SWE 265P
Reverse Engineering and Modeling

Lecture 3: Mental Models and Mental Simulation

Reality

"My personal strategy is to formulate hypotheses based on heuristics [largely names of things] and then test those hypotheses with print statements." – Crista Lopes [Professor, UC Irvine]

Reality

“Then, I […] and draw things out. Important classes, what functions are called to perform different features. Usually on a piece of paper.” – Kristina Nasr [Software engineer, Google]

Reality

"Often I use a debugger to understand information flow and a diagramming tool to help me build a mental model of the system." – Lee Martie [Research staff member and Advanced Prototyping Team Manager, MIT-IBM Watson AI Lab]

Today

- Last week's material
- Mental models
- Mental simulation
- In-class practice
- Key expert practices

Last week's material

- Information foraging
  - top-down, bottom-up, systematic, opportunistic comprehension
  - goal-driven
  - familiarity
- JPacMan
  - "understanding"
  - changes!
- Homework
- Any questions?

An example

```
class A{static char a=0,b=a++,e=a++,f=(char)(a/a);static char p(String s){return(char)Byte.parseByte(s,a);}public static void main(String[]z){long x=e,y=b;String c=((Long)x).toString(),d=((Long)y).toString();char l=p(c+c+d+c+c+d+d),m=p(c+c+d+d+c+d+c),o=(char)(l+a+f);b=p(c+d+d+d+d+d);e=b++;System.out.print(new char[]{p(c+d+d+c+d+d+d),m,l,l,o,e,p(c+d+c+d+c+c+c),o,(char)(o+a+f),l,(char)(m-f),b});}}
```

An example

```
class A
{

   //initializing some constants needed
   static char a = 0, b = a++, e = a++, f = (char) (a / a);

   //shorthand for parseByte() (codegolfing handyness)
   static char p(String s)
   {
      return (char) Byte.parseByte(s, a);
   }

   public static void main(String[] z)
   {
      long x = e, y = b;
      String c = ((Long) x).toString(), d = ((Long) y).toString();
      char l = p(c + c + d + c + c + d + d),
      m = p(c + c + d + d + c + d + c),
      o = (char) (l + a + f);
      b = p(c + d + d + d + d + d);
      e = b++;
      System.out.print(new char[]
      {
         p(c + d + d + c + d + d + d)
         ,
         m, l, l, o, e,
         p(c + d + c + d + c + c + c),
         o, (char) (o + a + f),
         l,
         (char) (m - f), b
      });
   }
}
```

Familiar techniques

- Print statements
- Assertions
- Debugger

https://github.com/SWE-265P/obfuscated

Mental model

- An explanation of someone's thought process about how something works in the real world
- A representation of the surrounding world, the relationships between its various parts and a person's intuitive perception about his or her own acts and their consequences
- Can help shape behavior and set an approach to solving problems and doing tasks

Properties of mental models

- Individual
- Uncertain
- Selective
- Malleable
- Dependent

Mental model (software, external)

- Mental models are an artifact of belief
  - users will plan and predict their future actions with a system based on their mental models [as constructed from the visible interface of the software, any documentation of what it does, and other means]

- Designers ideally should anticipate users' mental models so that their product communicates its function through its form

Mental model (software, internal)

- Mental models are an artifact of belief
  - developers will plan and predict their future actions with a system based on their mental models [as constructed from the source code, any documentation of how it works, and other means]

- Developers ideally should anticipate other developers' mental models so that their source code communicates its function through its form

Properties of mental models

- Individual
- Uncertain
- Selective
- Malleable
- Dependent

Mental simulation

The process of self-projection into alternate temporal, spatial, social, or hypothetical reality

Mental simulation is our mind's ability to imagine taking a specific action and simulating the probable result before acting

An example

```
public class HelloWorld {
    public static void main(String[] args) {
        // Prints "Hello, World"
        System.out.println("Hello, Word");
    }
}
```

An example

```
class A
{

   //initializing some constants needed
   static char a = 0, b = a++, e = a++, f = (char) (a / a);

   //shorthand for parseByte() (codegolfing handyness)
   static char p(String s)
   {
      return (char) Byte.parseByte(s, a);
   }

   public static void main(String[] z)
   {
      long x = e, y = b; //needed for some weird reason to save bytes
      String c = ((Long) x).toString(), d = ((Long) y).toString(); //creating a 1 and a 0 a string
      char l = p(c + c + d + c + c + d + d),  //binary digit voodoo
      m = p(c + c + d + d + c + d + c),  //more commonly used letters prepared
      o = (char) (l + a + f);
      b = p(c + d + d + d + d + d);
      e = b++;
      System.out.print(new char[]  //assembling the string with arithmetic and more binary stuff
      {
         p(c + d + d + c + d + d + d)
         ,
         m, l, l, o, e,
         p(c + d + c + d + c + c + c),
         o, (char) (o + a + f),
         l,
         (char) (m - f), b
      });
   }
}
```

Limitations of our mind

- Our mind is limited in capacity
  - short term memory
- Our mind is prone to forgetting aspects
  - long term memory
  - medium term memory
- Our mind cannot be accessed by others

Let's practice: JPacMan3

- You should still have a clone of JPacMan3, but if not
  - https://github.com/SWE-265P/jpacman3

- Open the project

JPacMan question 1 (locate a feature)

- Draw a model of where scoring is implemented
- https://jamboard.google.com/d/1qsCVHTYCvvy0RO0SfEyejmdQ6sH_t1P2M6OZ111A0VA/edit?usp=sharing

Break

JPacMan question 2 (understand a feature)

- Draw a model of how scoring works

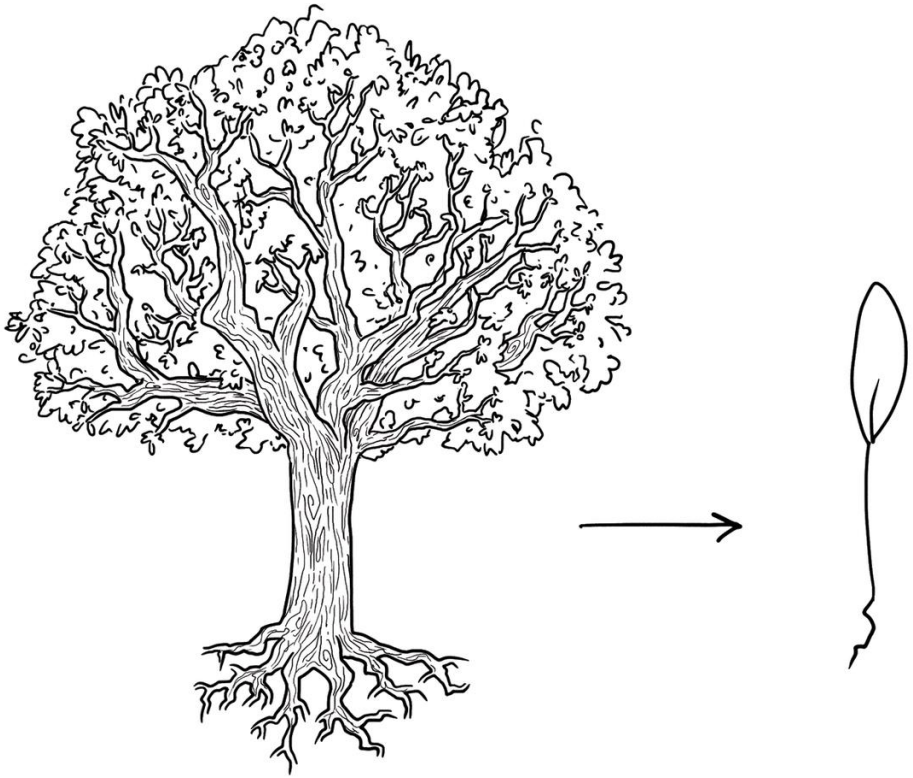
- https://jamboard.google.com/d/1CaP45xoNTX02JCVqlNNxG9GI41cjtE4UySKaThwGmJo/edit?usp=sharing

JPacMan question #3 (understand a feature)

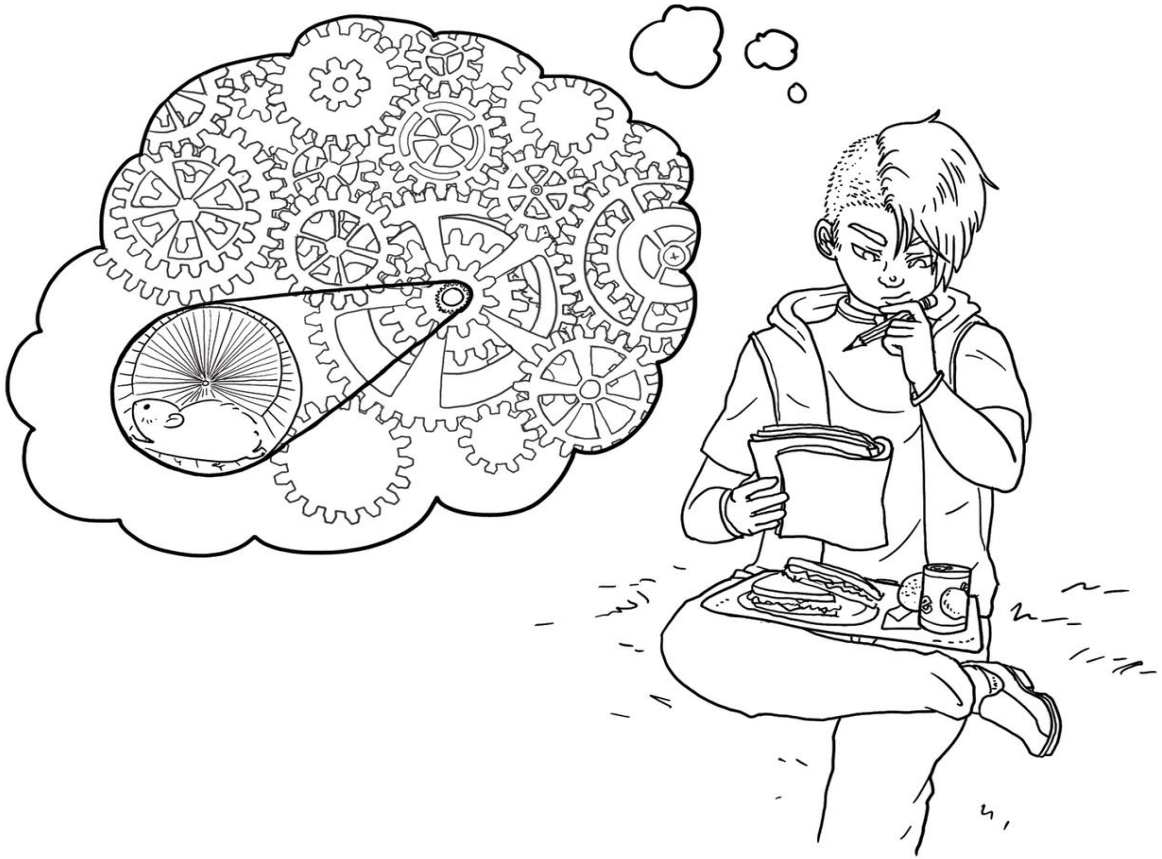
- Draw a model of how collisions work
- https://jamboard.google.com/d/1a5fygWcO5LMuk7HCxUfAewRpcQN2Vzg3QR7IMSP1S20/edit?usp=sharing

Key expert practices

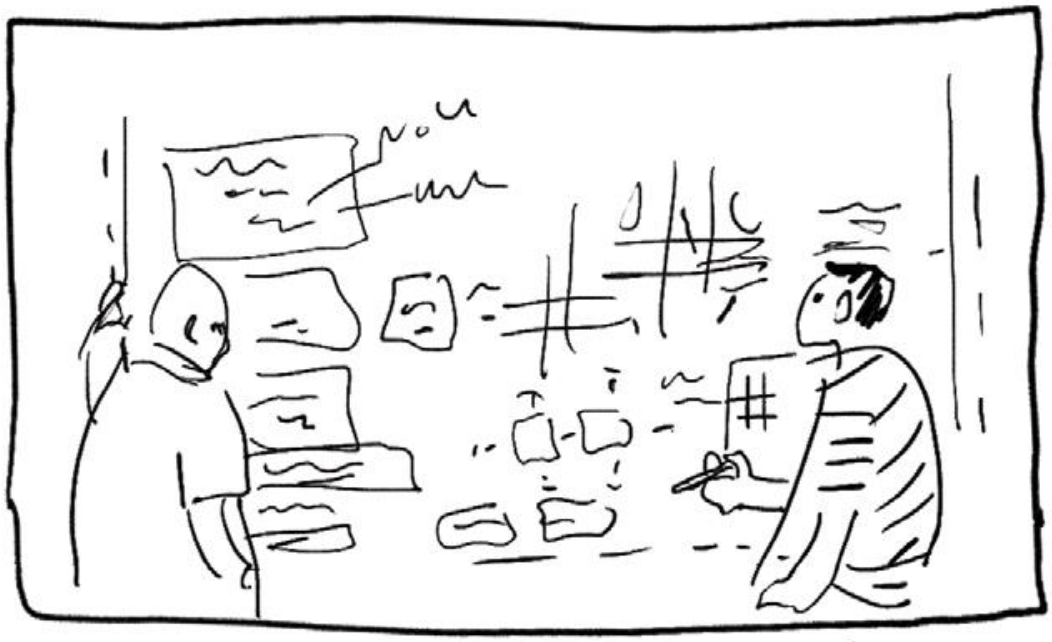
KEP #1: focus on the essence



KEP #2: simulate continually



KEP #3: work with others



Project work, part 1

- Without looking at the code and with your team, decide upon a small feature that you want to understand
- Submit via Canvas as a single PDF
- Due: Thursday @ 4pm

Project work, part 2

- Individually, draw a model of where that feature is located and draw a model of how that feature works
- Submit via Canvas as a single PDF
- Due: Sunday @ 4pm

Project work, part 3

- With your team, draw a model of where that feature is located and draw a model of how that feature works
- Submit via Canvas as a single PDF
- Due: Tuesday @ 4pm

Homework (individual)

- Tutorial: reading code
  - https://www.youtube.com/watch?v=cPVu9AJ8gGw&t=523s
- How to read code
  - https://www.youtube.com/watch?v=-KgU5sxGtuM
- Strategies for working with legacy code
  - https://www.youtube.com/watch?v=UH4dSpPieDE

Homework (individual)

- Make sure to regularly update your personal diary