Problem Solving

S1A

* Why do we need Problem Solving
* 6 Approaches to Solving Problems
* Software Engineering Phases
* Vocabulary

Why do we need Problem Solving Skills?

- Programming Computers effectively is largely dependent upon our ability to solve problems

- Programs essentially set out to resolve some problem

- Essentially we want to come up with a specific & efficient set of steps that can be repeated to solve our problem
  - This set of steps is called an **algorithm**
  - For example, if we want to find out the largest of 3 numbers → what is the process we employ to achieve this result

- Once we come up with our algorithms then we convert it to code so it can solve that problem repeatedly
  - Programs are usually consist of many algorithms

NOTE: We often have to solve problems that have not been solved before.

Problem Solving Process

1 – Analyze the problem

2 – Solve & Define the algorithm

3 – Test the algorithm

4 – Refine the algorithm

Problem Solving Process Explained

1- Analyze the problem
  - Try to gain a greater understanding of it
  - Break the problem down into smaller parts
    - AKA Divide and Conquer → big problems can be overwhelming!
  - use one of the methods we'll discuss later

2 – Solve & Define the algorithm
  - If can solve it once, then we can start to figure out what steps we employed.
  - Define these so we can solve it again.

Problem Solving Process Explained (2)

3 – Test the algorithm
  - Verify that it works
  - Follow your steps and make sure they solve the problem

4 - Refine the steps
  - Try to find a more efficient/effective way of solving the problem
  - Make sure you verify again

This all takes practice → and some trial and error

The key to innovation is not giving up
If one approach doesn't work then you've still made progress
→ you can eliminate that approach

Analyzing and Initial Solution

- Determine what the problem is
  - Try to come up with one possible solution

- This is where we will start in this class

- We will use a variety of techniques to analyze and solve problems:
  - Build upon what you know
  - Analyzing the problem state
  - Thinking outside the box
  - Look for similarities in previously solved problems
  - Means-Ends Analysis
  - Divide and Conquer

Build upon what you know

- Often in software development you will have a seemingly disjoint set of requirements
- One approach is to
  Build upon your current knowledge
- Analyze each fact and determine if you can deduce anything based on those facts that will lead towards a solution
  - Try to solve in some systematic order
  - Take one requirement at a time

Favorite Television Show

- Six friends are seated around a table discussing their favorite TV show
- The shows they are discussing are
  - The Walking Dead
  - Breaking Bad
  - Big Bang Theory
  - Criminal Minds
  - Parks & Recreation
  - Game of Thrones
- Based on the following information
  - Determine each person's favorite TV Show and where they are sitting

(2) A) The boy who likes "Breaking Bad" sits directly to the right of Sean
(1) B) Scott sits between two people, one watches "Game of Thrones" and the other one watches "Breaking Bad"
(4) C) No one sits between Trish and Danielle
(5) D) Danielle sits directly to the left of the boy who likes "The Walking Dead"
(6) E) The boy who likes "Big Bang Theory" does not sit next to a girl
(7) F) Danielle does not care for "Parks & Recreation"
(8) G) Carlos does not sit next to a girl
(3) H) Erik does not sit next to the girl who likes "Game of Thrones"

Name _____ Scott _____
Fav Show _____

Think about what we know → Which of the clues build upon our current knowledge?

Summary → build upon what you know

- The purpose of this drill is to start with one clue
  - systematically address each requirement
- Most problems in CS consist of many requirements
- It is important to address each requirement without violating another requirment

Analyzing & Solving Techniques

- Build upon what you know
- Analyzing the problem state
- Thinking outside the box
- Look for similarities in previously solved problems
- Means-Ends Analysis
- Divide and Conquer

Analyzing the Problem State

- Sometimes we can list each possible state.
  - analyze each possibility individually.
- The problem can be solved through the process of elimination
  - In other words --- which possibilities can be eliminated and which ones can't
- … or through a series of deductions

The Five Hats Problem

- Three men are condemned to die, one of whom is blind. The king decides he will offer them an opportunity to be set free.
- The three men are arranged in a circle facing one another.
- The king produces 5 hats: 3 white and 2 black.
- The men can't see the hats
- The king places a hat on the head of each person and then destroys the two remaining hats.

- The men have no idea which hats have been destroyed.
- The king instructs them, "The first one of you who can tell me the color of his hat will be set free."
- A period of time passes in silence and then finally the blind man tells the king the color of his hat and is set free.
- What color hat was the blind man wearing and how did he know?
- This does not have a "trick" answer – your answer should be very logical and well thought out. Be able to explain your answer from the viewpoint of each of the three prisoners.
- **Hint: Each sighted man can see the blind man's hat as well as that of the other sighted man. What does the pause in time infer?**
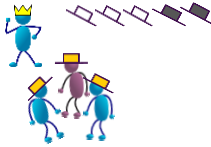
(c) Michele Rousseau   Problem Solving   16

Five Hats Problem

- For problems like this we know each of the possible outcomes

- Write them down

- Then try to analyze each possibility
  - Eliminate possibilities if you can

(c) Michele Rousseau   Problem Solving   17

Think about all the possibilities.

What are the potential arrangements of hats?

See1   See2   Bl

(c) Michele Rousseau   Problem Solving   18

Analyzing & Solving Techniques

- Build upon what you know
- Analyzing the problem state (continued)
- Thinking outside the box
- Look for similarities in previously solved problems
- Means-Ends Analysis
- Divide and Conquer

(c) Michele Rousseau   Problem Solving   19

Analyzing the Problem State

- The 5 hats / 3 prisoners problem is an example of analyzing the possible states
- Using this method
  - Identify that there is a limited number of choices
  - List out the choices and analyze them
  - Sometimes it is trial and error → this is okay for limited choices
- For the 5 hats → _____
  _____
- We examined all the possible states for all three prisoners (there were only 7)
- From these we deduced _____
  _____
  → otherwise the seeing prisoners would have spoken up
- Let's expand upon this technique to solve the next problem

(c) Michele Rousseau   Problem Solving   20

Missionary-Cannibal Problem

- There are 3 missionaries, 3 cannibals, and a boat.
- They want to cross the river.
- If there are more cannibals than missionaries on any side of the river the cannibals will eat the missionaries.
- Only 2 people fit in the boat at one time.
- How do the missionaries get the cannibals across the river without getting eaten. (both Missionaries and cannibals need to get to the other side)
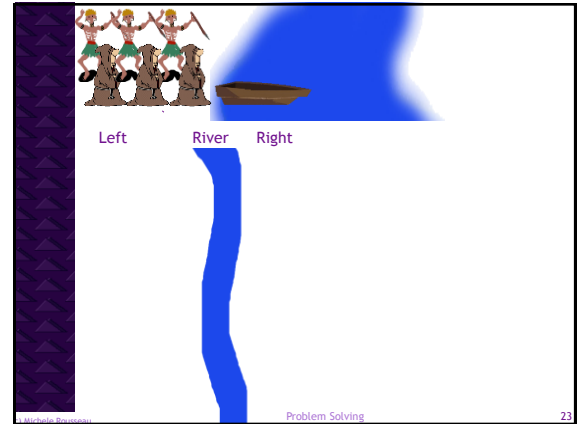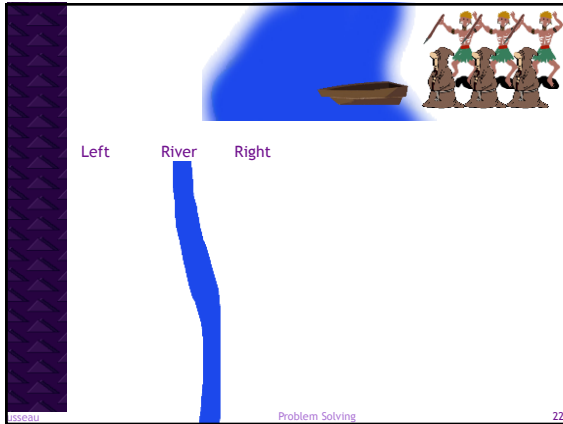
Looking at the possibilities → just like the 5 hats
Make deductions at each step

(c) Michele Rousseau   Problem Solving   21

Left   River   Right

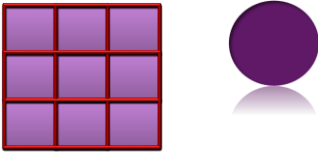Left   River   Right

Analyzing & Solving Techniques

- Build upon what you know
- Analyzing the problem states
- Thinking outside the box
- Look for similarities in previously solved problems
- Means-Ends Analysis
- Divide and Conquer
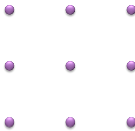
Thinking Outside the Box

- How many boxes are there?

Sometimes we put constraints on problems
→We need to look at the constraints and determine if the problem dictates those constraints or if we do
→ If it is us then we need to think beyond our preconceptions

Thinking Out of the Box (2)

Draw four line segments through the 9 points without lifting your pencil or retracing a line.

Analyzing & Solving Techniques

- Build upon what you know
- Analyzing the problem states
- Thinking outside the box
- Look for similarities in previously solved problems
- Means-Ends Analysis
- Divide and Conquer

Try to look for similarities

Look for similarities in other problems you have solved

For Example

Finding the heaviest & lightest weight

is really the same problem as

Finding the highest and lowest grades on a test

is really the same problem as

Finding the daily high and low temperatures

all 3 problems can be abstracted as

_____

Analyzing & Solving Techniques

- Build upon what you know
- Analyzing the problem states
- Thinking outside the box
- Look for similarities in previously solved problems
- Means-Ends Analysis
- Divide and Conquer

Means-Ends Analysis

- Beginning state and End state are often given
  - You need to define a set of actions that can be used to get from one to the other
  - Once you have a set of actions, you need to work out the details

Translated to computer programming

- Begin by writing down the inputs.
  - (Beginning state)
- What should the output be for those inputs?
  - (End state)
- What processing need to be performed to obtain the desired results?

Analyzing & Solving Techniques

- Build upon what you know
- Analyzing the problem states
- Thinking outside the box
- Look for similarities in previously solved problems
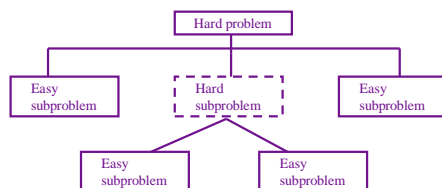- Means-Ends Analysis
- Divide and Conquer

Divide & Conquer

- Break up large problems into smaller manageable problems

```
              Hard problem
         _____|_____|_____
        |           |             |
     Easy         Hard          Easy
   subproblem   subproblem    subproblem
        |_____|
        |           |
     Easy         Easy
   subproblem   subproblem
```

Divide & Conquer Example

Compute the area of a circle

Problem statement

- We need an interactive program (user will input data) that computes the area of a circle. Given the circle radius, the circle area should be displayed on the screen

Input/Output description

- Input → Circle radius
- Output → Circle area

Algorithm development (set of steps, decomposition outline)

1. Read value of circle radius (r)
2. Compute circle area as $\pi* r^2$
3. Print the value of circle area
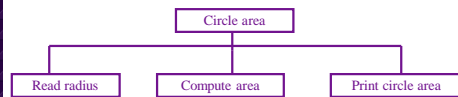
How do we represent more complex algorithms

- Pseudocode, flowcharts

Divide & Conquer Example (2)

A divide and conquer block diagram of our problem

```
                Circle area
       ┌────────────┼────────────┐
   Read radius   Compute area   Print circle area
```

Pseudocode

BEGIN

PROMPT the user for the circle radius

READ radius

CALCULATE Circle area (π* radius^2)

OUTPUT Circle area

END

---

5 Basic Software Development Phases

1. Requirements Phase
→ Analyze and specify the problem
→ Define the problem → inputs and outputs

2. Design Phase
→ Determine how the problem will be approach and solved (Problem Solving)
→ Verify your solution solves the problem specified

> In this class we will mostly focus on small scale Design and Implementation Phases

---

Software Development Phases (2)

3. Implementation Phase
→ Code & document the program (Concrete solution)
→ Basic testing

4. Testing
→ More formalize testing → make sure it meets the specifications

5. Maintenance
→ Use the Program
→ Modify (meet changing requirements)
→ Fix bugs missed in implementation

Going from Problem Solving to Programming - Vocabulary

o Algorithm
  → a step by step process for solving a problem.
o Top-Down Design (design methodology)
  → break a larger problem into small parts progressing from the general to the specific
  → The smaller parts are more manageable and easier to understand
  → AKA "divide & conquer"
o Hierarchical Input/Output "HIPO" Chart
  → A diagram of the top down design
  → Provides a hierarchical perspective of the systems input, output, and processing modules
o Module
  → One small part of the solution

Vocabulary (2)

o Flowchart
  → A diagram of an algorithm using specific symbols that represent programming constructs
o Pseudocode
  → A terse, English-like description of an algorithm
  → Used to understand the basic program flow without worrying about the correct syntax
o Desk Check
  → Walking through the algorithm manually – step by step
  → Draw each memory location and check the algorithm as though you were the computer
o Documentation
  → Anything that provides information about a program
  → Comments in the code, data tables that describe the data used in the code & external documents (flow charts, user's manual, the design, &etc)