1 Code Walk Scheduler, Part 1

1.1 Additional Preliminaries

Save your solutions for this problem to a file named sched-with-unavail.rkt.

Run the following expression (you must have required [extras.rkt](http://www.ccs.neu.edu/course/cs5010sp15/files/extras.rkt)) to check that your file is properly named and is in the proper directory:

(check-location "10" "sched-with-unavail.rkt")

Add these additional provides at the top of your file (below the requires), so that we can test your solution:

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) schedule/unavail-ok?)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) schedule/unavail)

1.2 Problem Description

The PDP staff needs your help scheduling codewalks. For this assignment, you will implement a codewalk scheduling algorithm.

A codewalk schedule is an assignment of students to codewalk times, as described in the data definitions below. Complete the rest of the [Data Design](http://www.ccs.neu.edu/course/cs5010sp15/recipe.html#%28part._data%29) step for these data definitions. Add any additional data definitions as you see fit.

|  |
| --- |
| ; A CodeWalks is a ListOf<CodeWalk> |
| ; Represents a codewalk schedule. |
|  |
| ; A CodeWalk is a (make-codewalk Time ListOf<StudentID> PosInt) |
| ; Represents a codewalk with time, assigned students, and a max capacity. |
| ([define-struct](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define-struct%29%29) codewalk (time students max)) |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) CW-TUE135 (make-codewalk TUE135 [empty](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28def._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._empty%29%29) 1)) |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) CW-TUE325 (make-codewalk TUE325 [empty](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28def._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._empty%29%29) 1)) |
|  |
| ; A Time is a Symbol |
| ; Represents a day of week and time. |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) TUE135 '1:35pmTues) |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) TUE325 '3:25pmTues) |
|  |
| ; A StudentID is a Symbol |
| ; Represents a student (or pair) via their ccs ID(s). |
|  |
| ; A StudentUnavail is a (make-student StudentID Preferences) |
| ([define-struct](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define-struct%29%29) student (id prefs)) |
| ; Represents a student and their unavailable times. |
|  |
| ; A StudentUnavails is a ListOf<StudentUnavail> |
| ; UPDATE 2015-03-28: there are no duplicate StudentIDs |
|  |
| ; A Preferences is a ListOf<Time> |
| ; Represents a list of code walk times. |

**Tasks:**

1. Implement schedule/unavail-ok?, a predicate that checks whether a schedule satisfies a list of student preferences. A schedule is valid according to some preferences if:
   1. each code walk is not over capacity,
   2. each student is scheduled exactly once,
   3. each student is scheduled in an acceptable time slot according to their preferences.

|  |
| --- |
| ; schedule/unavail-ok? : StudentUnavails CodeWalks -> Boolean |
| ; Returns true if cws is a valid schedule according to the given |
| ; student preferences. |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) (schedule/unavail-ok? students cws) [...](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._......%29%29)) |

1. Implement the schedule/unavail function.

|  |
| --- |
| ; schedule/unavail : StudentUnavails CodeWalks -> Maybe<CodeWalks> |
| ; Creates a codewalk schedule by assigning students to cws, |
| ; while satisfying students' constraints. |
| ; Returns #f if no schedule is possible. |
| ; UPDATE 2015-03-28: the given students are not already scheduled in cws |
| ; Strategy: ??? |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) (schedule/unavail students cws) [...](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._......%29%29)) |

2 Code Walk Scheduler, Part 2

2.1 Additional Preliminaries

Save your solutions for this problem to a file named sched-general.rkt.

Run the following expression (you must have required [extras.rkt](http://www.ccs.neu.edu/course/cs5010sp15/files/extras.rkt)) to check that your file is properly named and is in the proper directory:

(check-location "10" "sched-general.rkt")

Add these additional provides at the top of your file (below the requires), so that we can test your solution:

2.2 Problem Description

**Tasks:**

* Add the following data definitions:

|  |
| --- |
| ; A StudentAvail is a (make-student StudentID Preferences) |
| ; Represents a student and their available times, most-preferred first. |
| ; An unlisted time means the student is unavailable. |
|  |
| ; A Student is one of: |
| ; - StudentUnavail |
| ; - StudentAvail |
|  |
| ; A StudentAvails is a ListOf<StudentAvail> |
| ; UPDATE 2015-03-28: there are no duplicate StudentIDs |
|  |
| ; A Students is a ListOf<Student> |
| ; UPDATE 2015-03-28: there are no duplicate StudentIDs |

* UPDATE 2015-03-28: [As mentioned on Piazza](https://piazza.com/class/i39e0qbcnd87a9?cid=746) (thanks Yang), a more accurate data definition for Students, which reflects the fact that lists of students are homogenous, is:

|  |
| --- |
| ; A Students is one of: |
| ; - StudentUnavails |
| ; - StudentAvails |
| ; WHERE: there are no duplicate StudentIDs |

* Don’t worry about templates for these data definitions, since none of your functions should have to decompose the Student itemization.
* Abstract your program such that it’s easy to implement versions of the scheduling predicate and algorithm from part 1 for either version of student. You will almost certainly need to user higher-order functions, à la [map](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28def._htdp-intermediate-lambda._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._map%29%29), [filter](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28def._htdp-intermediate-lambda._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._filter%29%29), and friends.

As one possible example, you might define a function:

|  |
| --- |
| ; schedule/general : Students CodeWalks ... -> Maybe<CodeWalks> |
| ; UPDATE 2015-03-28: the given students are not already scheduled in cws |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) (schedule/general students cws [...](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._......%29%29)) [...](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._......%29%29)) |

Then schedule/unavail would simply be a call to schedule/general. However, you may also choose to abstract in a different manner.

Either way, if you’ve been following the concepts in the course, and have organized your code in a clean manner, this is an extremely easy refactoring of no more than a few lines of code and should take no more than a few minutes.

**NOTE:** Your abstractions should not come at the expense of code readability. All your signatures and purpose statements must remain valid and clear. You will fail this assignment if this is not the case.

(If you are struggling with this part of the assignment, you may try to first complete parts 1 and 3, before coming back to part 2. If you do this, you should observe a large amount of code duplication between parts 1 and 3, and the possible abstractions may be more evident.)

* Split your code into two files:
  + sched-with-unavail.rkt: All code in this file only handles the first kind of student,StudentUnavail. This file still implements the schedule/unavailschedule/unavail-ok? functions from part 1.
  + sched-general.rkt: All code in this file handles the general Student data definition. For example, you might put the schedule/general function mentioned above in this file. This file should be [require](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._require%29%29)d by sched-with-unavail.rkt. You may find it helpful to add (provide (all-defined-out)) at the top of this file (make sure you have the most recent ["extras.rkt"](http://www.ccs.neu.edu/course/cs5010sp15/files/extras.rkt) file). However, no functions from this file will be accessed by our test suite.

If you have followed the concepts from this course, sched-general.rkt file should contain the bulk of the code (not counting tests).

**Testing Note:** If all the code was properly tested before splitting the files, it’s acceptable if a few of the generalized functions in sched-general.rkt do not have full test coverage, since these functions are expected to be instantiated and tested in sched-with-unavail.rkt.

3 Code Walk Scheduler, Part 3

3.1 Additional Preliminaries

Save your solutions for this problem to a file named sched-with-avail.rkt.

Run the following expression (you must have required [extras.rkt](http://www.ccs.neu.edu/course/cs5010sp15/files/extras.rkt)) to check that your file is properly named and is in the proper directory:

(check-location "10" "sched-with-avail.rkt")

Add these additional provides at the top of your file (below the requires), so that we can test your solution:

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) schedule/avail-ok?)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) schedule/avail)

([provide](http://docs.racket-lang.org/reference/require.html#%28form._%28%28lib._racket%2Fprivate%2Fbase..rkt%29._provide%29%29) avg-choice)

3.2 Problem Description

Implement a predicate and scheduling algorithm for the second kind of student,StudentAvail. sched-with-avail.rkt should [require](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._require%29%29) and re-use functions from sched-general.rkt. If you have properly organized your code, implementing these functions should take no more than a few lines of code.

**Tasks:**

1. Implement schedule/avail-ok?, a predicate that checks whether a schedule satisfies a list of student preferences. A schedule is valid according to some preferences if:
   1. each code walk is not over capacity,
   2. each student is scheduled exactly once,
   3. each student is scheduled in an acceptable time slot according to their preferences.

|  |
| --- |
| ; schedule/avail-ok? : StudentAvails CodeWalks -> Boolean |
| ; Returns true if cws is a valid schedule according to the given |
| ; student preferences. |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) (schedule/avail-ok? students cws) [...](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._......%29%29)) |

1. Implement the schedule/avail function.

|  |
| --- |
| ; schedule/avail : StudentAvails CodeWalks -> Maybe<CodeWalks> |
| ; Creates a codewalk schedule by assigning students to cws, |
| ; while satisfying students' constraints. |
| ; Returns #f if no schedule is possible. |
| ; UPDATE 2015-03-28: the given students are not already scheduled in cws |
| ; Strategy: ??? |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) (schedule/avail students cws) [...](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._......%29%29)) |

1. Implement avg-choice, which computes the average of the rank of the codewalks assigned to each student. For example, if each student is assigned their first choice, then avg-choice returns 1.

UPDATE 2015-03-20: More examples: If each student is assigned their second choice, then the function returns 2. If there are two students and one receives their first choice and the other receives their second choice, then the function returns 1.5.

This roughly indicates how well the scheduling algorithm performs.

|  |
| --- |
| ; avg-choice : StudentAvails CodeWalks -> PosReal |
| ; WHERE: (schedule/avail-ok? students cws) = #t |
| ([define](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._define%29%29) (avg-choice students cws) [...](http://docs.racket-lang.org/htdp-langs/intermediate-lam.html#%28form._%28%28lib._lang%2Fhtdp-intermediate-lambda..rkt%29._......%29%29)) |

1. Tinker with your implementation of schedule/avail, with the goal of producing a schedule that

UPDATE 2015-03-20: minimizes (not maximizes) avg-choice. In other words, the scheduler should try to give each student a more preferred choice whenever possible.

We will apply your scheduler to a dataset of (Boston) student-submitted preferences. Any submission that beats our reference implementation will be rewarded some (to-be-determined) number of bonus points.