FROM (LISP) TO ROBOTS FOR COMPUTATIONAL THINKING

PROF. DR. MARCUS SPECHT
WELTEN INSTITUTE
RESEARCH CENTRE FOR LEARNING, TEACHING AND TECHNOLOGY
OPEN UNIVERSITEIT NEDERLAND
MARCUS.SPECHT@OU.NL
TWITTER, FLICKR, FACEBOOK: MARCUSPECHT
Open Access, Open Content, Open Data, Running National and European Research Projects, Working with National Network of Schools and Innovators
90’s Research Questions

- **Developing Expertise in CT**: Syntax vs. Conceptual Knowledge Development in Visual Programming

- **Developing Expertise in CT**: Example Based Programming and Personalised Examples

- **Feedback and Learning**: Design of Feedback in Online Learning, formative feedback, stacked feedback, freedom of exploration

- **Technical and Pedagogical Integration** of Evaluation and Tutoring Services with Web-Based Textbooks

- **Technical and Pedagogical Design** and effects adaptive navigation support and personalised recommender systems
Syntax vs. Conceptual Knowledge

Figure 1: The syntax-driven structure editor

Towards Adaptive Learning Environments

Peter Brusilovsky, Marcus Specht, and Gerhard Weber
University of Trier
E-Mail: {plb | specht | weber}@cogpsy.uni-trier.de
User Modeling and Adaptive Navigation Support in WWW-Based Tutoring Systems

Gerhard Weber and Marcus Specht*

Department of Psychology, University of Trier, Germany

To the Exercises

Exercises for Practicing

Here are some exercises for the calling of list access functions. The correct solutions are given and explained on the following page. But firstly, try to solve the problems yourself.

Exercises:
You should work at the following exercises:

What is the result of evaluating the following expression?

(ERROR, if the evaluation will result in an error!):

```lisp
(FIRST (REST (BROT KAFFEE MILCH ZUCKER)))
```

```
Error
```

```lisp
(FIRST (REST '((BROT KAFFEE MILCH ZUCKER)))
```

```
Kaffee
```

```lisp
(FIRST '((REST (BROT KAFFEE MILCH ZUCKER)))
```

```lisp
```
```
ADI: recommendations and open learner models

Figure 2: Inspecting the Learner Model. The student may decide to visit a topic of low knowledge (e.g. “History of Heroin”) or to “protest” against the agent’s estimation. In case of a protest, after the student performed some tests, the model will be updated.
#1 How to design an integrated Syllabus?
**CS concepts in Scratch**

- Sequence, Iteration (loops)
- Random, Boolean logic
- Variables, Lists (arrays)
- Events, Threads, Synchronisations
- Procedures, Parameters
- Cloning, Physical Sensing

**Computational Concepts Supported in Scratch**

In the process of creating projects with Scratch, young people develop as computational thinkers. They learn concepts, engage in practices, and develop perspectives they can use to express their ideas with code. This list features fundamental computational concepts that are supported in Scratch.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Explanation</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>sequence</td>
<td>To create a program in Scratch, you need to think systematically about the order of steps.</td>
<td>go to x: 100, y: -100; say &quot;let's begin&quot; for 2 secs; play sound snap; play drum 0.5; move 10 steps</td>
</tr>
<tr>
<td>iteration (looping)</td>
<td>forever and repeat can be used for iteration (repeating a series of instructions)</td>
<td>repeat 5; play drum 1.25 for 0.25 beats; move 10 steps; turn (90°) express</td>
</tr>
<tr>
<td>random</td>
<td>pick random selects random integers within a given range.</td>
<td>set x to pick random (100 to 150)</td>
</tr>
<tr>
<td>conditional statements</td>
<td>if and if else check for a condition.</td>
<td>if x position &gt; 200, then move 200 secs</td>
</tr>
<tr>
<td>boolean logic</td>
<td>and, or, not are examples of boolean logic</td>
<td>if touching sprite and x position &gt; 200, then play sound move -30 degrees</td>
</tr>
<tr>
<td>variables</td>
<td>The variable blocks allow you to create variables and use them in a program. Variables can store numbers or strings. Scratch supports both global and object-specific variables.</td>
<td>when sprite clicked; forever; set x to 10; move 10 steps; when touching sprite; if x &gt; 200, then change x by 1</td>
</tr>
<tr>
<td>lists (arrays)</td>
<td>The list blocks allow for storing and accessing a list of numbers and strings. This kind of data structure can be considered a “dynamic array.”</td>
<td>set list to [food, fruit, vegetable]; say item of list; for 2 secs</td>
</tr>
<tr>
<td>string manipulation</td>
<td>You can change or get information about strings of letters using length of, letter of, and join.</td>
<td>if length of your word &gt; 10; say part of your word; change counter by 1; repeat length of list of fruit; say item of list of food; for 2 secs;</td>
</tr>
</tbody>
</table>

To learn more computational thinking, see scratched.gse.harvard.edu/ct/
Scenarios of using Scratch

• Storytelling: from simple animations to real stories

• Games development: from mathematical games to interactive visual games

• Arts and Creativity: geometry, mandalas, complex mathematical visualisations

• Music and Interactivity

• …
## Integration in Curriculum

<table>
<thead>
<tr>
<th>CT Concept, Capability</th>
<th>CS</th>
<th>Math</th>
<th>Science</th>
<th>Social Studies</th>
<th>Language Arts</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data collection</strong></td>
<td>Find a data source for a problem area</td>
<td>Find a data source for a problem area, for example, flipping coins or throwing dice</td>
<td>Collect data from an experiment</td>
<td>Study battle statistics or population data</td>
<td>Do linguistic analysis of sentences</td>
</tr>
<tr>
<td><strong>Data analysis</strong></td>
<td>write a program to do basic statistical calculations on a set of data</td>
<td>count occurrences of flips, dice throws and analyzing results</td>
<td>analyze data from an experiment</td>
<td>identify trends in data from statistics</td>
<td>identify patterns for different sentence types</td>
</tr>
<tr>
<td><strong>Data representation</strong></td>
<td>use data structures such as array, linked list, stack, queue, graph, hash table, etc</td>
<td>use histogram, pie chart, bar chart to represent data; use sets, lists, graphs, etc. to contain data</td>
<td>summarize data from an experiment</td>
<td>summarize and represent trends</td>
<td>represent patterns of different sentence types</td>
</tr>
<tr>
<td><strong>Problem decomposition</strong></td>
<td>define objects and methods; define main and functions</td>
<td>apply order of operations in an expression</td>
<td>do a species classification</td>
<td></td>
<td>write an outline</td>
</tr>
<tr>
<td><strong>Abstraction</strong></td>
<td>use procedures to encapsulate a set of often repeated commands that perform a function; use conditionals, loops, recursion, etc.</td>
<td>use variables in Algebra; identify essential facts in a word problem; study functions in algebra compared to functions in programming; use iteration to solve word problems</td>
<td>build a model of a physical entity</td>
<td>summarize facts; deduce conclusions from facts</td>
<td>use of simple and metaphor; write a story with branches</td>
</tr>
<tr>
<td><strong>Algorithms &amp; procedures</strong></td>
<td>study classic algorithms; do</td>
<td>do long division, factoring; do</td>
<td>do an experimental procedure</td>
<td></td>
<td>write instructions</td>
</tr>
</tbody>
</table>
#2 How to define learning tasks?
LEGO WeDo 2.0 & Scratch

The LEGO® Education WeDo 2.0 is an introductory invention kit you can use to build your own interactive machines. You can snap together Scratch programming blocks to interact with your LEGO WeDo creations and add animations on the screen.

1. Install Device Manager
The Device Manager lets you connect WeDo 2.0 to Scratch using Bluetooth
   Download for Mac OSX
   Download for Windows 10+

2. Setup & Help
Connect your WeDo 2.0 by following the steps in the Tips Window

3. Create
Use the WeDo extension blocks to turn on lights, control motors, and make your project interactive
# WeDo Extension Blocks

## LEGO WeDo Extension Blocks

<table>
<thead>
<tr>
<th>Block</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>turn motor on for secs</td>
<td>Turns a specific motor or the lights on for a certain amount of time. There are five options for the block, listed as “motor”, “motor A”, “motor B”, “light” and “everything”.</td>
</tr>
<tr>
<td>turn motor on</td>
<td>Turn a specific motor or the lights on indefinitely.</td>
</tr>
<tr>
<td>turn motor off</td>
<td>Turn a specific motor or the lights off.</td>
</tr>
<tr>
<td>set motor power</td>
<td>This block sets the power of a specific motor or the lights, controlling the speed at which the motor is spinning or the brightness of the lights.</td>
</tr>
<tr>
<td>set motor direction</td>
<td>This block sets the direction that a specific motor should turn with. There are three options for the direction, listed as “this way”, “that way”, and “reverse”. The first two are equivalent to clockwise and counter-clockwise. Reverse switches the direction.</td>
</tr>
</tbody>
</table>

## Sensor Blocks

<table>
<thead>
<tr>
<th>Block</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>when distance &lt;</td>
<td>This hat block runs a script when the distance becomes less (or greater) than a specified value.</td>
</tr>
<tr>
<td>when tilt</td>
<td>This hat block runs a script when the tilt value becomes equal (or not equal) to a specified value. The tilt sensor returns 0-4, with 0 indicating not tilted, 1 tilted down, 2 tilted right, 3 tilted up, and 4 tilted to the left.</td>
</tr>
<tr>
<td>distance</td>
<td>It reports the distance sensor value.</td>
</tr>
<tr>
<td>tilt</td>
<td>It reports the tilt sensor value.</td>
</tr>
</tbody>
</table>
Making use of the sensors

Build a motorized “crane” or lift-arm using the WeDo kit. The crane must be motorized and must have a tilt sensor attached to the arm.

Program with Scratch so that the motor turns on and stops after the crane is “up” (according to the tilt sensor value).

Bonus: Reverse the motor and lower the crane until the arm is down again.

Build a game controller with a tilt sensor so that you can control an on-screen Sprite to go right, left, up and down as you tilt the controller.

In Scratch, create a maze using a ball (Sprite 1) and walls (Sprite 2). Use the controller to travel through the maze.
SRA Programming EV3

- Higher analytical skill when applying more SRA programming
- Tendency for higher mathematical skill when applying more SRA loops
#3 How to raise awareness and link to families and everyday life?
Geef je kind een kickstart de toekomst in
Leerzame video's en leuke challenges voor kinderen van 6+ (en hun ouders!)

**Uitvinden met de Micro:bit**
In deze missie ga je een digitaal huisdier, games en wearables uitvinden en maken met de Micro:bit.

**Bedenk, ontwerp en bouw een App**
In deze missie ga je niet alleen een app bedenken en ontwerpen... Ja gaat 'm ook echt bouwen!

**Robots ontwerpen en programmeren**
In deze missie ontdek je van alles over robots: hoe ze werken, welke taal ze spreken en hoe ze te besturen.

https://youtu.be/EXfWOWJbo7w
Verzamel de materialen:

1. Microbit
2. battery pack + 2 batterijen
3. laptop of computer
4. usb kabel
5. schaar
6. lijm
7. sjabloon geprint op karton
8. spul om te customizen

https://youtu.be/YkJq9MJ21C8
Key Challenges

• #1 How to design an integrated Syllabus? Integration with Curricula and Subjects for teacher professionalisation

• #2 How to design learning tasks? Empirical Evidence for the effects on 21st century skills, consequences of SRA programming with robots

• #3 How to motivate? Broader embedding in everyday life and parents involvement.