

## Lightbot Lessons

- The act of directing a computer to do something ... called *programming*
- The Lightbot 2.0 game exhibited many properties of programming.

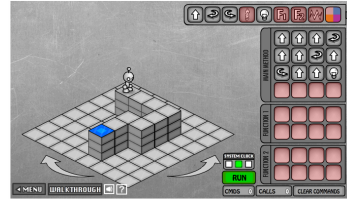
10/1/13

© 2010 Larry Snyder, CSE

1

## As Experienced Lightbot Hackers

- What are you doing in Lightbot?



- Commanding a robot through a “blocks world”
- Programming is **commanding** an agent

10/1/13

© 2010 Larry Snyder, CSE

2

## Agent, Instructions, Intent

- Other aspects of “commanding”
  - The **agent** is usually a computer, but it could be a person, or other device (animated robot?)
  - The agent follows the commands a/k/a **instructions**, flawlessly, and stolidly, doing only what it is asked
  - When the *program* executes, it doesn't always do what you *intended* for it to do. Rather, it does precisely what you **told** it to do. No ambiguity!

10/1/13

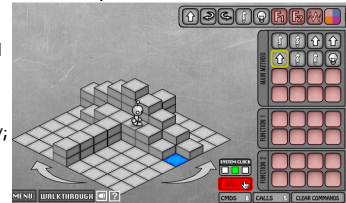
© 2010 Larry Snyder, CSE

3

## Sequencing

- Instructions are *given* in sequence
- They are *executed* in sequence – essential

- Instructions ...
  - From a limited repertoire
  - All are within agent's ability; no JUMP\_3
  - Executed one-at-a-time




- A “program counter” keeps track of agent's progress

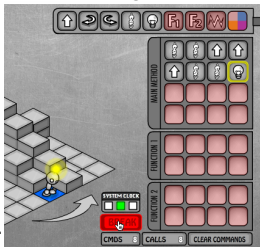
10/1/13

© 2010 Larry Snyder, CSE

4

## Instructions Formed of Simpler Instructions

- Check out this screen shot of the Lightbot
- It is partway through an instruction ... its beacon is lit, but not the tile
- To a programmer the instruction  is monolithic (one thing)
- To an agent each instr. is a series of steps



An Instruction *abstracts* those steps

10/1/13

© 2010 Larry Snyder, CSE

5

## Abstraction

- The word “abstraction” is used a lot in computing.
- As a general definition, abstraction *eliminates details to focus on essential properties*
- The instruction example just given illustrates *functional abstraction* meaning that we have given a **name** to a **series of operations** that perform a coherent (and to us meaningful) activity; the name is the instruction, the series of operations are the bot's actions to implement it

10/1/13

© 2010 Larry Snyder, CSE

6

## Abstracting

- Collecting the operations together and giving them a name is *functional abstraction*
  - The group of operations perform some function but we ignore all of the details
  - Giving it a name is *functional abstraction*
  - It doesn't seem like a big deal ... and if it wasn't AMAZINGLY powerful, it wouldn't be
  - What makes it powerful, is we can forget about the operations and think only about the function they do; more about this later
- Let's do some functional abstraction

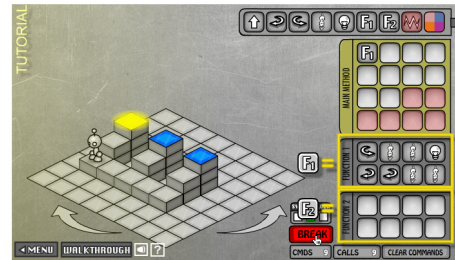
10/1/13

© 2010 Larry Snyder, CSE

7

## Functions Package Computation

- F1() packages actions: E.G. "process a riser"



10/1/13

© 2010 Larry Snyder, CSE

8

## The Function Becomes A Concept

- Because F1() "processes a riser," I think of the programming task as

Process a riser	F1
Move to next riser	F1
Process a riser	F1
Move to next riser	F1
Process a riser	F1

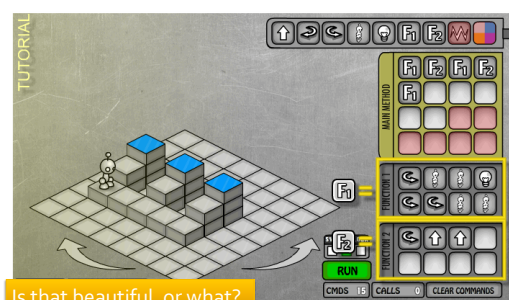
- With F1() as a concept, I simplify the programming to just 5 steps rather than 21
- It also suggests another concept:
  - Move\_to\_next\_riser()

10/1/13

© 2010 Larry Snyder, CSE

9

## A Five Instruction Program



10/1/13

© 2010 Larry Snyder, CSE

10

## Here Is What Is Beautiful ...

- Did everyone see 1 idea, 2 applications?

Slide 5

- To a programmer the instruction is monolithic (one thing)
- To an agent each instruction is a series of steps

Slide 10

F1(): Process Riser  
F2(): Move To Next Riser

It is one concept here (monolithic), but here it is a series of eight instructions

10/1/13

© 2010 Larry Snyder, CSE

11

## Abstraction ...

- Formulating blocks of computation as a "concept" is **functional abstraction**
- What we did is important here ...
  - We spotted a coherent (to us) part of the task
  - We solved it using a sequence of instructions
  - We put the solution into a function "package", gave it a name, "process a riser," and thus created a new thing, a concept, something we can talk about & use
  - Then we used it to solve something more complicated ... and probably repeat this approach at the next higher level

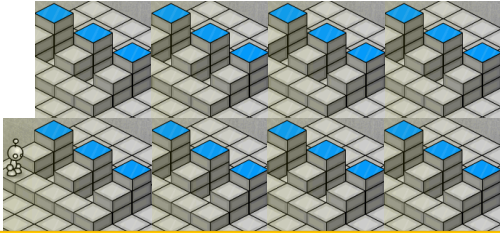
10/1/13

© 2010 Larry Snyder, CSE

12

## Keep Using Abstraction ...

- If M.C. Escher handed us a problem ... what would we do?



It only simplifies our thinking; the bot still does all the work

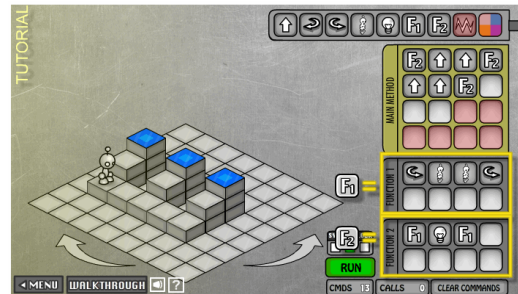
10/1/13

© 2010 Larry Snyder, CSE

13

## The Function Is Just The Packaging

- Another way to use a function for the risers



10/1/13

© 2010 Larry Snyder, CSE

14

## Lightbot 2.0 summary (so far)

- Programming is **commanding** an agent
  - **Agent:** usually a computer, person, or other device
  - Agent follows **instructions**, flawlessly & stolidly
- Instructions are *given* in sequence
- ... and *executed* in sequence
  - Limited repertoire, within ability, one-at-a-time
  - “Program counter” keeps track current instruction
- Formulating computation as a “concept” is **functional abstraction**

10/1/13

© 2010 Larry Snyder, CSE

15