

**28-0: Program Design**

- You are writing a large program
  - Example: Next project is Mastermind
- What do you do first?

**28-1: Program Design**

- You are writing a large program
  - Example: Next project is Mastermind
- What do you do first?
  - You do *not* open up Eclipse and start writing!

**28-2: Program Design**

- First thing you need to do is design your program
  - What classes should your program have?
  - What data members & methods are in each class?
  - How should the classes communicate with each other?

**28-3: Program Design**

- How to decide on classes and methods?
  - Describe the project in a short English paragraph
  - Nouns == classes
  - Verbs == methods
  - Iterate, iterate, iterate!

**28-4: Program Design**

- Example: Mastermind!
  - What are some classes that you might need?
  - What methods for each class?

**28-5: Program Design**

- Example: Mastermind!
  - Classes:
    - Code
    - Guess
    - Response
    - Color (enumerated type)
    - Mastermind (main program)
  - More than one way do to this, of course!

**28-6: Program Design**

- Example: Mastermind!
  - Response class
    - Pretty simple
    - What should the methods be?
    - What instance variables do you need?

**28-7: Program Design**

- Response class
  - Methods:
    - Constructor to create response, # of black/white pegs
    - Get the number of black/white pegs
    - (set # of black/white pegs)

**28-8: Program Design**

- Example: Mastermind!
  - Code class (representing the hidden code)
    - What should the methods be?

**28-9: Program Design**

- Code class (representing the hidden code)
  - Methods
    - Create a new hidden code
    - Randomize hidden code
    - Given a guess, give a response

**28-10: Enumerated types**

- Every enumerated type has a static values() method
- Returns an array of all legal values for that type

```
enum Times {MORNING, NOON, NIGHT}
Times validTimes = Times.values();
```

- Handy for creating a random value ...

**28-11: Design Issues**

- When should a variable be an instance variable. When should a variable be a local variable?

**28-12: Design Issues**

- When should a variable be an instance variable. When should a variable be a local variable?
  - If a variable is only used in a single method, it should probably be a local variable

- If you need to keep track of the value of a variable across method calls, it should probably be an instance variable
- Temporary variables should not be instance variables (most of the time)

**28-13: Design Issues**

- How many different actions should each method try to accomplish?

**28-14: Design Issues**

- How many different actions should each method try to accomplish?
  - Each method should just do one thing
  - That “one thing” can be a high-level, complicated task
    - Call other methods to handle the details

**28-15: Design Issues**

- What should you do if you find yourself cutting and pasting code?

**28-16: Design Issues**

- What should you do if you find yourself cutting and pasting code?
  - Cut-and-paste is usually bad. (why?)
  - Write code once, re-use it
  - May need to rewrite code a little to be more general ...

**28-17: Design Issues**

- What should you do if you find yourself typing an arbitrary constant (like 25, 4, or 17) in the body of a method?

**28-18: Design Issues**

- What should you do if you find yourself typing an arbitrary constant (like 25, 4, or 17) in the body of a method?
  - Define a constant instead!

**28-19: Design Issues**

- How can a method return > 1 different element?

**28-20: Design Issues**

- How can a method return > 1 different element?
  - Create a class that holds all the elements you want to return
  - As much as possible, this should be a nice logical unit
    - Guess, SecretCode, Point, etc.

**28-21: Design Issues**

- When should you call “new”?

**28-22: Design Issues**

- When should you call “new”?
  - When you want to create a new piece of memory on the heap
  - Usually do *not* want to call new on temporary pointers used to traverse linked structures
  - Remember, new does not create a new pointer, it creates a new *Object*.