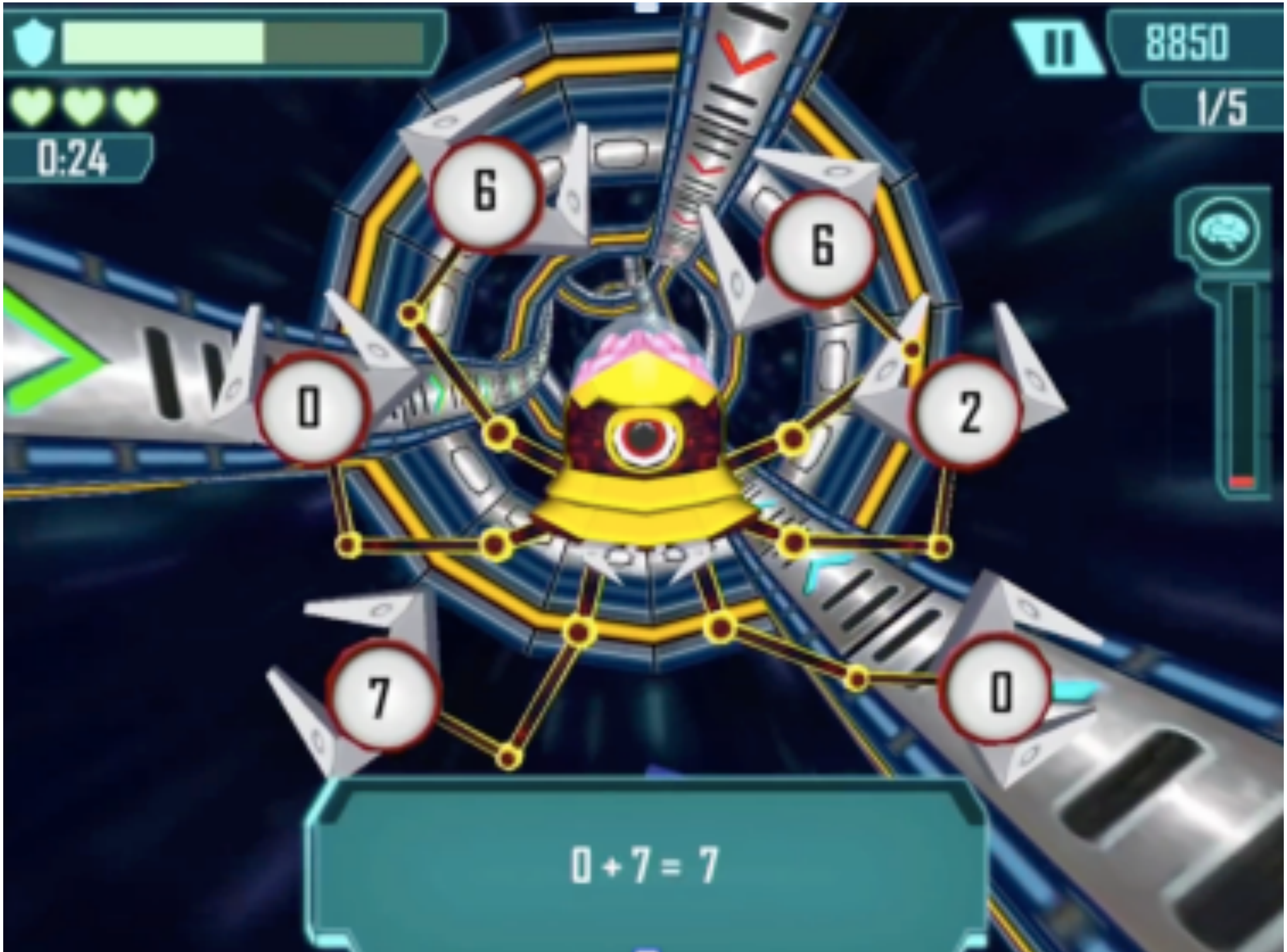
COSC 442: Mobile Educational Game Development

Dr. Bowen Hui

University of British Columbia Okanagan

Math Blaster



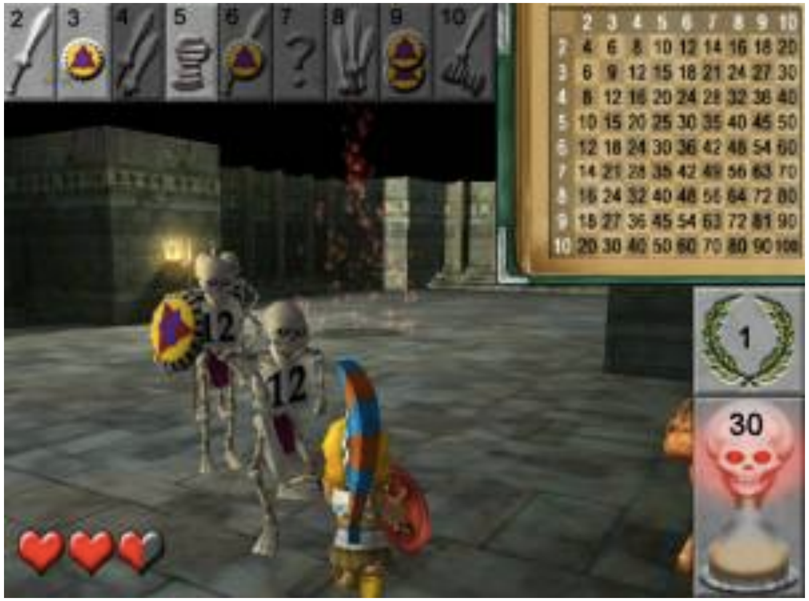
0 + 7 = 7

Math Blaster

- Academic subject: arithmetic
- Exercises: Repetitive drilling
- Interleaves drills with engaging gameplay
- Heavily criticized as "chocolate-covered broccoli"
- What about seamless interweaving of academic subject matter with gameplay?

Zombie Division

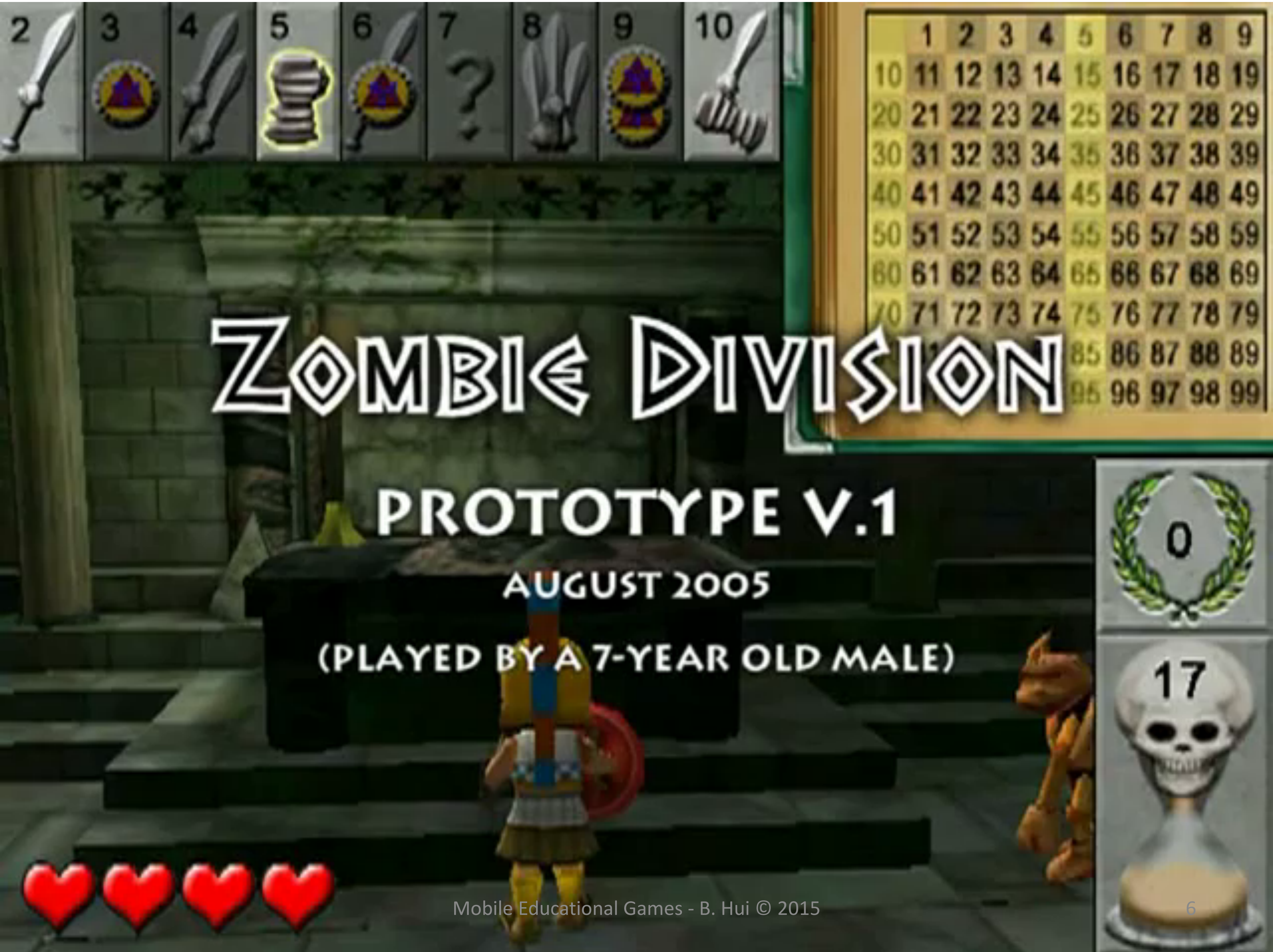
- Ancient hero faces skeletons (number)
- Weapon (number) attacks against skeleton correspond to a kind of division



	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

gameplay tightly woven with academic content

Image taken from serious.gameclassification.com



Zombie Division

Prototype v.1

August 2005

(played by a 7-year old male)

Criticisms

- Diversity
  - Narrow scope of academic material covered
  - E.g., Math is just division
- Sustainability
  - Game eventually collapses once player learns the material
  - No long term play and/or no replay value
- Solution:
  - Ideas from strategy games and learning theories

Learning Theories

- Metacognition
- Self-theory
- Feedback

Learning Theories

- Metacognition
  - Reflecting on one's own learning process
  - Examples:
    - Awareness you have difficulty with names
    - Reminding yourself to take notes to help remember content
    - Monitoring your own progress
    - Developing strategies of improvement
  - Ability to optimize depth of learning and transfer
- Self-theory
- Feedback

Learning Theories

- Metacognition
- Self-theory
  - Beliefs we hold about our own ability to learn
  - Growth mindset: belief one can improve (high resilience to failure)
  - Fixed mindset: belief one's intelligence is a fixed quantity (detrimental to skill acquisition)
- Feedback

Learning Theories

- Metacognition
- Self-theory
- Feedback
  - Enables effective metacognition
  - Facilitates self-reflection
  - Effective feedback: timely and formative
  - Note:
    - Formative: how task is done (focus on thought and behaviour)
    - Summative: how well task is done (focus on results)

Interaction between Feedback and Mindset

- Fixed mindset
  - Intelligence is innate, unchanging
  - Avoids challenges, gives up easily
  - Defensive about feedback, easily discouraged
  - Feedback: “You’re so smart!”

- Growth mindset
  - Results of hard work, intelligence can be improved
  - Embrace challenges, perseveres
  - Welcomes feedback, focus on making changes
  - Feedback: “You’ve worked really hard!”

Importance of Feedback

- Opportunity to implement effective feedback mechanisms into games
  - Use 1-on-1 competition as feedback mechanism

- Good to mix growth mindset feedback and fixed mindset feedback
  - Don’t provide only fixed mindset feedback

Chess and StarCraft II

- Design motivation from successful games
- Diversity:
  - Both games have rich body of theory surrounding the game play strategies
- Sustainability:
  - Both games have extraordinary replay value

StarCraft II

- Real time strategy game by Blizzard
- Released in 1998
- Players assume control of a futuristic army and struggle for military dominance
- 4 million competitive players
- Unofficially dubbed the “national sport” of South Korea

More Theory

- Both games …
  - Produce domain experts
  - Require incredible skill, precision, focus, acquired knowledge
  - Take years to master (estimated at 50,000-100,000 hours of time on task)

How to design a new game that could sustain 50,000 hours of diverse play?!

Designing for Balance

- Carefully monitor data from all games
- Periodically “rebalances” game by releasing patches that tweak relative value of certain game units
- Enforces balance among players
- Maintains diverse gameplay

Assessment and Feedback Mechanisms

- Single player games
  - Assessments and feedback must come from environment
  - Designing assessments and feedback falls to designer
  - Diversity depends on types of activities that can be checked and reported

Assessment and Feedback Mechanisms

- 1-on-1 competitions
  - Assessments and feedback must come from the opponent – scales well
  - Win/loss becomes results driven, reduces the complexity of a task to a 1-bit judgment
  - Reasons to winning/losing become unclear
  - Community support for post-mortem activities, peer learning, strategy analysis
  - Aggregate wins/losses create meaningful ranks

Survey on Metacognitive Activities: Chess

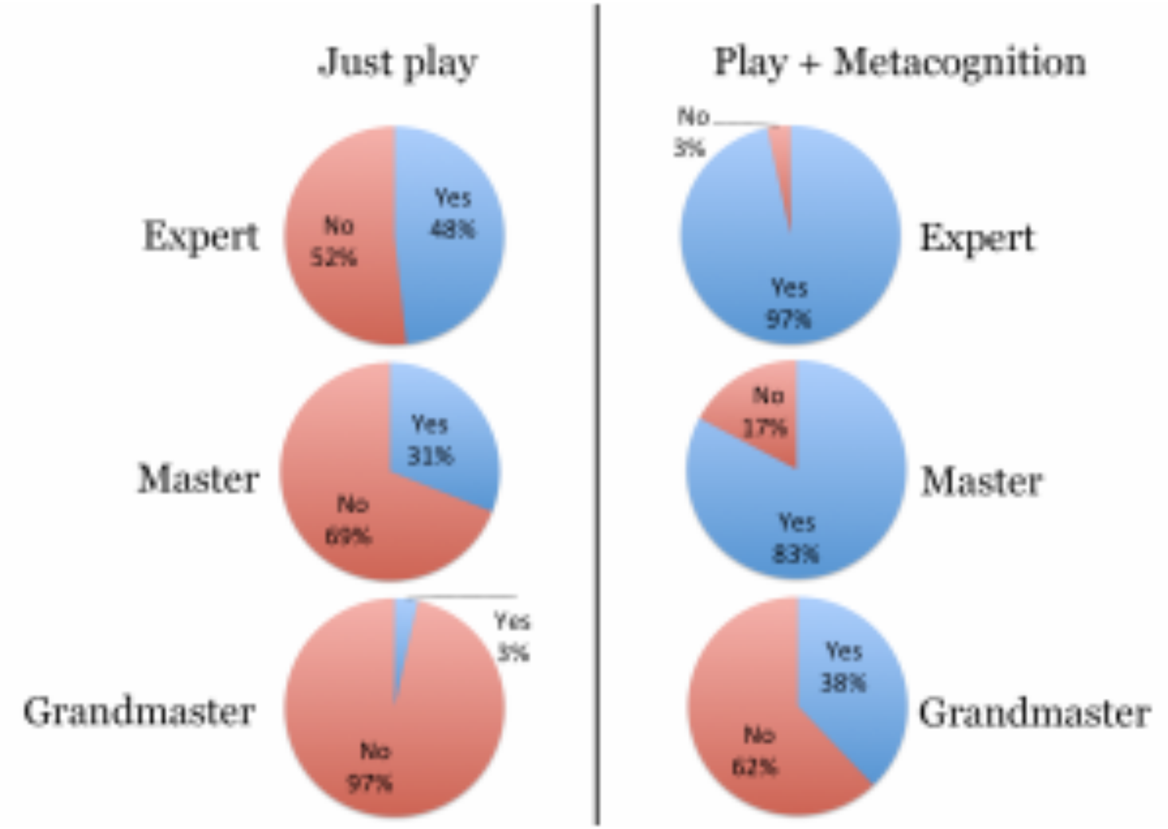


Figure 2. Results of a survey investigating chess players' perceptions of how proficient one can get with and without the benefit of metacognitive activities. For example: 48% of those surveyed believed that the level of Expert could be achieved by *only* playing chess.

Survey on Metacognitive Activities: StarCraft II

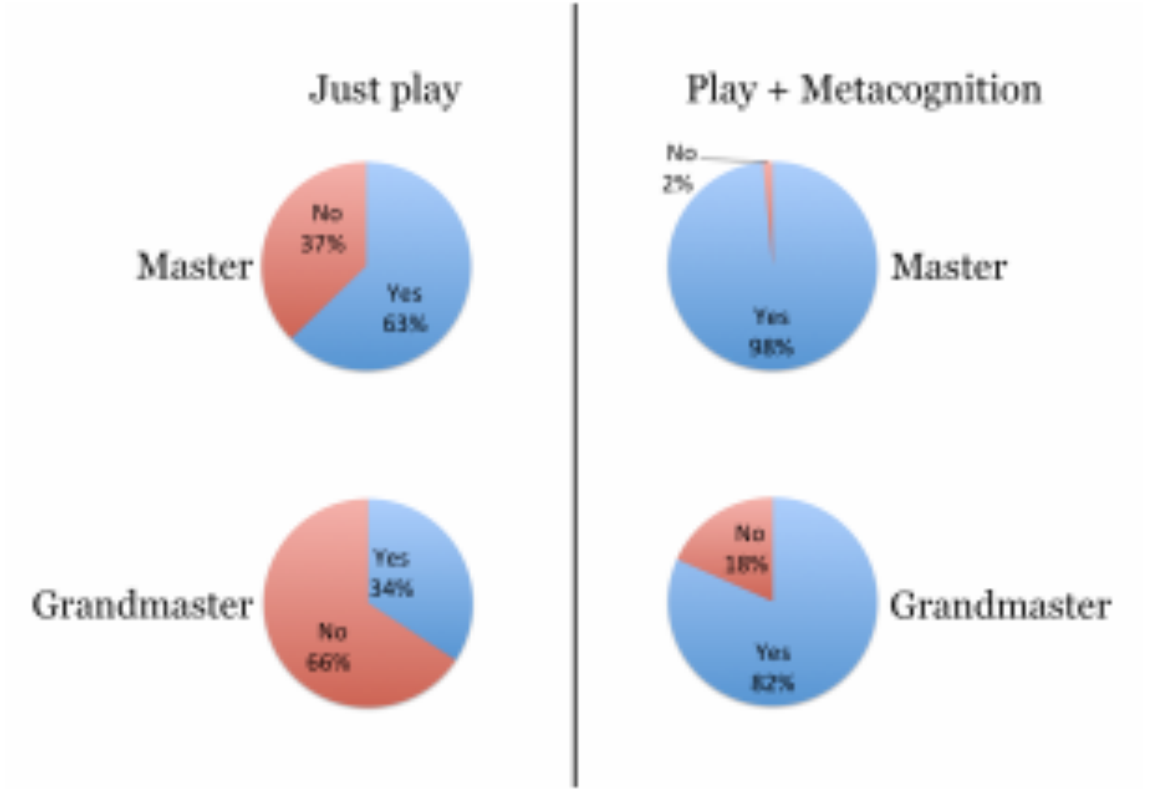


Figure 3. Results of a survey investigating *StarCraft II* players' perceptions of how proficient one can get with and without the benefit of metacognitive activities. For example: 34% of those surveyed believed that the level of Grandmaster could be achieved by *only* playing *StarCraft II*.

Survey on Self-Theory

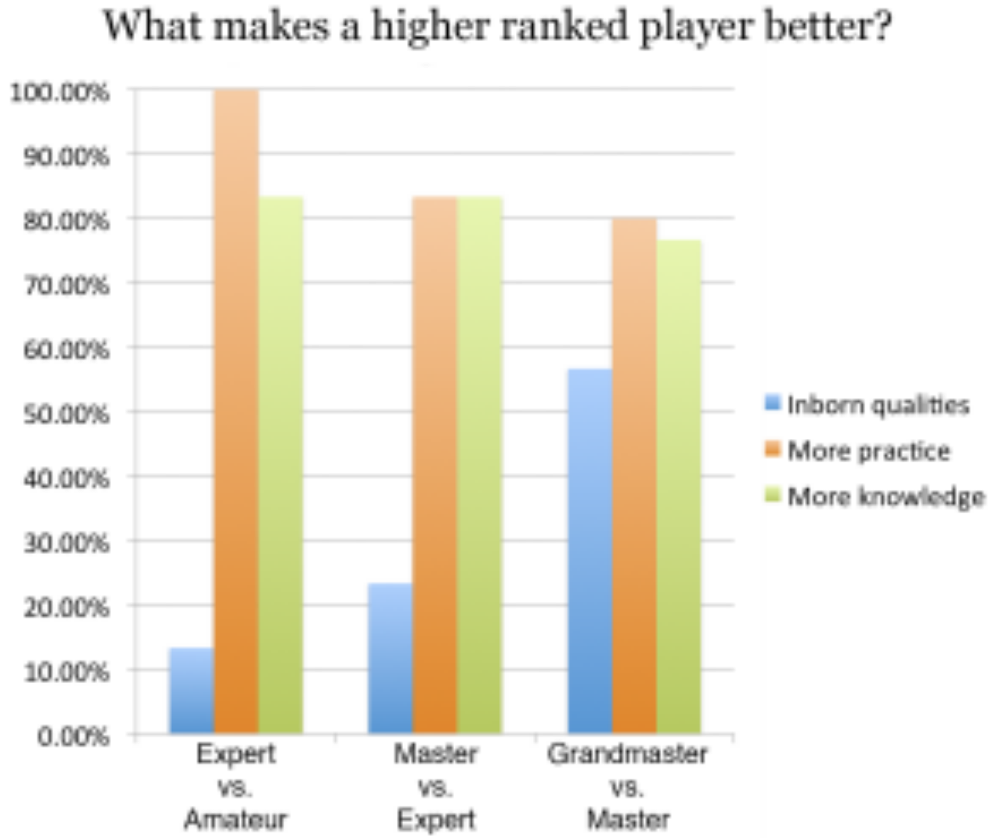
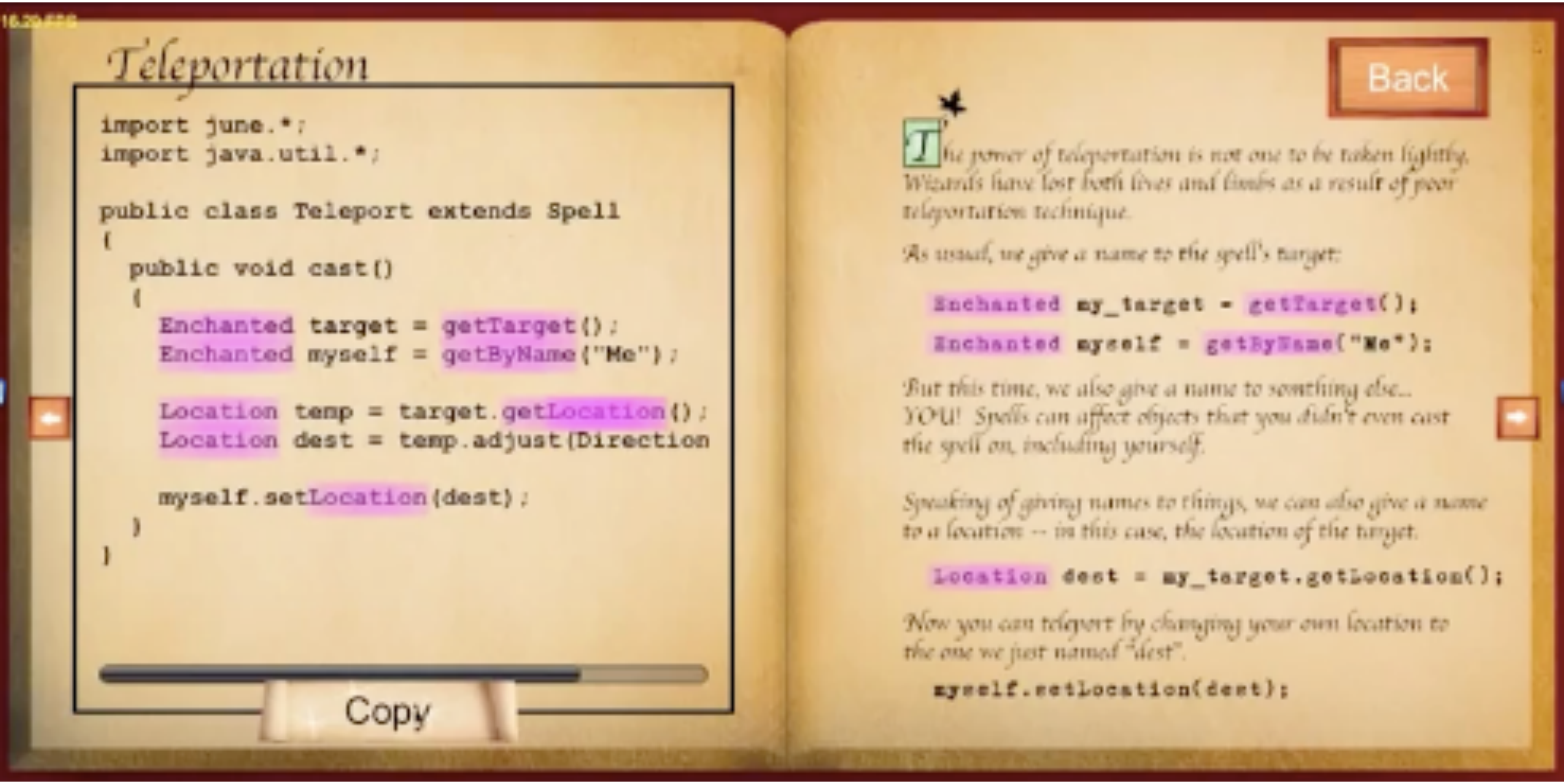


Figure 4. Results of a survey investigating chess players' perceptions of what makes a higher ranked player better than those in a slightly lower-ranked category. For example: 100% of those surveyed believed that experts are better than amateurs due (in part) to having practiced more.

CodeSpells

- A single player non-competitive game
- Targeted at novices learning Java
- In-game IDE for players to write spells
  - Spell-crafting Java API for manipulating the game 3D environment

CodeSpells



Teleportation

```
import june.*;
import java.util.*;

public class Teleport extends Spell
{
  public void cast()
  {
    Enchanted target = getTarget();
    Enchanted myself = getByName("Me");

    Location temp = target.getLocation();
    Location dest = temp.adjust(Direction

    myself.setLocation(dest);
  }
}
```

The power of teleportation is not one to be taken lightly. Wizards have lost both lives and limbs as a result of poor teleportation technique.

As usual, we give a name to the spell's target:

```
Enchanted my_target = getTarget();
Enchanted myself = getByName("Me");
```

But this time, we also give a name to somthing else... YOU! Spells can affect objects that you didn't even cast the spell on, including yourself.

Speaking of giving names to things, we can also give a name to a location -- in this case, the location of the target.

```
Location dest = my_target.getLocation();
```

Now you can teleport by changing your own location to the one we just named "dest".

```
myself.setLocation(dest);
```

Cooperative Context

Competitive Sorcery in CodeSpells

- Provide a large possibility space
  - Players write Java code to craft spells to control various in-game objects in simulated world
- Provide competitive feedback
  - Iterated 1-on-1 competitions
- Perform a long term study
  - Put theory into practice – test various theoretical applications
- Small group of users
  - Shows game diversity even with small group

Competitive Sorcery Design

- Implement 1-on-1 competition into existing game
- One red and one blue wizard compete in arena
- Cause collisions between opponent and wooden boxes
  - Collision reduces wizard health
  - Boxes of opposite colour cause collisions
  - Boxes (of same colour) can be assembled into structures (e.g. towers)
  - Create and move boxes via code snippets

Evaluation

- Evaluate:
  - Pedagogical merit in community discussions outside game
  - Sustainability
  - Diversity
- 3 players all with < 2 years of undergrad instruction
- Over 2 months of play
- Players met 3+ times per week for 2 hours each
  - Total 150 hours of play time and/or metacognitive activities

The “Bomb” Strategy

- Write an infinite loop that spawned a new box at some arbitrary location on each iteration
- Caused physics engine to spawn flying boxes in non-deterministic directions
- Brings consistent victory to whomever used it
- Outcome: forces both players to use same strategy
- Ends with game’s imposed limit of 100 boxes
- Strategy prevailed for two days

The “Battering Ram” Strategy

- One player discovered she could cast a spell to pick up and carry a box
  - Use loop to change relative location of box
- Carry the box towards opponent to inflict damage
- More precise than the bomb
- The 100-box limit not applicable
- Game became a fencing match

The “Fokker” Strategy

- Player discovered that he could cast spell to move a box while standing on top of it
  - Effect: character is now flying
- While flying, character can simultaneously apply the Bomb strategy
- Also gave rise to the Fokker Battering Ram strategy

The “Gun” Strategy

- Spawn a box from the character’s location, then launch it forward
- Creates a dangerous projectile
- Initially created as a ground based strategy against the Fokker
- Later developed into the Fokker Gun Strategy

Diversity and Sustainability

- Pool of strategies began to grow
- Wiki page describing 30+ strategies after 150 hours of play
- Other strategies include:
  - Building walls, bridges, stairwells
  - Moving platforms

Pedagogical Merit

- Coding constructs used
  - A mix bag of first year programming techniques
  - Spells employed conditions, loops, arrays, boolean expressions, etc.
  - More powerful spells tend to be more flexible ones (with more control structures)

- Software engineering skills
  - Modularity
  - Design (strategy) against changing requirements

Pedagogical Merit

- Metacognition
  - Post-mortem activities to analyze game play
  - Losses provide feedback to highlight coding weaknesses
  - “I lost because I need to write that spell faster next time. I couldn’t remember it well enough.”
  - “I messed up the loop, so the spell didn’t work.”
  - Problem solving discussions to develop better counterattacks

KickStarter Project

A4 Deployment

- Need email to add to UBCO’s account at Google Play
  - “Mobile Educational Games @ UBCO”
  - Will invite one member from each team to the account
  - Make sure you can “Create Application”
  - Allow sufficient review time before due date (at least 3 days)

A4 Features

- Extra features to implement are up
- “Something you come up with”
  - Based on your A3 data, you can propose your own feature(s) to use
  - Must get approval for these features by last day of classes