Clicker question: Feedback on RQ8

- To which extent did the reading you did for today (RQ8) help you understand programming concepts?

  A. It helped me a lot.
  B. It helped a little.
  C. It didn’t help – I already knew the material.
  D. It didn’t help – I still have difficulties.

Selected RQs

(a) Some similarities I noticed between Scratch and Javascript are their use of loops and variables (i.e., “If … else” block in Scratch, and “If … else” script in Javascript). Is Scratch built based on Javascript, or should Scratch be considered as a totally different programming language?

(b) The Scratch Tutorial and Javascript for Beginners web pages both mentioned the programming language Java, while the Javascript for Beginners webpage mentioned that Java and Javascript are completely different scripting languages, are there absolutely nothing in common between the two types of scripting languages?

(submitted by Jeremy)

Selected RQs

- Is Javascript a programming language? If so, what is it compiled by? The internet browser? Or is it not compiled but interpreted by an internet browser?

(submitted by Anne-Sophie – 2011W1 Student)
Loops

for (i=1; i <= max; i++) {...}

is the same as

i=1;
while (i <= max)
{
  ...
i++;
};

Clicker question: for-loops

How often is the body of the following for loop executed?

for (n=1; n <= 100; n++) {...}

A: once
B: 100 times
C: 101 times
D: 99 times

Clicker question: for-loops

How often is the body of the following for loop executed?

for (n=1; n <= max; n++) {...}

A: once
B: max times
C: max+1 times
D: max-1 times

Clicker question: for-loops

How often is the body of the following for loop executed?

for (n=0; n <= max; n++ n+1) {...}

A: once
B: max times
C: max+1 times
D: max-1 times
Clicker question: for-loops

How often is the body of the following for loop executed?

```c
for (n=0; n < max; n=n+1) {...}
```

A: once  
B: max times  
C: max+1 times  
D: max-1 times

Clicker question: for-loops

What is the value of variable n after the loop has finished executing?

```c
n=0;
while (n < max)
{
    ...
    n++;
}
```

A: 1  
B: max  
C: max+1  
D: max-1

Clicker question: for-loops

What is the value of variable n after the loop has finished executing?

```c
for (n=0; n <= max; n=n+1) {...}
```

A: 1  
B: max  
C: max+1  
D: max-1

Tips & Tricks

How to figure out what a piece of code does:

1. Model – form a hypothesis of what a piece of code does  
   (often best done through running / simulating it for simple / small inputs)
2. Predict – predict what it will do for inputs not yet tried / simulated
3. Experiment – run / simulate to check your prediction
4. Refine – modify your hypothesis based on the result from the experiment; continue from Step 2.

Note: This is also known as the **scientific method.**
Tips & Tricks

Simulating a piece of code:

Trace values of all important variables through all statements.

Experimenting with a piece of code:

Instrument the code with additional output statement, so you can see the values of variables at various points of execution.

Exercise: for and while

Rewrite the following while-loop as a for-loop:

```java
sum = 0;
i = 1;
while( i <= 5 ) {
    sum = sum + i;
    i++;
}
```

Hint: To check your solution, trace the values of `sum` and `i` through all iterations of the loop (both versions).

Tips & Tricks

Tracking down reasons for incorrect behaviour (bugs):

Add automatic checks (using if-statements, covered later) to your code that ascertain that certain conditions you expect to hold actually do hold, or that catch incorrect behaviour (= deviations from your expectations).

What next?

Fundamental concepts:

- Instructions (input, output, drawing, math, …)
- Sequences of instructions
- Variables
- Loops / repetition (repeat, until, while, for)
- Procedures / functions
- Conditional statements (if, else)
- Arrays
Clicker question: Functions

Advantages of using functions:

(A) Break up a big process into smaller parts
(B) Give a name to part of a process
(C) Allows you to re-use components (share) parts between processes, and between people
(D) All of the above.

Key concept: Functions

• Break up a big process into smaller parts
  – Handle the complexity of big processes
• Give a name to part of a process
  – Improve the readability of your processes
• Allows you to re-use components (share) parts between processes, and between people
  – Independence: you can change in one place for everyone
  – Abstraction / Teamwork: you don’t have to know how it works, and someone else can write your functions

Function: Build a House

• Build a house

Function: Build a House

• Build a house
  – foundation
  – build frame
  – build floors
  – electrical work
  – plumbing
  – drywall
  – ...
Function: Build a House

- Build a house
  - foundation
  - build frame
    - ...
      - hammer a nail
  - build floors
    - ...
      - hammer a nail
- build floors
  - ...
    - hammer a nail

Exercise: Functions

Consider the following function:

```javascript
function showTip(amount) {
  var tip;
  tip = amount * 0.12;
  document.write("the tip is: $" + tip);
}
```

Modify this function so that the tip rate (set to 12% above) is a parameter.
Answer:

```javascript
function showTip(amount, tipRate) {
    var tip;
    tip = amount * tipRate;
    document.write("the tip is: "+ tip);
}
```