## Inlämningsuppgift 2, Mandelbrot

- Beräkna och rita bild av Mandelbrotmängden.
- Färdigskrivet användargränssnitt.
- Ganska mycket och ganska komplext — börja i tid!



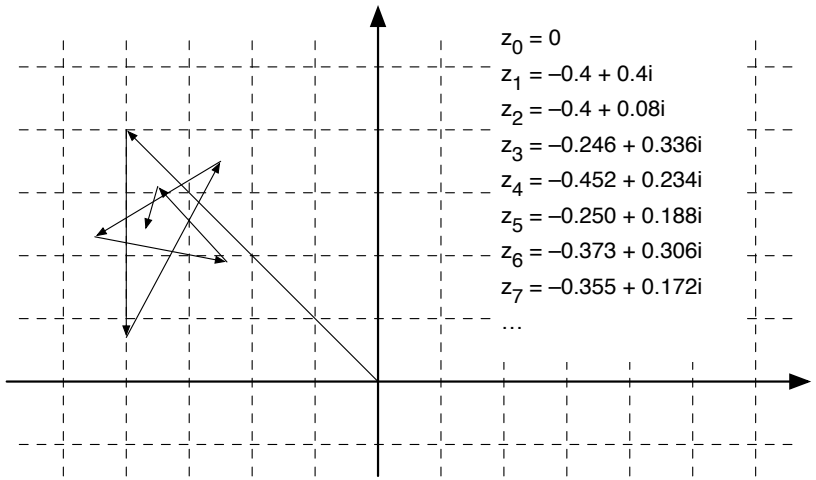
## Mandelbrots talföljd

$$z_k = \begin{cases} 0, & k = 0 \\ z_{k-1}^2 + c, & k = 1, 2, \ldots \end{cases}$$

- $c$ och $z$ är komplexa tal.
- Starta med $z_0 = 0$.
- Beräkna $z_1$ genom att kvadrera $z_0$ och addera $c$.
- Beräkna $z_2$ genom att kvadrera $z_1$ och addera $c$.
- ... och så vidare.



## Talföljden, $c = -0.4 + 0.4i$






## Talföljden, $c = 0.4 + 0.4i$

Med $c = 0.4 + 0.4i$ konvergerar inte talföljden:

$z_0 = 0$
$z_1 = 0.4 + 0.4i$
$z_2 = 0.4 + 0.72i$
$z_3 = 0.042 + 0.976i$
$z_4 = -0.551 + 0.481i$
$z_5 = 0.472 - 0.130i$
$z_6 = 0.606 + 0.277i$
$z_7 = 0.690 + 0.736i$
$z_8 = 0.335 + 1.412i$
$z_9 = -1.492 + 1.348i$
$z_{10} = 0.808 - 3.621i$
$z_{11} = -12.06 - 5.452i$
$z_{12} = 116.1 + 131.9i$
$z_{13} = -3920 + 30613i \ldots$

## Mandelbrotmängden

Mandelbrotmängden $M$ är en delmängd av de komplexa talen.

**Definition**

Ett komplext tal $c$ tillhör $M$ om Mandelbrotföljden med $c$ som startvärde *inte* divergerar.

Exempel:

- $c = -0.4 + 0.4i \in M$.
- $c = 0.4 + 0.4i \notin M$.

Men hur vet man om talföljden konvergerar eller inte?

**Sats**

Om det för något $k$ gäller att $|z_k| > 2$ så divergerar Mandelbrotföljden.



## Algoritm

Man bestämmer ett heltal `MAX_ITER`, 200 kan vara lagom.

1. Välj ett komplext tal $c$
2. $k = 0, z_0 = 0 + 0i$
3. så länge $k <$ `MAX_ITER` och $|z_k| \leq 2$:
   Öka $k$
   Beräkna $z_k$
4. Om $k <$ `MAX_ITER`
   Följden är divergent (säkert)
   annars
   Följden är konvergent och $c$ tillhör $M$ (nästan säkert)



## Vi behöver komplexa variabler

```
/** Skapar en komplex variabel med realdelen re och
    imaginärdelen im */
Complex(double re, double im);

/** Tar reda på realdelen */
double getRe();

/** Tar reda på imaginärdelen */
double getIm();

/** Tar reda på talets absolutbelopp i kvadrat */
double getAbs2();

/** Adderar det komplexa talet c till detta tal */
void add(Complex c);

/** Multiplicerar detta tal med det komplexa talet c */
void mul(Complex c);
```



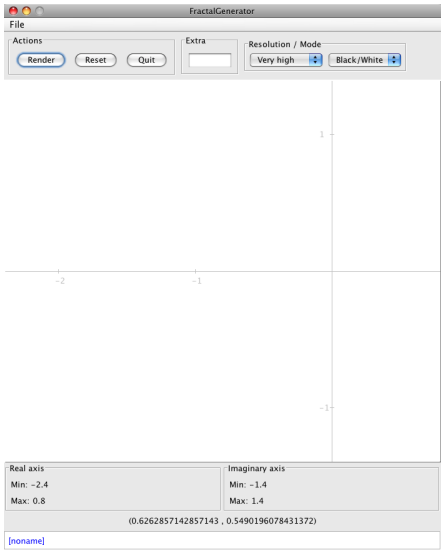
## Kommentarer till `Complex`

Man kan alltså skriva så här:

```
Complex a = new Complex(0.4, -0.8);
Complex b = new Complex(1, 2);
b.add(a); // b = 1.4 + 1.2i
b.mul(a); // b = (1.4 + 1.2i) * (0.4 - 0.8i) = 1.52 - 0.64i
```

- Skriv ett eget testprogram för att kontrollera att klassen fungerar.
- Tänk särskilt på `z.mul(z)`.
- `getAbs2` för att man ska slippa att dra kvadratrot.
  Mandelbrotalgoritmen måste modifieras något.

## MandelbrotGUI



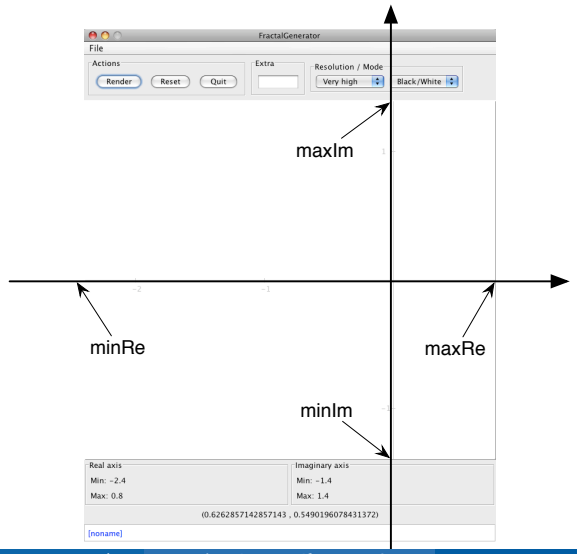



## Huvudprogram (ofullständigt)

```
class Mandelbrot {
    public static void main(String[] args) {
        MandelbrotGUI gui = new MandelbrotGUI();
        while (true) {
            switch (gui.getCommand()) {
                case MandelbrotGUI.RENDER: ...; break;
                case MandelbrotGUI.RESET:  ...; break;
                case MandelbrotGUI.QUIT:   ...; break;
                case MandelbrotGUI.ZOOM:   ...; break;
            }
        }
    }
}
```

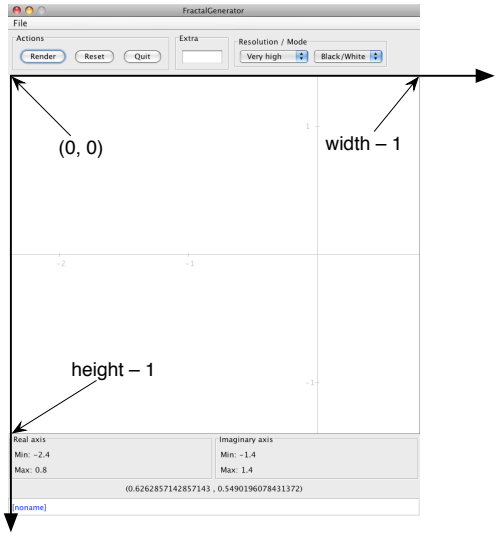


## Koordinatsystem: talplanet






## Koordinatsystem: ritsystemet



## Samband mellan koordinatsystem

Vi ska åskådliggöra det komplexa talplanet (den del som syns i användargränssnittet) med en bild. Varje punkt i ritsystemet motsvaras av ett komplext tal (man lägger koordinatsystemen "ovanpå" varandra):

```
(0, 0)                 motsvaras av   (minRe, maxIm)
(width-1, height-1)    motsvaras av   (maxRe, minIm)
```

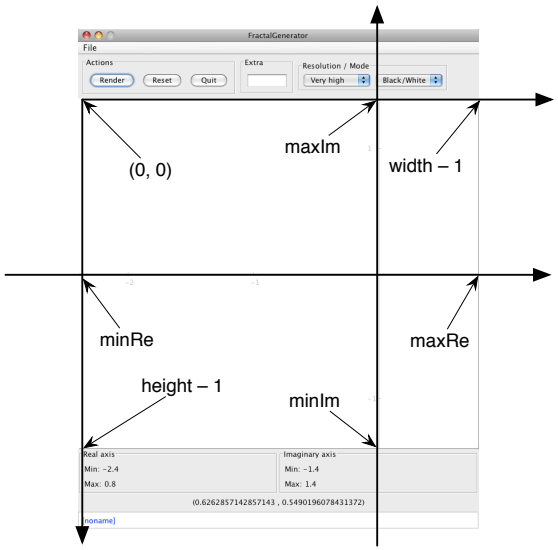
Skapa en matris och fyll den med komplexa tal:

```
Complex[][] complex = new Complex[height][width];
for (int i = 0; i < height; i++) {
    for (int j = 0; j < width; j++) {
        complex[i][j] = new Complex(<re>, <im>);
    }
}
```

där `<re>` och `<im>` räknas ut med formler så att värdena blir korrekta.



## Samband mellan koordinatsystem, bild






## Rita enkel bild

Skapa en bildmatris och fyll den med `Color`-objekt, visa bilden:

```
Color[][] picture = new Color[height][width];
for (int i = 0; i < height; i++) {
    for (int j = 0; j < width; j++) {
        picture[i][j] = <någon snygg färg som beräknas utgående
                         från värdet på complex[i][j]>;
    }
}
gui.putData(picture, 1, 1);
```

Börja med att rita en färgad cirkel med olika färger i de olika kvadranterna. Det ger en kontroll av att du har räknat rätt.



## Rita med olika upplösning

Användaren kan välja upplösning (`VERY_HIGH`, `HIGH`, `MEDIUM`, `LOW`, `VERY_LOW`). När `VERY_HIGH` används motsvaras varje pixel i `picture`-matrisen av en punkt i `complex`-matrisen. (Det är detta vi använt tidigare.)

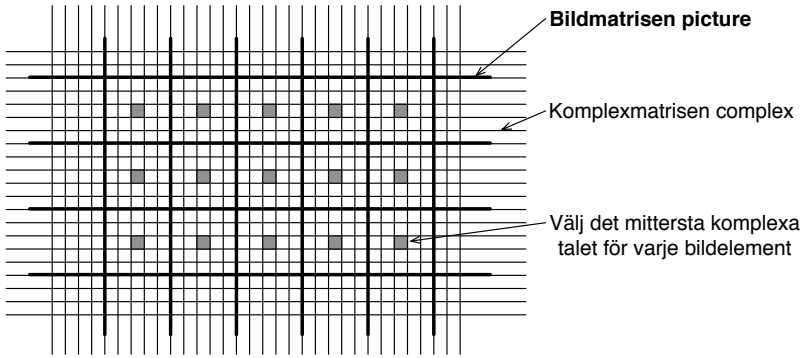
Med andra upplösningar ska `picture`-matrisen vara mindre än `complex`-matrisen. En pixel i `picture`-matrisen motsvarar till exempel $3 \times 3$ (`HIGH`), $5 \times 5$ (`MEDIUM`), $7 \times 7$ (`LOW`), $9 \times 9$ (`VERY_LOW`) punkter i `complex`-matrisen. Bilden av cirkeln ska bli "taggig".

Alltså:

- Hämta upplösningen från användargränssnittet.
- Beräkna `pixelSize` (1, 3, 5, 7, 9).
- Skapa lagom stor `picture`-matris.
- Välj rätt punkter i `complex`-matrisen (se nästa bild).
- Rita bilden:

```
gui.putData(picture, pixelSize, pixelSize);
```

## Upplösningsbild, `pixelSize` = 5






## Rita bild av Mandelbrotmängden

Generering av bild:

```
for (int i = 0; i < ...; i++) {
    for (int j = 0; j < ...; j++) {
        picture[i][j] = <någon snygg färg som beräknas utgående
                         från värdet på complex[i][j]>;
    }
}
```

För att rita en svartvit bild av Mandelbrotmängden använder man `complex[i][j]` som startvärdet $c$ och itererar Mandelbrot-följden. Om följden konvergerar sätter man punkten till svart, annars till vit.



## Rita färgbild, utökning

Bilden blir mycket snyggare om man inte markerar alla punkter där Mandelbrotföljden divergerar med vitt utan med en färg som talar om "hur snabbt" följden divergerar.

- Skapa en färgkarta. Jämför med `grayLevels` i laboration 8.
- Iterera Mandelbrotföljden i varje punkt.
- Välj färg ur färgkartan utgående från hur många steg man itererade innan man kunde avgöra att följden divergerar.

Valfri utökning:

- Använd Extra-rutan för att mata in data till programmet, till exempel antalet iterationer.