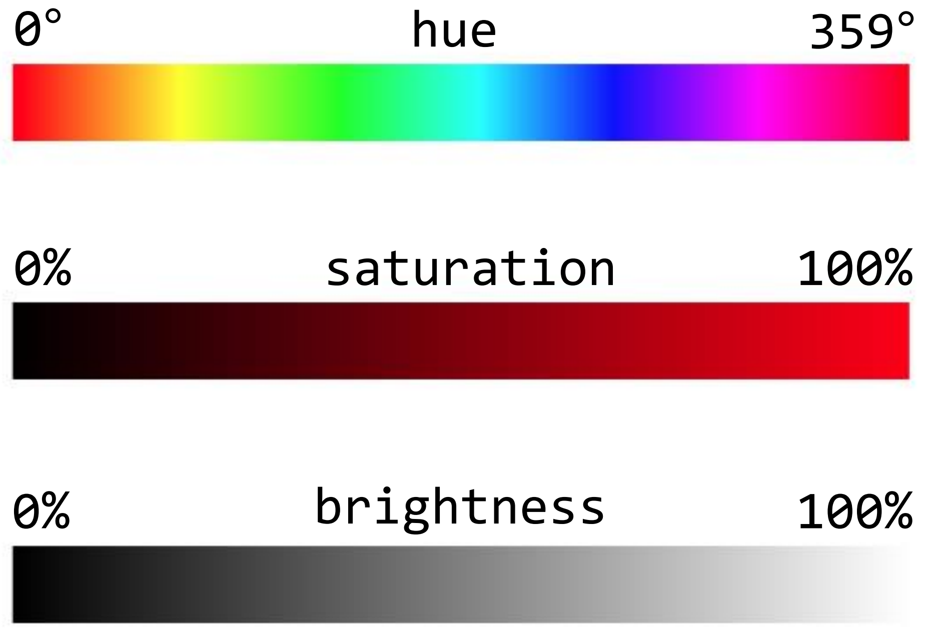


Java Programming

CPT111 – Week 7

Erick Purwanto

Xi'an Jiaotong-Liverpool University
西交利物浦大学

CPT111 Java Programming

Week 7 Exercise and Coursework-1

Introduction to Objects

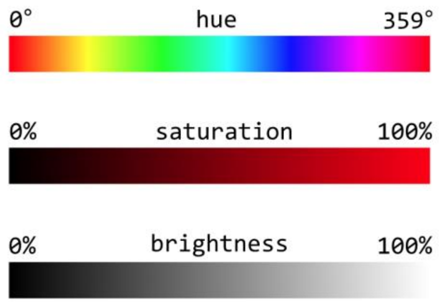
Coding and Submission

- Coding in your NetBeans
  - Start with the skeleton code given in the course LMO
  - Add your own test cases

- Submitting into Learning Mall Quiz
  - Do not submit the whole class
  - Only submit the constructor or the method
    - read carefully the instructions
  - You can submit your own private helper method
    - but do not add another public methods

- Any questions?
  - Please ask in the forum

Exercise #7 ColorHSB



- Complete a class ColorHSB that represents a color in hue – saturation – brightness (HSB) format
  - A skeleton file ColorHSB.java is given

- A color in HSB format is composed of three components:
  - Hue is an integer between 0 and 359
    - It represents a pure color on the color wheel in degrees (°), with 0° for red, 120° for green, and 240° for blue
  - Saturation is an integer between 0 and 100
    - It represents the purity of the hue in percentage (%)
  - Brightness is an integer between 0 and 100
    - It represents the percentage (%) of white that is mixed with the hue

Exercise #7.1 ColorHSB Constructor

- Complete the constructor of the class ColorHSB
- It takes three arguments: h, s, and b
  - and creates a new ColorHSB object with hue h, saturation s, and brightness b

- Test cases:

```
ColorHSB green = new ColorHSB(100, 100, 50);

System.out.println(green);                    → (100, 100, 50)
```

Exercise #7.2 ColorHSB toString

- Complete the method toString of the class ColorHSB
- It takes no argument
    - and return a string composed of the integers for hue, saturation, and brightness (in that order), separated by commas, and enclosed in parentheses

- Test cases:

```
ColorHSB green = new ColorHSB(100, 100, 50);

System.out.println(green);                    → (100, 100, 50)
```

Exercise #7.3 ColorHSB isGrayscale

- Complete the method isGrayscale of the class ColorHSB
- A color in HSB format is a shade of gray if either its saturation or brightness component is 0% (or both)
  - return true if it is, and false otherwise

- Test cases:

```
ColorHSB orange = new ColorHSB(25, 100, 100);
ColorHSB gray = new ColorHSB(0, 0, 50);

System.out.println(orange.isGrayscale());             → false
System.out.println(gray.isGrayscale());               → true
```

Exercise #7.4 ColorHSB squareDist

- Complete the method squareDist of the class ColorHSB that returns the squared distance between two colors
- The squared distance between two colors (h_1,s_1,b_1) and (h_2,s_2,b_2) is defined to be

$$\min\{(h_1 - h_2)^2,\ (360 - |h_1 - h_2|)^2\} + (s_1 - s_2)^2 + (b_1 - b_2)^2$$

- Test cases:

```
ColorHSB green = new ColorHSB(100, 100, 50);
ColorHSB orange = new ColorHSB(25, 100, 100);
ColorHSB gray = new ColorHSB(0, 0, 50);

int distGreenOrange = green.squareDist(orange);
System.out.println(distGreenOrange);              → 8125
System.out.println(gray.squareDist(orange))       → 13125
```

CW1 Week #7

- Complete a class Clock that represents time on a 24-hour clock, such as 00:00, 15:30, or 23:59
    - Time is measured in hours (00 – 23) and minutes (00 – 59)
    - Times are ordered from 00:00 (earliest) to 23:59 (latest)
    - A skeleton file Clock.java is given

- Each part of CW1 #7.x is worth 50 points

CW1 #7.1 Clock Constructor 1

- Complete the first constructor of the class Clock
- It takes two arguments: h and m
    - and creates a new clock object whose initial time is h hours and m minutes

- Test cases:

```
Clock clock1 = new Clock(1, 0);

System.out.println(clock1);                    → 01:00
```

CW1 #7.2 Clock Constructor 2

- Complete the second constructor of the class Clock
- It takes one string argument: s
  - s is composed of two digits, followed by a colon, followed by two digits, so the format is HH:MM such as 02:30
  - it creates a new clock object whose initial time is HH hours and MM minutes

- Test cases:

```
Clock clock2 = new Clock("02:30");

System.out.println(clock2);                          → 02:30
```

CW1 #7.3 Clock toString

- Complete the method toString of the class Clock
- It returns a string representation of this clock, using the format HH:MM
    - that is, the format is the hours (2 digits), followed by a colon, followed by the minutes (2 digits), for example, 00:00 and 23:59

- Test cases:

```
Clock clock1 = new Clock(1, 0);
Clock clock2 = new Clock("02:30");

System.out.println(clock1);                    → 01:00
System.out.println(clock2);                    → 02:30
```

CW1 #7.4 Clock isEarlierThan

- Complete the method isEarlierThan of the class Clock
- It returns true if and only if the time on *this* clock (the current referenced object) is earlier than the time on *that* clock (in the argument)
- Test cases:

```
Clock clock1 = new Clock(1, 0);
Clock clock2 = new Clock("02:30");

System.out.println(clock1.isEarlierThan(clock2));     → true
```

CW1 #7.5 Clock tick

- Complete the method tick of the class Clock
- It adds 1 minute to the time on this clock
  - for example, one minute after 01:00 is 01:01; one minute after 23:59 is 00:00

- Test cases:

```
Clock clock1 = new Clock(1, 0);
System.out.println(clock1);                          → 01:00

clock1.tick();
System.out.println(clock1);                          → 01:01
```

CW1 #7.6 Clock tock

- Complete the method tock of the class Clock
- It adds delta minute(s) to the time on this clock, where delta is a positive integer
    - for example, 100 minutes after 02:30 is 04:10
- Note that must **not** use the method in CW1 #7.5 tick().

- Test cases:

```
Clock clock2 = new Clock("02:30");
System.out.println(clock2);                    → 02:30

clock2.tock(100);
System.out.println(clock2);                    → 04:10
```

Thank you for your attention!

- This is the end of Week 7 Exercise and Coursework 1 Task Sheet