

HW2: Binary Heaps

Due: Monday, November 16, at 11:59 PM, on Canvas

You may work on your own or with one (1) partner.

In this assignment, you will implement a fixed-size binary heap. The structure of the heap is already defined for you in `binheap.rkt`¹. The heap is represented using an ASL vector² to contain the elements. Each heap will also contain a comparison function for ordering the elements of the heap, so that your implementation can support heaps of integers, heaps of strings, heaps of whatits, heaps of sporkles, etc.

Your task

In `binheap.rkt`, I've supplied a definition of a function `create` that returns a new, empty heap given a capacity and ordering function. Implementing the remaining operations is up to you:

```
insert!      : [Heap X] X -> Void      ;  $\mathcal{O}(\log n)$ 
find-min     : [Heap X] -> X           ;  $\mathcal{O}(1)$ 
remove-min!  : [Heap X] -> Void        ;  $\mathcal{O}(\log n)$ 
```

For details, see the function headers provided in `binheap.rkt`, which include purpose statements as well. Each operation must have the worst-case shown above, where n is the number of elements in the heap. In order to help you factor your program effectively, I've included at the bottom of `binheap.rkt` a list of helper functions with names, signatures (types), and purpose statements (brief functional descriptions). You are free to use as much or as little of my design as you like.

Extra credit

Make your heap expand as necessary to accomodate any number of assertions. To achieve this, instead of failing when the heap is full, `insert!` should allocate a new vector that doubles the capacity and copy the existing elements over from the old vector.

Deliverables

- The provided file `binheap.rkt`, containing:
 - the `insert!`, `find-min`, and `remove-min!` functions fully defined, and
 - sufficient tests to convince yourself your code's correctness.

¹<http://users.eecs.northwestern.edu/~jesse/course/eecs214-fa15/hw/2/binheap.rkt>

²ASL vectors are like other languages' arrays in that the size is fixed at create time.