CS10 Spring 2018 Quest Answers

Question 1: Which of the following is the worst example of Abstraction? Designing a traffic simulation and including everything about all cars in your model: color, year, make, etc. [The point of Abstraction is Removing Detail and Generalization, how does the color of a car affect its performance in a traffic simulation? It doesn't, we would normally remove that detail to focus on the important aspects, like its (x,y,z) position over time]

Question 2: What is the order of these numbers, smallest to largest: \( A=11000_2 \); \( B=29_{10} \); \( C=1E_{16} \)? \( A=11000_2 \) which is \( 1*16+1*8+0*4+0*2+0*1=24_{10} \), \( B=29_{10} \), and \( C=1*16+14*1=30_{10} \), so it's \( A < B < C \).

Question 3: If the sprite starts in the middle of the stage facing up and runs the script to the right with line width set to 1 (the default), what is drawn? It's a rectangle (a square “smeared” over and over 100 times).

Question 4: Given the expression that runs without error, what is your best guess as to the Domain and Range of \( \text{map} \)? (select ALL that apply for each side) \( \text{map} \) return a list so the Domain is lists (at least, it might take more than that), and \( \text{not} \) takes in a Boolean so the Range is Booleans.

Question 5: If the output from Mystery is false, which can you say for sure? If \( A \) is true, it returns true so \( A \) must be false (in which case it goes to the else and returns \( B \), but if it returns false then \( B \) must be false too!)

Question 6: You realize you could replace the entire body of Mystery with a single report (as shown below). “What could go in there so that it will function exactly the same as the original Mystery block?” The block returns true when \( A \) is true OR when \( A \) is false and \( B \) is true, which is exactly what \( A \) or \( B \) does.

Question 7: Match each programming paradigm with properties that describe it. Seems like “magic”; great for logic puzzles? Declarative. Doesn’t allow for any side-effect procedures? Functional. It’s all about message passing and inheritance? Object-Oriented. Do this, then that, then that. Aka sequential? Imperative

Question 8: The makers of the Nozama smart speakers learn that users who ask for the weather don’t want the exact temperature, they want a temperature category: Cold, Cool, Warm or Hot, based on the table below. E.g., Cool is between 40 and 60. They write code for it, shown to the right. However, when given the following temperatures, what is actually returned?" 30? Less than 80 so Warm! 50? Less than 80 so Warm! 70? Less than 80 so Warm! 90? Not less than 80, not less than 60 so Cold.

Question 9: You have a list of NUMBERS; if any two different numbers add to 100, AddTo100 should be set to true. Here is our (possibly buggy) algorithm. When will it return true on a list of 10 numbers? If the first two are 40,60 and the rest are 0? No. If the last two are 40,60 and the rest are 0. No. If the last one is 50 and the rest are 0. Yes. all ten numbers are 10? No

Question 10: What is the running time of the algorithm in Question 9? Quadratic, which is Reasonable Time.

Question 11: Are the left and right expressions below always the same? If stuck, put your own reporters and predicates in there, try it on an easy list, and see if they are the same… \( \text{map(keep())} \) vs \( \text{keep(map())} \)? What if \( \text{map} \) squares numbers and \( \text{keep} \) preserves only negative numbers (and \( D \) contains half positive and negative)? Then \( \text{map(keep())} \) would have half the numbers (the squares of the negatives) and \( \text{keep(map())} \) would be empty! So NO. \( \text{map(R1(R2))} \) vs \( \text{map(R1)over(map(R2))} \)? Each element elt, either case, is effectively \( R1(R2(elt)) \) so YES. \( \text{keep(P1(P2))} \) vs \( \text{keep(P1)suchthat(keep(P2))} \)? The domain of \( P1 \) is elements, but in the earlier case \( P1(P2(elt)) \) means that \( P2 \) returns Booleans and \( P1 \) may not have Booleans as its Domain, since \( P1 \) needs to be looking at elements! NO \( \text{keep(P1 and P2) vs keep(P1)suchthat(keep(P2))} \)? You keep the element when it passes both the \( P1 \) and \( P2 \) filter, which is AND! YES \( \text{keep(P1 or P2) vs keep(P1)suchthat(keep(P2))} \)? You keep the element when it passes both the \( P1 \) and \( P2 \) filter, not when it passes either! NO