Design Recipe

For solving Pixel Magic
CPE101

Design Recipe

- You should create a blueprint of your solution before typing your code.
- The design recipe lets you efficiently design a solution.
- The idea is taken from this book:
    - "How To Design Programs" by Matthias Felleisen, et. al.

Step 1: Define Data

- The basic question that you're trying to answer is this: "What data types/structures in my programming language will I use to represent the elements of my problem?"

Representing an image in your program

```
P3
300 168
255
245 246 241
245 246 241
245 246 241
...
```

Representing an image in your program

```
P3
300 168
255
245 246 241
245 246 241
245 246 241
...
```

- 1-D list of pixels

Representing an image in your program

```
P3
300 168
255
245 246 241
245 246 241
245 246 241
...
```

- 1-D list of pixels
- 2-D list (grid) of pixels

Representing a pixel in your program

```
P3
300 168
255
245 246 241
245 246 241
245 246 241
...
```

Representing a pixel in your program

```
P3
300 168
255
245 246 241
245 246 241
245 246 241
...
```

- a list of three ints.

Representing a pixel in your program

```
P3
300 168
255
245 246 241
245 246 241
245 246 241
...
```

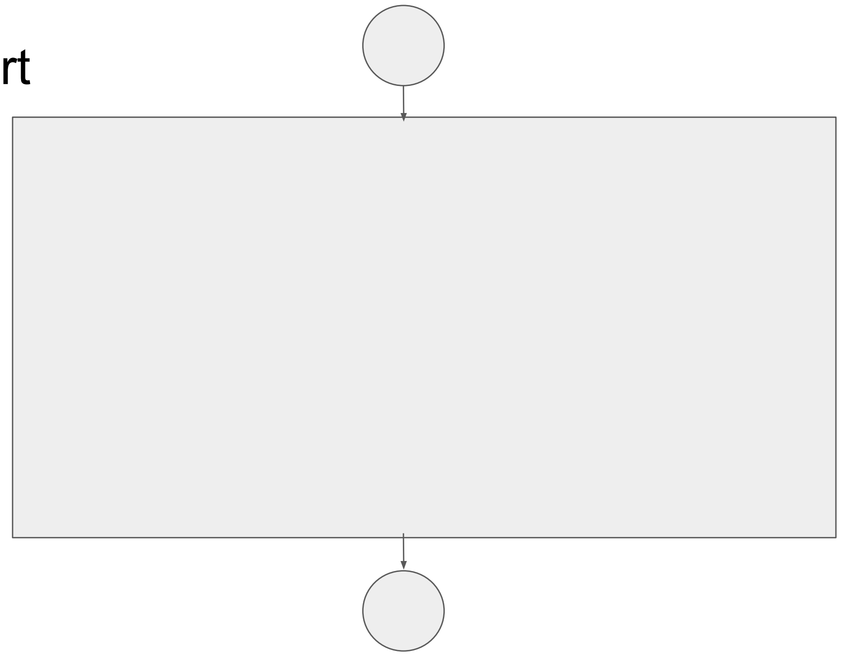
- a list of three ints.
- a tuple of three ints.

Representing a pixel in your program

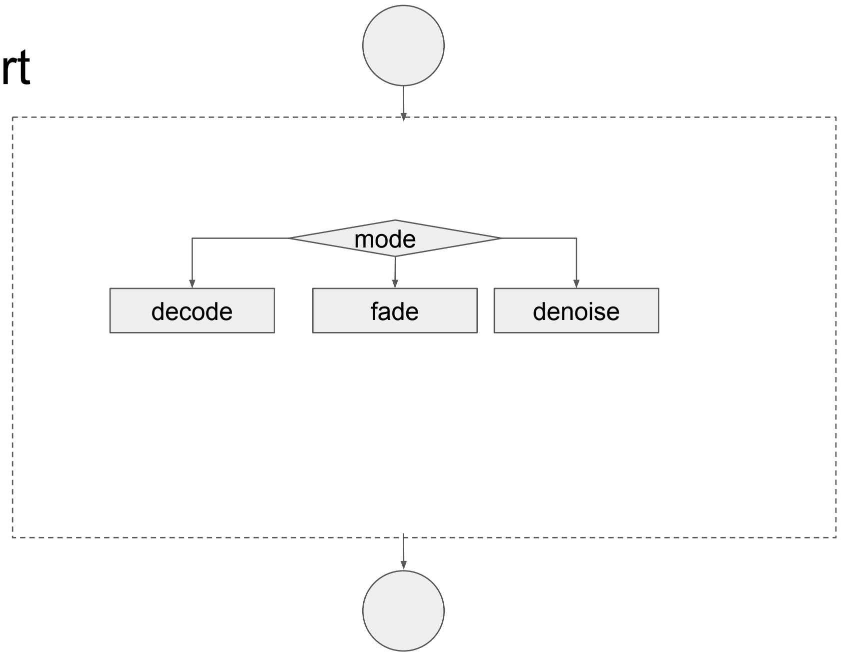
```
P3
300 168
255
245 246 241
245 246 241
245 246 241
...
```

- a list of three ints.
- a tuple of three ints.
- an object with three int attributes.

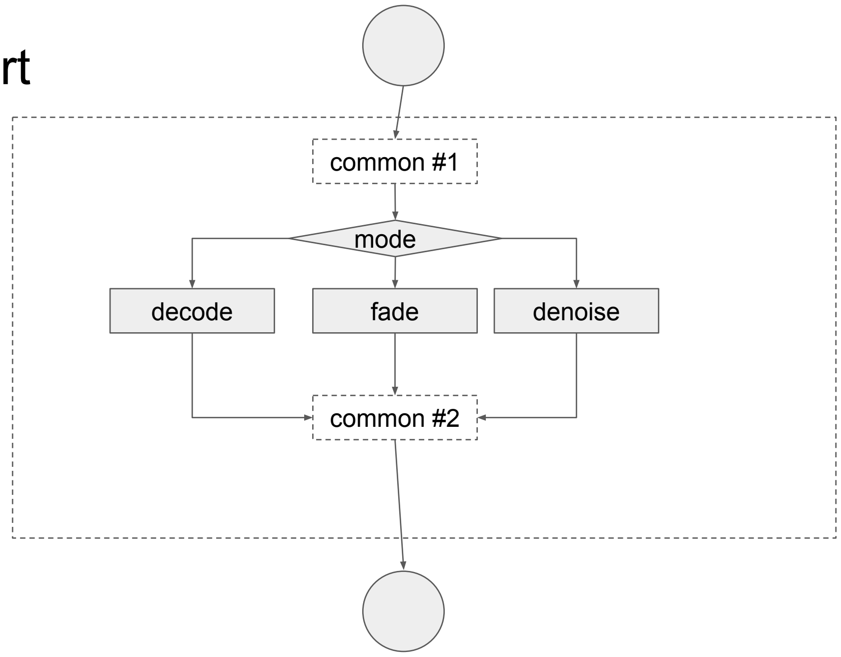
Flow chart



Flow chart



Flow chart



Step 2: Signature, Purpose Statement, Header

- Function Header
  - Choose meaningful function name and argument names.
  - Do not write the function body yet (Write a dummy for the body: pass).

Step 2: Signature, Purpose Statement, Header

- Function Header
  - Choose meaningful function name and argument names.
  - Do not write the function body yet (Write a dummy for the body).
- Signature describes inputs and outputs of your function
  - describes what kind of values the function accepts, and what kind of value it returns.

Step 2: Signature, Purpose Statement, Header

- Function Header
  - Choose meaningful function name and argument names.
  - Do not write the function body yet (Write a dummy for the body).
- Signature describes inputs and outputs of your function
  - describes what kind of values the function accepts, and what kind of value it returns.
- Purpose statement describes the purpose of the function
  - help other programmers understand what the function is supposed to do.
  - Remind yourself what your function is supposed to do.

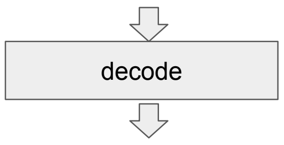
Step 3: Test cases

- Test cases are functional examples of your solution.
- Identify all kinds of cases your function needs to handle.
  - General cases and Edge cases (Base,Error, and other special cases)
  - If you realize that not all cases can not be solved by your solution, go back to the step #2 or the step #1.

Step 3: Test Cases

- For testing functions for Pixel Magic, create a small image data.
  - For example, create data for 4 pixels representing a 2 by 2 image.

(12, 13, 14), (15, 16, 17), (18, 19, 20), (26, 25, 24) *This is not Python code.



(120, 120, 120), (150, 150, 150), (180, 180, 180), (255, 255, 255)

Step 4: Template

Write pseudo code

- Add placeholders in the function body
    - Loop
    - Branches
        - Special cases and typical cases
    - Variables
        - Fields in objects
    - Helper function

Step 5: Write the function body

- Replace pseudo code with actual code.
- Write clean and concise code by following a coding style.
    - Your code needs to be readable so that it is easy to understand, change, and extend.

Step 6: Test

- Test your function using the test you wrote in the step #3.
- Go back and fix your code if the test fails.
- Sometimes, you have to fix your tests as well.

Summary

Step 1: Data Definitions

Step 2: Signature, Purpose Statement, Function Header

Step 3: Test Cases - Write functional examples

Step 4: Template - Write pseudo code with placeholders for values

Step 5: Write the body of the function

Step 6: Test