Arizona State University
Computer Science and Engineering

CSE 340 – Fall 2023B

Final Exam

Instructions

Welcome to the CSE340 final exam. General instruction is given in this Second page. Please read carefully. Follow the instructions carefully before answering each question.

Availability Date and Time: November 28th 12:00 AM MST to November 29th 11:59 AM MST
Time Limit: 36 hours. Should be submitted as a pdf file before November 29th, 11:59 AM MST.

Late Submissions: No late submissions will be accepted. Reasons including last-minute server issues will not be accepted. Please turn it in 10-15 minutes, well before the deadline.

Submission: Submit your solutions in a PDF file. Docement that are typed is preferred. If handwritten, it should be legible. Zero marks will be awarded for solutions that are illegible. Please provide detailed steps for arriving at the answer.

Do not post questions to the Discussion forum. You can contact the instructor in case of an apparent issue in a question. You should contact the instructor at least 12 hours before the end of the deadline or within 24 hours from the start.

Total Score: 80

Academic Integrity Statement:

You must follow the Academic Integrity Rules while taking the exams. Any violation of Academic Integrity Rules will result in a grade of zero in exam and/or a final grade of E in the course. Students will be reported to the ASU Academic Integrity Office as well.

1. You are not allowed to collaborate or get help from other individuals (students or non-students).
2. You can have typed/hand-written paper notes (no limit on the number of pages). If your handwriting is not legible, that question will not be scored.
3. You are not allowed to share exam or any other exam-related questions in the discussion forums, Slack channel, or any other media while the midterm is still open.
4. Any discussion of the exam materials in the discord/slack or any private channel is against the honor code and is not allowed. Do not discuss the exam material with others.

This is an honor-based system. If caught with cheating or plagiarism, consequent actions as mentioned in the syllabus will be initiated.

PROBLEM 1 (20 points) Consider the following type declarations:

```
TYPE
T1 : int
T2 : pointer to int
T3 : pointer to int
T4 : structure {a: int;}
T5 : structure {b: T3; a: pointer to T6;}
T6 : structure {a: T2; b: pointer to T5;}
T7 : array [2][3] of T5
T8 : array [2][3] of T6
```

1. (10 points) Which types are structurally equivalent? Show the step by step table.

Now Consider the following variable declarations in conjunction to the above type declarations:

```
VAR
x : T5;
y : T6;
z : T7;
w : T8;
u, v : pointer to T4;
p : T5;
```

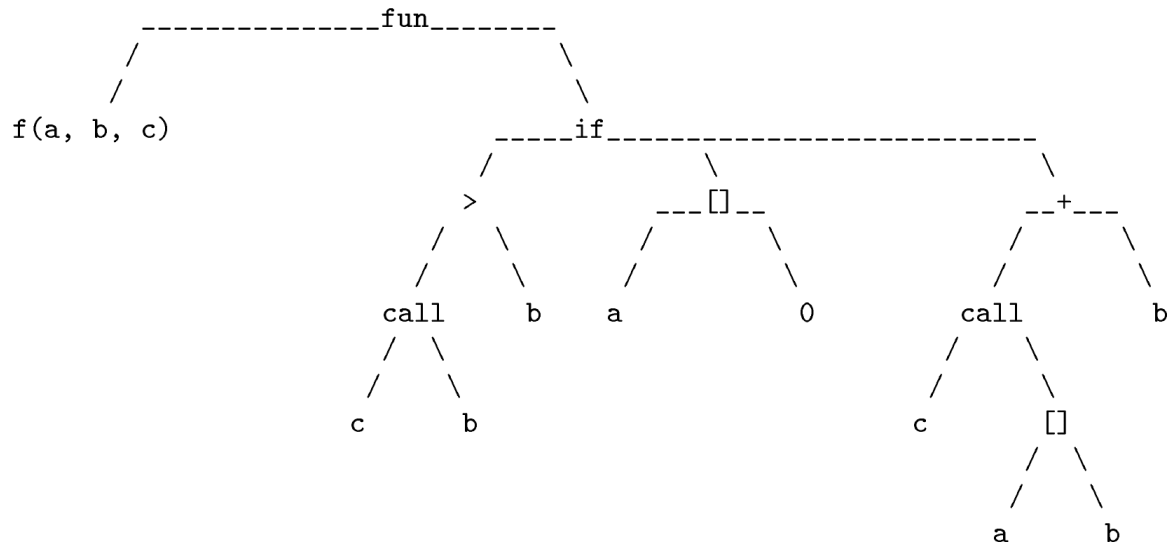
2. (2.5 points each blank) Assume that assignments between variables are allowed if the types of the variables are equivalent. For each of the following, list all type equivalence schemes under which the expression is valid. Consider name equivalence, internal name equivalence, and structural equivalence for each case. Assume that if two variables are equivalent under name equivalence, they are also equivalent under internal name equivalence.

   - `x = y;` _____
   - `u = v;` _____
   - `w = z;` _____
   - `x = p;` _____

PROBLEM 2 (20 points, 5 points each)

```
fun f (a, b, c) = if c(b) > b then
                          a[0]
                  else
                          c(a[b]) + b
```

The abstract syntax tree of **`f`** is the following:



(Note: ‘`call`’ is similar to the ‘`apply`’, we had used in the lecture slides)

Fill in the blank, reducing all types to basic types and type constructors if possible:

- Type of `a` = ———————————————————
- Type of `b` = ———————————————————
- Type of `c` = ———————————————————
- Type that `f` returns =———————————————————

PROBLEM 3 Consider the following code in C syntax (use static scoping):

```
# include <stdio.h>
int bar (int i)
{
     i = 15;
     return i + 5;
}

int foo (int e, int y)
{
     int a;
     int i;
     a = e + y;
     e = e + 2;
     i = y;
     y  =  i  -  1;
     return bar (e);
}

int main ()
{
     int a = 0;
     int e[3] = {1, 2, 3};
     int x;
     x = foo (a, e[a]);
     printf ("%d -%d -%d -%d -%d\n", a, e[0], e[1], e[2], x);
     return 0;
}
```

- **(5 points)** If parameters are passed by value, the output of this program is (Explain by showing the call arguments to each function)

- **(5 points)** If parameters are passed by reference, the output of this program is (explain with box and circle)

PROBLEM 4 (4 points each) Fully β-reduce the following expressions, show β reduction steps.

1. (λ x . x (λ x . λ y . y) (λ y . λ x . y)) (λ x . λ y . y)

2. (λa. λe. a e) (a e x)

3. (λa. λi. i a) (alpha) (λa. a a)

4. (λ x . x λ y . λ z . z y x) (y z)

5. ((λe. e e) (λa. a)) (a e)

Problem 5 (2.5 points each) Find the correct binders to the variables in the expressions and determine whether each λ-expression is open or closed. A closed expression is an expression without any free variables.

(λz. y z x) z λx. λz. x y z (λz. y z x z)

(λx. (λz. x y) z)

λx. λy. λx. x (λz. x y z)

λx. λx. z (λx. x y z)