OBJECTIONS TO THE CHINESE ROOM ARGUMENT

TWO OBJECTIONS

1. The 'Systems' Objection
2. The 'Implementation' Objection

THE 'SYSTEMS' OBJECTION

- Perhaps the person *in* the Chinese room does not understand Chinese but the *Chinese Room itself* understands Chinese
- Since the Chinese room is the proper analogue to the computer program, and not the person *in* the Chinese room, Searle's example proves nothing

SEARLE'S REPLY

> *My response to the systems theory is quite simple: let the individual internalize all of these elements of the system. He memorizes the rules in the ledger and the data banks of Chinese symbols, and he does all the calculations in his head. The individual then incorporates the entire system. There isn't anything at all to the system that he does not encompass. We can even get rid of the room and suppose he works outdoors. All the same, he understands nothing of the Chinese, and a fortiori neither does the system, because there isn't anything in the system that isn't in him. If he doesn't understand, then there is no way that the system could understand because the system is just a part of him.*

PRYOR'S REBUTTAL (I) SEARLE'S ARGUMENT IS INVALID

> *Searle: "[The man in the room] understands nothing of the Chinese, and [therefore] neither does the system, because there isn't anything in the system that isn't in him"*

- This is a bad inference—compare:

> *Searle doesn't weigh 3 pounds, and therefore neither does his heart, because there is nothing in his heart that isn't in him*

- the form of inference Searle uses here doesn't generalize to other inferences with the same kind of form
  - leaves open the possibility that the particular argument Searle makes here is true

PRYOR'S REBUTTAL (II) INTERNALIZATION IS IRRELEVANT

> *Searle: "If he doesn't understand, then there is no way that the system could understand because the system is just a part of him."*

- Consider a software emulator
  - allows one operating system to run 'on top of' another using the same hardware
    - Mac computers can emulate the Windows OS

- Assume a Mac runnning its OS *plus* an emulation of Windows OS

    1. The Windows OS is integrated or incorporated into the Mac OS
    2. Nevertheless, the states of the 'incorporated' Windows OS are in many ways independent of the Mac OS and its states
        - Windows may crash and become unresponsive, while the Mac software (including the emulator) keeps running
        - Windows might be treating Internet Explorer as the frontmost, active program; but–if you don't have the emulator software active in your Mac–the Mac software could be treating Safari as its frontmost, active program

> *when Jack memorizes all the instructions in the Chinese book, he becomes like the Mac software, and the Chinese room software becomes like the emulated Windows software. Jack fully incorporates the Chinese room software. That does not mean that Jack shares all the states of the Chinese room software, nor that it shares all of his states. If the Chinese room software crashes, Jack may keep going fine. If the Chinese room software is in a state of believing that China was at its cultural peak during the Han dynasty, that does not mean that Jack is also in that state. And so on. In particular, for the Chinese room software to understand some Chinese symbol, it is not required that Jack also understand that symbol.*

- Problem 2: 'Internalizing' the Chinese room program is irrelevant
  - two programs running on the same hardware need not share all of the same (or any of the same) states

SUMMARY OF PRYOR'S REBUTTALS:

1. Searle's argument is invalid
    - the form of inference Searle uses here doesn't generalize to other inferences with the same kind of form in a way that preserves truth
2. 'Internalization' is irrelevant
    - two programs running on the same hardware need not share all of the same (or any of the same) states

THE IMPLEMENTATION OBJECTION

PROGRAMS VS. IMPLEMENTATIONS

> 1. *Programs are purely formal (syntactic)*
> 2. *Human minds have mental contents (semantics)*
> 3. *Syntax by itself is neither constitutive of, nor sufficient for, semantic content*
> 4. *∴ Programs by themselves are not constitutive of nor sufficient for minds*

- We need to distinguish between a *program* and an *implementation of the program*

> *Programs are abstract computational objects and are purely syntactic. Certainly, no mere program is a candidate for possession of a mind. Implementations of programs, on the other hand, are concrete systems with causal dynamics, and are not purely syntactic. An implementation has causal heft in the real world, and it is in virtue of this causal heft that consciousness and intentionality arise. It is the program that is syntactic; it is the implementation that has semantic content. (Chalmers 1996, 327)*

CHALMERS'S PARODY:

1. Recipes are syntactic.
2. Syntax is not sufficient for crumbliness.
3. Cakes are crumbly.
4. ∴ Implementing a recipe is insufficient for a cake.

> *A recipe implicitly specifies a class of physical systems that qualify as implementations of the recipe, and it is these systems that have such features as crumbliness. Similarly, a program implicitly specifies a class of physical systems that qualify as implementations of the program, and it is these systems that give rise to such features as minds. (Chalmers, 327)*