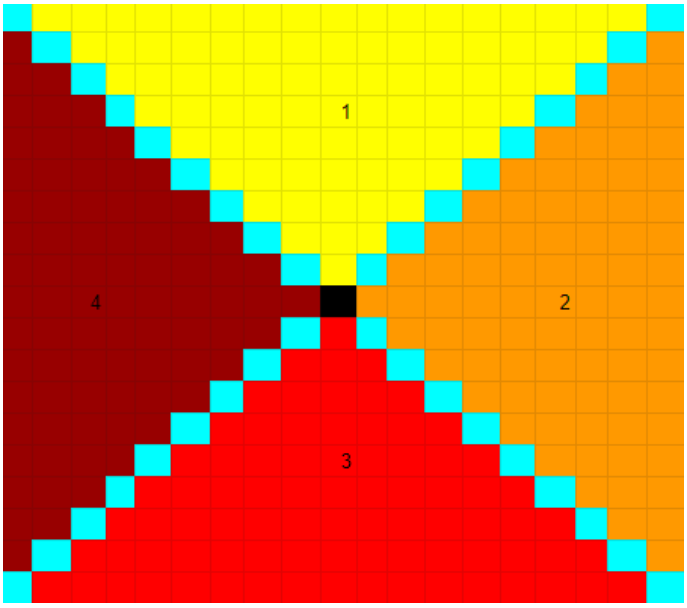
```
for (int p = 0; p < bodyNaMape.length; p++) {
    if (bodyNaMape[p]!=null){
        //zjistovani prumeru kolem prazdne casti mapy
        //zjistovani prumeru nalevo
        for (int i = bodyNaMape[p].getCoordY()-11;
        i <bodyNaMape[p].getCoordY()+11; i++) {
            if (mapa[i][bodyNaMape[p].getCoordX()-11]!=null){
                prumerNalevo[p] +=
                mapa[i][bodyNaMape[p].getCoordX()-11]
                .getVyska().getvalueOfVyska();
```

```
            }
        }
        prumerNalevo[p] /= 21;
    }
}
```

Změna se vypočítá následovně.

```
změna = (výška vybraného bodu - průměr strany)/9
```

Dále vyrobím stejný for-cyklus jako v metodě *vytvorMapuSMezerama*. Jako první zjistím, jestli jsem na souřadnicích vybraného bodu. Pokud ano tak se bod, na kterém se nacházím, rovná vybranému bodu. Jestli ne tak zjistím v jaké v části čtverce jsem. (Modré body znázorňují body, které budou určeny více než jednou.)



Obrázek č. 2 - čtverec znázorňující jeho části

Zdroj: Vlastní

Jestli se nacházím v první/žluté a s ní sousedící modré části čtverce, tak platí následující podmínky: k značí pozici na ose y a l značí pozici na ose x a obě hodnoty patří do intervalu od 0 do 19.

```
k < 10 && (l>k-1 && l <= 19-k)
```

A výška se bude rovnat součtu výšky bodu s y souřadnicí *bodyNaMape[p].getCoordY()-11* a k-násobku změny nahoře.

```
mapa[bodyNaMape[p].getCoordX()-10+l][bodyNaMape[p].getCoordY()-11]
    .getVyska().getvalueOfVyska()+ k*zmenaNahore[p])
```

Pokud se nacházím v druhé/ oranžové části čtverce, tak platí podmínky:

```
l>10 &&((k>(19-l)-1 && k <= 19-(19-l)))
```

A výška se bude rovnat součtu výšky bodu s x souřadnicí *bodyNaMape[p].getCoordX()+11* a *(20-l)*-násobku změny napravo

```
mapa[bodyNaMape[p].getCoordX()+11][bodyNaMape[p].getCoordY()-10+k]
.getVyska().getvalueOfVyska()+ (20-l)*zmenaNapravo[p]
```

Jestli se nacházím v čtvrté/tmavě červené části čtverce tak platí podmínky:

```
l>10 &&((k>(19-l)-1 && k <= 19-(19-l)))
```

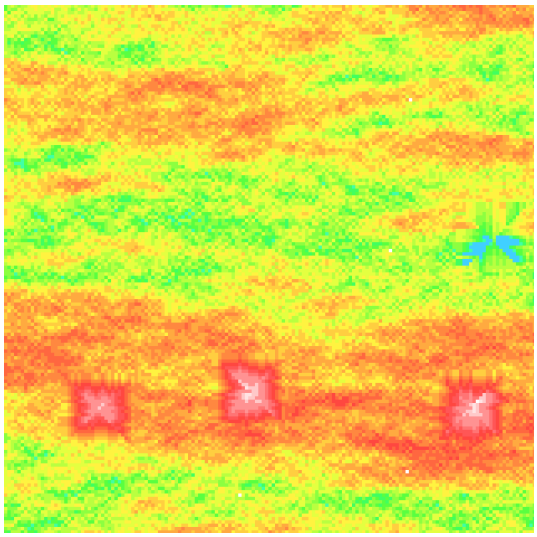
A výška se bude rovnat součtu bodu s x souřadnicí *bodyNaMape[p].getCoordX()-11* a l-násobku změny nalevo.

```
mapa[bodyNaMape[p].getCoordX()-11][bodyNaMape[p].getCoordY()-10+k]
.getVyska().getvalueOfVyska()+ l*zmenaNalevo[p])
```

A body pro které se dosud nesplnily podmínky, budou náležet 3/červené části čtverce a jejich výška bude rovna součtu bodu s y souřadnicí *bodyNaMape[p].getCoordY()+11* a *(20-l)*-násobku změn

```
mapa[bodyNaMape[p].getCoordX()-10+l][bodyNaMape[p].getCoordY()+11]
.getVyska().getvalueOfVyska()+ (20-k)*zmenaDole[p])
```

A výsledná mapa vypadá následovně:



Obrazek č. 3 - mapa vytvořená uživatelem

Zdroj: Vlastní

6 Pohled za strany

Pohled ze strany je funkce, která vytvoří nové okno, na kterém lze pozorovat nejvyšší bod z každého sloupce. Používá se v něm třída *HBod*. Ta vytvoří objekt třídy *Line* s šířkou *bodSize*, příslušnými souřadnicemi a výškou. Souřadnice x budou stejné jako na *mape* a souřadnice y nepotřebuju.

```
    double dilek = (double)(velikostHorizontuY)/(double)(MAX);


    public HBod(int x, double dilek, Vyska vyska){
    this.vyska = vyska;
    setStrokeWidth(bodSize);
    setStartX(x*bodSize);
    setEndX(x*bodSize);
    setStartY(velikostHorizontuY);
    setEndY(velikostHorizontuY - vyska.getvalueOfVyska()*dilek);
    setStroke(Color.BLACK);
}
```

Celé okno s pohledem ze strany se vytvoří pomocí metody *generujHorizont*, která pomocí for-cyklu[1] vyhledá nejvyšší bod z každého sloupce a ten zapíše do pole *HBodů*, ty se následně připíšou na scénu.

```
int[] horizont = new int[mapa.length];
    for (int i = 0; i < mapa.length; i++) {
        int max = Integer.MIN_VALUE;
        int max = Integer.MIN_VALUE;
        for (int j = 0; j < mapa[i].length; j++) {
            if (mapa[i][j] == null){
            }else  if
            (mapa[i][j].getVyska().getvalueOfVyska() > max) {
                max = mapa[i][j].getVyska().getvalueOfVyska();
            }
        }
```

```
        horizont[i] = max;
    }
```

Dále se vygeneruje *souradnicovyCtverec*, který snímá x-ovou souřadnici místa, na které uživatel klikl myší, pomocí *event.getX()*. Tuto souřadnici používá k získání hodnoty z pole horizont a tuto hodnotu nastavuje jako text *labelu* s názvem *labelVyska*. Současně se také nastavuje textová hodnota řádku pomocí metody *setText* na *textRada*. Je zde také přítomno měřítko.

7 Přeměna výškomapy

Nejdříve je inicializováno 2D pole bodů *mapa*, do kterého se v budoucnu budou vkládat vytvořené body. Dále je vytvořeno tlačítko s názvem *Zde berte obrázky*, které nás pomocí metody

```
getHostServices().showDocument();
```

přesměruje na stránku *topographic-map.com*, ze které můžeme brát výstřižky. Dále se zde nachází tlačítko *Návod*, na které když najedeme myší, zobrazí se nám návod k použití. Dále se zde nachází tlačítko konverujMapu, a to nás odkáže na scénu, kde je možné mapu konvertovat. Na této scéně je Label *dragNDrop*[1], na který je možné přesunout soubor této metodě.

```
dragNDrop.setOnDragOver(event -> {
    if (event.getDragboard().hasFiles()) {
            event.acceptTransferModes(TransferMode.COPY);
    }
    event.consume();
```

```
});
```

Zjistí, jestli soubor končí ,, .png ”, a jestli ano tak získá jeho adresu.

```
boolean hasPng = dragboard.getFiles().stream()
    .anyMatch(file -> file.getName().toLowerCase().endsWith(".png"));
    if (hasPng) {
    event.acceptTransferModes(TransferMode.COPY);
    }
```

Jako poslední vytvořím List souboru, do které se dají všechny soubory, jenž byli přetaženy na *dragNDrop*[1].

```
Dragboard dragboard = event.getDragboard();
        boolean success = false;
        if (dragboard.hasFiles()) {
            // Get the list of dropped files
            List<File> files = dragboard.getFiles();
            // Process the dropped files as needed
        for (File file : files) {

            if (file.getName().toLowerCase().endsWith(".png"))
            {
                getAdresa(file.getAbsolutePath());
             } else {
                 file.getAbsolutePath());
            }
```

```
        }
        success = true;
    }
    event.setDropCompleted(success);
    event.consume();
```

Nyní je potřeba získat barvu každého potřebného pixelu a zjistit pomocí ní, jakou výšku bych měl k příslušnému bodu přiřadit. K zjištění barvy daného pixelu slouží metoda *getBarvaPixelu*. Ta zjistí pomocí třídy PixelReader barvu příslušného pixelu.

```
int sirka = (int) ((image.getWidth()/velikostMapy)*x);
int vyska = (int)((image.getHeight()/velikostMapy)*y);
PixelReader pixelReader = image.getPixelReader();
barvaPixelu = pixelReader.getColor(sirka,vyska);
```

Jelikož *barvaPixelu* není hexadecimální číslo, ale má i znaky navíc je nutno odstranit první 2 a poslední 2 znaky. O to se postará metoda *removechars*[1].

```
if (str.length() <= 4) {
    return "";
} else {
    return str.substring(2, str.length() - 2);
}
```

Aby bylo možné k příslušné barvě najít výšku, musel jsem vytvořit stejné měřítko jako je v *topographic-map.com*. Také je nutno zjistit k jakému hexadecimálnímu kódu výšky má hexadecimální kód barvy nejblíže. K tomu existuje metoda *najdiNejmensiRozdil*[1].

```
int minDifference = Integer.MAX_VALUE;
int minIndex = -1;
for (int i = 0; i < hodnoty.length; i++) {
    int difference = Math.abs(barva - hodnoty[i]);
    if (difference < minDifference) {
        minDifference = difference;
        minIndex = i;
    }
}
return hodnoty[minIndex];
```

Dále se vytvoří výška pomocí metody *vytvorVysku* vytvoří výška příslušná k barvě pixelu.

Jakmile je vložena správná soubor do *dragNDrop*[1] zobrazí se obrázek na scéně. Jestli obrázek nemá správné rozměry, tak se smrští nebo rozšíří podle potřeby. V tuto chvíli se na obrazovce objeví tlačítko *Konvertuj mapu*, na které když kliknu tak se spustí for-cyklus, jenž každému bodu v poli přiřadí příslušnou výšku a souřadnice.

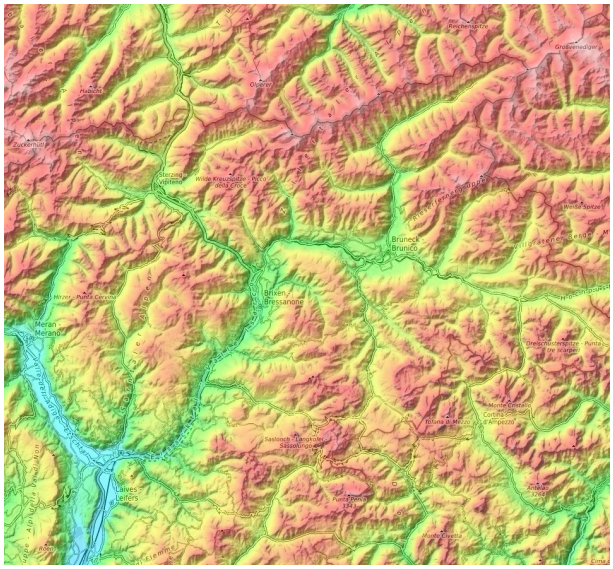
```
for (int i = 0; i<mapa.length;i++){
    for (int j = 0;j<mapa.length;j++){

        mapa[i][j] = new Bod(i*bodSize, j*bodSize,
        vyska1.vytvorVysku(getBarvaPixelu
        (i*bodSize,j*bodSize, image)));

        System.out.println(" i = " + i + " j = " + j);
```

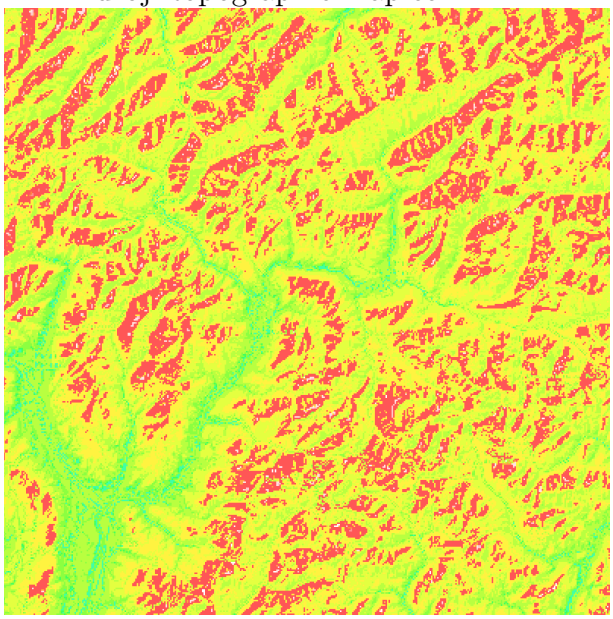
```
            root.getChildren().addAll(mapa[i][j]);
        }
    }
```

Stejně jako u generování mapy, je zde přítomen souřadnicový čtverec a možnost pohledu ze strany. Obojí funguje stejně jako u generování mapy. Výsledná přeměna vypadá následovně:



Obrázek č. 4 - výstřižek z topographic-map.com

Zdroj: topographic-map.com



Obrázek č. 5 - přeměněná mapa

Zdroj: vlastní

8 Závěr

Jako ročníkový projekt jsem si vybral generování výškového terénu, protože jsem předpokládal, že to bude velmi zajímavé. Naučil jsem se pracovat více s javou FX, s dvojdrovměrnými poli, obrázky a také jsem se naučil systematicky řešit bugy. Programování jsem si celkově velmi užil. Program bych vylepšil vyšší proměnlivostí terénu nebo přidáním možností ke generování velmi specifického terénu. Problémy, na které jsem narazil, jsem zvládnul vyřešit někdy s většími a někdy zase s menšími potížemi.

Seznam Tabulek

1 Vygenerovana mapa 10
2 čtverec znázorňující jeho části..... 13
3 mapa vytvořená uživatelem 15
4 výstřižel z topographic-map.com 22
5 přeměněná mapa 22

Reference

[1] OpenAI. Chatgpt, 2023.