A

FAILURE TO COMPLETELY FILL THE TOP PART OF THIS PAGE WILL LOSE 2%

THE UNIVERSITY OF BRITISH COLUMBIA
CPSC 110: MIDTERM 1 - MAY 18, 2012

Last Name: ___________________ First Name:__________________ UBC Student #: ____________

Signature: ___________________________

Lab Section: ___________ Lecture Section (circle): 201 Gregor 202 Jim

Important notes about this examination

1. **This exam has 2 separate parts.** You have 120 minutes to complete both parts. When you are done with part A raise your hand and we will collect that and give you part B. We recommend that you save yourself 30 minutes for part B.

2. Except for questions 1 and 2, this exam will be graded largely on how well you follow the design recipes. You have been given a copy of the Recipe Exam Sheet. Use it!

3. Put away books, papers, laptops, cell phones... everything but pens, pencils, erasers and this exam.

4. Good luck!

Rules Governing Formal Examinations

1. Each candidate must be prepared to produce, upon request, a UBCcard for identification.

2. Candidates are not permitted to ask questions of the invigilators, except in cases of supposed errors or ambiguities in examination questions. No questions will be answered in this exam. If you see text you feel is ambiguous, make a reasonable assumption, write it down, and proceed to answer the question.

3. No candidate shall be permitted to enter the examination room after the expiration of one-half hour from the scheduled starting time, or to leave during the first half hour of the examination.

4. Candidates suspected of any of the following, or similar, dishonest practices shall be immediately dismissed from the examination and shall be liable to disciplinary action:
   - having at the place of writing any books, papers or memoranda, calculators, computers, sound or image players/recorders/transmitters (including telephones), or other memory aid devices, other than those authorized by the examiners;
   - speaking or communicating with other candidates; and
   - purposely exposing written papers to the view of other candidates or imaging devices. The plea of accident or forgetfulness shall not be received.

5. Candidates must not destroy or mutilate any examination material; must hand in all examination papers; and must not take any examination material from the examination room without permission of the invigilator.

6. Candidates must follow any additional examination rules or directions communicated by the instructor or invigilator.
Problem 1 - Mechanisms

(A) What is the value of

\((* (+ 3 4) (/ 6 2))\)

(B) Given

\(\text{define-struct snarf (x b k)}\)

\(\text{(define S1 (make-snarf "a" 6 12))}\)

what is the value of:

\(\text{(+ (string-length (snarf-x S1))}
\text{ (snarf-b S1))}
\text{ (snarf-k S1))}\)
Problem 2 - Mechanisms

Given

(define (foo x)
  (cond [(= x 2) 4]
        [else
         (* x 3)]))

Show the step-by-step evaluation of the following expression. Including the original expression and the final result there should be 7 steps.

(foo (- 6 5))
Problem 3 - Identifying Form of Information

In this problem you will be given small fragments of problem descriptions. Each fragment describes some information in a problem domain that must be represented using data in a program. In each case you need to choose the form of data that would be appropriate to represent this information. Circle a single answer for each question. If you feel that more than one choice is appropriate just circle one of them.

(A) A hockey player’s name and number.

[simple atomic] [interval] [enumeration] [itemization] [compound] [arbitrary-sized]

(B) The population of a city.

[simple atomic] [interval] [enumeration] [itemization] [compound] [arbitrary-sized]

(C) The status of a printer, which is either on or off.

[simple atomic] [interval] [enumeration] [itemization] [compound] [arbitrary-sized]

(D) The first name of every character in a book.

[simple atomic] [interval] [enumeration] [itemization] [compound] [arbitrary-sized]
Problem 4 - Designing Functions Operating on Simple Atomic Data

Design a function that consumes an image and produces the larger of the width and height of the image. So if the image is tall, the function should produce its height, if the image is wide it should produce its width. If it is square it produces either. Be sure to include signature, purpose, stub, appropriate check-expects, template and final function definition. To save space you can abbreviate check-expect to c-e. Also remember that the function rectangle consumes four arguments, for example:

(rectangle 10 20 "solid" "red")
Problem 5 - Designing Functions Operating on Lists

As part of your job working at the Ministry of Magic you have been asked to design a program to collect a list of the kinds of wood used in every wand checked in at the front desk. Lucky for you Hermione already started the project, and you have the following data definitions to work with.

(define-struct wand (wood core length))
;; Wand is (make-wand String String Number)
;; interp. a magic wand, with the wood it is made of, the core material and its length
(define W1 (make-wand "yew" "phoenix feather" 13.5))
(define W2 (make-wand "elder" "thestral hair" 15))
#
(define (fn-for-wand w)
  (... (wand-wood w)
       (wand-core w)
       (wand-length w)))

;; ListOfWand is one of:
;;  - empty
;;  - (cons Wand ListOfWand)
;; interp. a list of magic wands
(define LOW1 empty)
(define LOW2 (cons W1 (cons W2 empty)))
#
(define (fn-for-low low)
  (cond [(empty? low) (...)]
        [else
         (... (fn-for-wand (first low))
              (fn-for-low (rest low)))]))

You need to design a function that consumes ListOfWand and produces ListOfString. (Assume you have the typical definition for ListOfString.) The function should produce a list of the wood every wand is made of. It is OK for their to be duplicates in the result list.
Problem 6 - Designing World Programs - Part 1

In this problem you will start the design of a world program that displays a rotating box in the middle of the screen. When the world program starts the screen should contain a small box in the middle of it. As time goes by the box should rotate slowly. Pressing the space bar should make the box get bigger.

The to-draw handler for this program will end up using the Dr Racket rotate function, which consumes an angle in degrees and an image and produces the image rotated by that many degrees. Don’t worry about the details, we will be providing you with the complete design for that function.

For this part of the problem you need to do both the domain analysis and the design of the data definitions for changing information. Those are parts A and B of this question. When you are done with parts A and B of this question, and you have checked over your work on the rest of this exam booklet hand it in to a TA. We will give you another exam booklet, which will contain our answer to this problem, as well as a main function and several stubs. We will ask you to complete the design of 1 function. We suggest you save 20 - 30 minutes for that.

(A) Do the domain analysis for this problem following the HtDW recipe. Specifically you should:
- draw several sketches
- identify constant information
- identify changing information and
- identify big-bang options required.

Be sure to make your diagrams clear and neat. Be sure to clearly distinguish constant information from changing information. Do not worry about the exact names of the big bang options, just put something that makes it clear you know which options are needed.
(B) Based on the changing information part of your domain analysis design a data definition to represent that changing information. Your data definition should include all the elements of a data definition described in the HtDDD recipe section of the exam sheet. Be sure to also include the template rules used.
Use this page for extra space, clearly label any problem solution you write here.