Thicccque

October 21, 2018

Dr. Parham-Mocello

Assignment 2 Part 2

Readdress Orientation:

1. After seeing other papers that peers read, how did this change your view on the SIGs you chose or the papers you chose?
    - I'm feeling a bit petty, so to be perfectly honest, the only thing I'd really change is the amount of time and effort I put into the descriptions and summaries. I think the students I reviewed had a very different understanding of the assignment than I did, because one didn't follow the instructions and the other still seemed like they might have only read the abstracts for their summaries. I put a little too much effort in, and these assignments elucidated that, even if I doubt one or both of them are going to get a perfect score on this assignment. I'm glad I chose the SIGs I did, and I've realized the papers I read were maybe too long (at least the YouTube AI proceeding). Either my papers and SIGs were way more interesting to me than the SIGs my review-ees chose were to them, or they didn't particularly care about the assignment. I did find it interesting that both my students chose very practical SIGs that are about skills important to working in industry, whereas I intentionally tried to stray from that. There's always the chance that they find these things fascinating, but they aren't all that interesting to me--yet, at least. I like my SIGs and I like my papers, but theirs were cool too!

2. After seeing other definitions by peers, how do your definitions of the CS terms change?
    - One of the sets of definitions I read were extremely clear and had examples in general terms, and I really appreciated them. I'd definitely add things like "think of shareware as anything that says 'free trial'" for example. I think we used similar sources for our information, which I found interesting, because the verbiage is similar. My other student used extremely simplified and very much non-thorough definitions that didn't act as any kind of impetus for me to adjust my own.

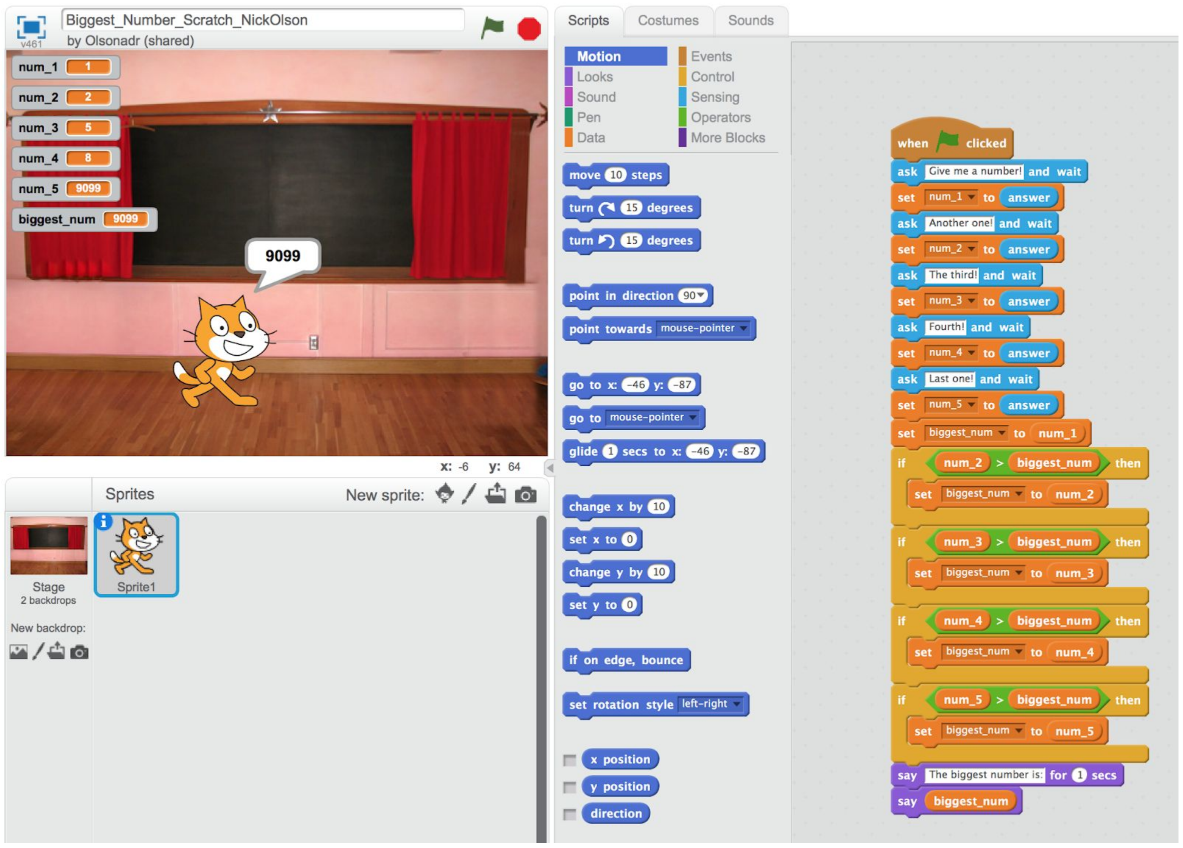
Readdress Computation:

3. After reviewing your peers' problem-solving steps and programs…
    - How would you change the way you expressed your problem-solving steps?
        - Again, I put **way** too much effort into these steps. Not only did I go too deep into every little possible way someone could break my algorithm, but I also used a wordy prose

structure that isn't super conducive to smaller projects or problems like this. It might work well for me in bigger, more complicated situations, but I'd definitely try to mimic the bullet-point or ordered-list structure of my review-ees in the future, at least once each. Their representations of the first two steps were more readable. I do like that I did follow the problem-solving steps exactly, which neither of my review-ees did fully, and I'm starting to see the benefits of doing so, even if it can be tedious. I wouldn't change how I showed each step, actually writing things out on paper, or using screen-grabs in case the code doesn't get uploaded correctly again.

- How would you change the content in these steps?
    - I do like the detail I put in, and I had fun writing it in a somewhat joking tone, but my detail went too far sometimes. I do think it's a good idea to put a lot of thought into the 'understanding the problem' and 'devise a plan' steps, but theirs were serviceable without the extra time and effort. I do think I did a good job of fully fleshing out the steps and doing them justice.
- How would your implementation of the plan change?
    - I didn't go too crazy, but I hadn't considered at all that using things like lists was too advanced for where we currently are in the class. I think I could keep using new things we haven't learned yet, but only if I keep up the excessive commenting I used to ensure clarity. I am worried that maybe a student who is new to computer science or programming might still be confused, so it might be worth limiting myself to what we've learned, as an exercise if not a standard. I thought it was interesting that I could clearly tell (by the conventions followed or not, the use of while loops or counter variables, or the code's efficiency [in code length or memory]) how experienced with programming my review-ees were. One of them was clearly brand new and the other definitely had some experience. The new person had super easy to read code even if it wasn't all that nuanced, whereas the more experienced person's code was also easy to read, but that's coming from an experienced reader. The lack of comments on the while loop, for example, might have tripped up an inexperienced reader, so reviewing this code definitely enforced in my mind that I want to explain or link to professional explanations of absolutely everything I use that we haven't gone over in class, to make sure anyone can get something out of my work. The most surprising thing though is that my less-experienced review-ee put an exit() statement at the end of their code. Curious.

Extra Credit:



```
when [flag] clicked
ask [Give me a number!] and wait
set [num_1] to (answer)
ask [Another one!] and wait
set [num_2] to (answer)
ask [The third!] and wait
set [num_3] to (answer)
ask [Fourth!] and wait
set [num_4] to (answer)
ask [Last one!] and wait
set [num_5] to (answer)
set [biggest_num] to (num_1)
if <(num_2) > (biggest_num)> then
    set [biggest_num] to (num_2)
if <(num_3) > (biggest_num)> then
    set [biggest_num] to (num_3)
if <(num_4) > (biggest_num)> then
    set [biggest_num] to (num_4)
if <(num_5) > (biggest_num)> then
    set [biggest_num] to (num_5)
say [The biggest number is:] for (1) secs
say (biggest_num)
```

(The scratch file is in the Assignment_2_Part_2.zip download).