cs1120: Exam 1

Due: 10:01am, Wednesday, 7 October

Name:

UVa Email ID:

Directions

Work alone. You may not discuss these problems or anything related to the material covered by this exam with anyone except for the course staff between receiving this exam and class Wednesday.

Open resources. You may use any books you want, lecture notes, slides, your notes, and problem sets. You may not use DrScheme, or any other Scheme interpreter. You may also use external non-human sources including books and web sites. If you use anything other than the course book, slides, and notes, cite what you used. You may not obtain any help from other humans other than the course staff (who will only answer clarification questions).

Answer well. Answer as many of questions 1-11 as you can and the optional, ungraded questions on the last page. A "full credit" answer for each question is worth 10 points (but it is possible to get more than 10 points for especially elegant and insightful answers). Question 0 is your name, Email ID, and Pledge (hopefully everyone will get the full 10 points for this!) You may either: (1) write your answers on this exam or (2) type and write your answers into the provided Word template:

http://www.cs.virginia.edu/cs1120/exams/exam1/exam1.doc.

Whichever you choose, you must turn in your answers printed on paper and they must be clear enough for us to read and understand.

You should not need more space than is provided in the marked boxes to write good answers, but if you want more space you may use the backs or attach extra sheets. If you do, make sure the answers are clearly marked.

The questions are not necessarily in order of increasing difficulty. There is no time limit on this exam, but it should not take a well-prepared student more than two hours to complete. It may take you longer, though, so please do not delay starting the exam.

Full credit depends on the clarity and elegance of your answer, not just correctness. Your answers should be as short and simple as possible, but not simpler. Your programs will be judged for correctness, clarity and elegance, but you will not lose points for trivial errors (such as missing a closing parenthesis).

Pledge:

Scores:

Problem	Score	Notes
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		
Total		

Language

1. A name in Scheme is any sequence of letters, digits, and special characters (like ! and ?) that starts with a letter or special character. Assume the nonterminals *Letter*, *Digit*, and *SpecialCharacter* are defined to produce the set of all letters, digits, and special characters respectively. Define a BNF grammar that describes the set of valid names in Scheme. Your language should include *a*, *u2*, *!yuck*, and *yikes?47*, but should not include *2a* or the empty string.

Evaluation Rules

2. For each of the Scheme expressions below, give the value the expression evaluates to or explain why it is an error. No explanations are necessary, but if the expression evaluates to a procedure you should explain clearly what the procedure is. (Please remember that you are not allowed to use a Scheme interpreter for this exam.)

a. (+ 1 1)

b. *car*

c. (*cdr* (*list* 1))

d. (**lambda** (*x*) 17)

e. ((**lambda** (*p*) (*list-accumulate* (**lambda** (*a b*) (**if** (> *a b*) *a b*)) **0** *p*)) (*list* **1 2 3**))
Assume list-accumulate is defined as in Section 5.4.2 of the book.

This page left intentionally almost blank. You may use this for scratch work, but we will not grade anything on this page unless clearly directed to do so from the boxes.

List Procedures

3. Define a procedure *list-increment* that takes as input a List of numbers and produces as output a List containing each element in the input List incremented by one. For example,

(*list-increment* (list 1 2))

evaluates to the List (2 3). (This is Exercise 5.17 in the book. You may use any of the procedures defined in Chapter 5 in your answer.)

4. Define a procedure *list-combiner* that takes as input two Lists of the same length and produces as output a List whose elements are pairs of the corresponding elements in the two input lists. For example,

(*list-combiner* (list 1 2 3) (list 4 5 6))

evaluates to the List containing three cons pairs: ((1. 4) (2 . 5) (3 . 6)). (It is okay if your procedure produces an error if the two input lists have different lengths.)

This page left intentionally almost blank. You may use this for scratch work, but we will not grade anything on this page unless clearly directed to do so from the boxes.

Analyzing Procedures

5. What is the asymptotic running time for the list-count-matches procedure defined below:

```
(define (list-count-matches p v)
  (if (null? p) 0
      (if (= (car p) v)
          (+ 1 (list-count-matches (cdr p) v))
          (list-count-matches (cdr p) v))))
```

For full credit, your answer must provide a tight bound on the running time and include a clear and convincing explanation. You may assume the input *v* is a number between 0 and 9999.

6. What is the asymptotic running time for the *list-find-first-duplicate* procedure defined below:

```
(define (list-find-first-duplicate p)
  (if (null? p)
      (error "No duplicate found")
      (if (> (list-count-matches (cdr p) (car p)) 0)
          (car p)
          (list-find-first-duplicate (cdr p)))))
```

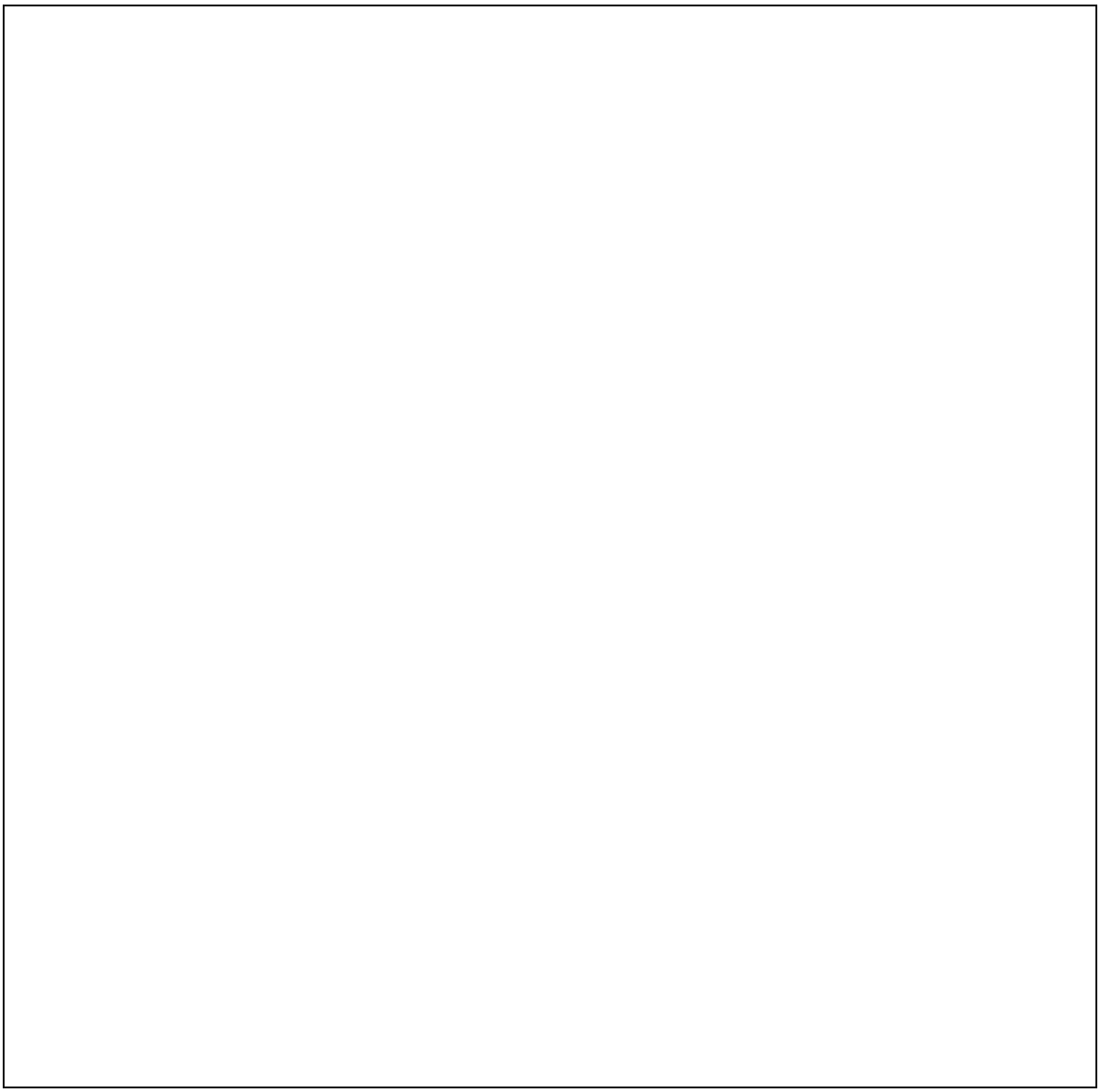
For full credit, your answer must provide a tight bound on the running time and include a clear and convincing explanation. You may assume all the elements in *p* are numbers between 0 and 9999 but for full credit must explain why an assumption like this is necessary.

This page left intentionally almost blank. You may use this for scratch work, but we will not grade anything on this page unless clearly directed to do so from the boxes.

7. Draw a picture illustrating the asymptotic growth rates of the following functions and sets of functions:

a. 2^n
b. $O(n^2)$
c. $\Theta(n)$
d. $\Omega(n^2)$
e. $3n + 6$

The center of your picture should be the slowest growing functions, and as you move further from the center, functions grow faster (similar to Figure 7.2 in the book). If you are depicting a set, use arrows or color to make it clear what space is included in the set. (There is no need for a fancy drawing. It is fine to hand draw something clear.)



This page left intentionally almost blank. You may use this for scratch work, but we will not grade anything on this page unless clearly directed to do so from the boxes.

Finding Repeats

One of the main things Colossus did to break the Lorenz cipher was to look for key settings where there were many repeated letters. This was useful since most languages (including German) use repeated letters (for example the two t's in letters) more often than would occur if letters were distributed randomly.

8. Define a procedure, *count-repeats*, that takes as input a List of numbers. It produces as output a number that indicates the number of repeated numbers in the input list. We consider a number a repeat if it matches the previous number in the list. So,

(*count-repeats* (list 1 1 2 0)) should evaluate to 1 (the second 1 is a repeat)
(*count-repeats* (list 2 2 2)) should evaluate to 2 (the second and third 2 are repeats)
(*count-repeats* (list 1 2 1 2 1)) should evaluate to 0.

For full credit, your procedure must work correctly for all possible inputs that are Lists of numbers.

9. Analyze the asymptotic running time of your *count-repeats* procedure. You may assume all numbers in the input list are less than 9999.

This page left intentionally almost blank. You may use this for scratch work, but we will not grade anything on this page unless clearly directed to do so from the boxes.

[These last two questions are meant to be challenging. You are encouraged to answer them, but if you answer the rest of the questions well it is not necessary to answer these questions to achieve "A"-level performance on the exam.]

10. Define a procedure, *count-unique*, that takes as input a list of numbers. It produces as output a number that indicates the number of unique numbers in the input list. So,

(*count-unique* (*list* 1 1 2 0)) should evaluate to 3.
(*count-unique* (*list* 2 2 2)) should evaluate to 1.
(*count-unique* (*list* 1 2 1 2 1)) should evaluate to 2.

For full credit, your procedure must work correctly for all possible inputs that are Lists of numbers.

11. Analyze the asymptotic running time for your *count-unique* procedure. Carefully define any variables you use and explain any assumptions needed for your analysis to be correct.

This page left intentionally almost blank. You may use this for scratch work, but we will not grade anything on this page unless clearly directed to do so from the boxes.

The questions on this page are ungraded and optional. We do most appreciate your answers to them, though. There are no boxes for these answers, so feel free to use additional space on the back if you wish.

12. (optional, ungraded) Do you feel your performance on this exam will fairly reflect your understanding of the course material so far? If not, explain why.

13. How long did it take you to complete this exam?

14. Do you trust your classmates to follow the honor expectations in this class? (Feel free to write comments instead or in addition to checking one of the options.)

___ Yes, I trust them completely

___ I worry that there may be a few transgressions, but I believe the vast majority of the class is honorable and it is fair and beneficial to rely on this.

___ I think this class places too high a burden on students' honor, and there are enough dishonorable students that it is unfair on the honorable students.

___ I have direct knowledge of other students violating the honor policy on problem sets.

___ I have direct knowledge of other students violating the honor policy on this exam.