Ryan Draper

CS 326

Reading • Ch. 6: Section 6.7–6.15

1. A)I’m unclear as to how Mark-sweep is done how can something become empty if the program is being run. Oh I see that it is a process of allocating all heap to garbage then pointers and then finally empty.

B)Is there technically pointers in Haskell I mean in the way C/C++ use them? Or is this not possible due to variables being immutable.

2. I understand why deleting of a pointer especially a pointer to an array is as important as it is in C++/C I always thought it was good practice but I didn’t understand the extent of problems that can occur with such carelessness. Also, the union in C/C++ is the only reason they are not a strongly typed language? I would have, if you only gave me the definition of a strongly typed language assumed that C/++ was a strongly typed language.

3. C/C++ cannot type check a union with more than one type and so a bet string representation is used. In Haskell it uses a discriminated union that uses a tag to allow for type checking of a union.

 4. Heap collection, **reference counters:** reclaimed cells is incremental and done as inaccessible ones are created. Every cell has a number of pointers that are to that cell. As those number of pointers get decremented the count become zero and is then garbage. **Mark-sweep:** runtime system allocates storage cells as garbage and pointers clean up and reallocate as pointers mark their use, all cells still garbage are considered free.

5. A strongly typed language always type checks a variable, such as C/C++ don’t type check their union, where java is strongly typed, javascript is only similar in syntax but is a dynamically typed.