CS2030 Lab 2

Cruise Loaders

Qi Ji Shuming

2nd September 2019

Lab 1 Recap

- Get used to connecting to plab server (using WinSSHTerm)
- Get used to gvim commands, etc.
- Comments are up on codecrunch

Common Mistakes

Styling

- Generally ok.
- Try to make it readable (we are reading the code too)
- Use gg=G command (Demo)

Lab 2 Brief

Topic Coverage

- Inheritance
- Polymorphism
- Method overriding

Task

Write a program that reads in the number of cruises in the schedule as an integer, and a list of cruises that will arrive for that day.

The program will output the loader allocation schedule.

- 2 types of cruises
  - normal
  - big
- 2 types of loaders
  - normal
  - recycled (less useful)

Levels

1. Representing a normal cruise
2. Representing a normal loader
    - Use loaders to serve cruises
3. Introduce Big Cruises
    - Inheritance and polymorphism.
4. Introduce Recycled Loaders
    - See above
5. Finishing up
    - Same as Lab 1 Level 5

Level 1: Represent a Cruise

Cruises:

- Takes a fixed 30min for a loader to fully load.
- Requires only one loader for it to be fully served.

Level 1: Things to do

- Constructor that takes in `String id` and `int arrivalTime`
- Instance method `getServiceCompletionTime()`: 30 minutes after `arrivalTime`
  - Cases to consider
  - $130 + 30 = 200$
  - $145 + 30 = 215$
  - Hint: Abstract into a private function
- Overriden `toString()` method (next page)

Level 1: Output format

- `cruiseCode@arrivalTime` string representation of a cruise
  - `arrivalTime` must be formatted as a 24h time
- Hint: Use `String.format("%04d", arrivalTime)`
  - E.g. `String.format("04d", 23)` returns `"0023"`

Level 2: Represent a Loader

Loaders:

- As many loaders are used to serve a cruise as required.
- Cannot serve another cruise until the current service is done.
- Output a list of cruises it has served.

Level 2: Things to do

- Constructor that takes in `int id`
  - Hint: Consider using a `static int` as counter.
- Instance method `serve(Cruise cruise)`: serves a cruise
  - Does the loader need to know what cruises it has served?
  - Does the loader need to know when is the next available time?
- Overriden `toString()` method (next page)

Level 2: Output format

```
Loader id serves:
    cruise1@hhmm
    cruise2@hhmm
    ...
```

Level 3: Big cruises

- Needs X loaders $0 \leq X \leq 9$
- Needs T service time. $0 \leq T \leq 99$
- Use inheritance
    - What new information is there?

Level 4: Recycled loaders

- Takes a 60 min break **after** each service.
- Every third loader is recycled.
    - Loader 1
    - Loader 2
    - Loader 3 (recycled)
    - Loader 4
    - ...

Level 4: Things to do

- Inheritance again
  - What's new?
- Override `serve(Cruise)`: the next available timing is the cruise's service time + 60 minutes

Level 5: Putting it together

- Start with no loaders
- For each cruise, find the first available loader
- If there are not enough loaders, purchase a new loader
    - Every 3rd loader is recycled