**CSE340 FALL 2021 HOMEWORK 4**

**Due Friday November 12, 2021 by 11:59 PM**

**PLEASE READ THE FOLLOWING CAREFULLY**

1. Your answers must be typed.
2. On Gradescope, you should submit the answers to separate question separately.
3. Read carefully the required answer format. The required format will make it easier for you to answer and for the graders to grade. Answers that are not according to the required format will not be graded
4. You do not need to show your work, but remember that this homework is very important for EXAM 2.

**Problem 1 (Lambda Calculus)** The goal of this problem is to give you practice with lambda calculus. Each part of this problem will have an expression that you are asked to evaluate or simplify as much as possible. The following are some examples

Example 1. plus2 = 𝜆n. succ (succ n)

what does the following evaluate to: 4 plus2 2

**Answer**. 10

Example 2. quad = 𝜆x. 𝜆y. 𝜆z. 𝜆w. pair (pair x y ) (pair z w)

what does the following evaluate to: succ ( fst ( snd (quad 1 3 5 7 ) ) )

**Answer**. 6

We will use the following definitions in what follows

quad = 𝜆x. 𝜆y. 𝜆z. 𝜆w. pair (pair x y ) (pair z w)

1st = 𝜆q. fst (fst q)

2nd = 𝜆q. snd (fst q)

3rd = 𝜆q. fst (snd q)

4th = 𝜆q. snd (snd q)

tri = 𝜆x. 𝜆y. 𝜆z. pair x (pair y z)

f0 = 𝜆p. pair **(**TIMES (fst p) (snd p)**)** **(**TIMES (fst p)(snd p)**)**

f1 = 𝜆q. quad (2nd q) (3rd q) (4th q) (1st q)

f2 = 𝜆t. tri ( OR (fst t) **(**EQUAL (fst(snd t)) (snd (snd t))**)** ) // first element

( TIMES (fst (snd t)) 2 ) // second element

(snd (snd t) ) // third element

For each of the following, give the value that the expressions evaluates to

1. What is f0 (pair 1 2)?
2. What is f0 (f0 (pair 1 2))?
3. What is f0 (f0 (f0 (pair 1 2)))?
4. What does the function 𝜆n. fst (n f0 (pair 1 2)) calculate for a Church’s numeral n? Give a *compact* description.
5. what is f1 (quad 1 1 1 1)?
6. what is f1 (f1 (quad 2 2 2 2))?
7. what is f1 (f1 (f1 (quad 3 3 3 3)))?
8. what does the function 𝜆n. 1st (n f1 (quad n n n n)) calculate for a Church’s numeral n? Give a compact description.
9. what is f2 (tri fls 40 45)?
10. what is f2 (f2 (tri fls 40 45))?
11. what is f2 (f2 (f2 (tri fls 40 45)))?
12. what does the function 𝜆n. 𝜆p. 𝜆q. fst (n f2 (Tri fls p q)) calculate? Give a compact description.

**Problem 2 (Linked Lists with Lambda Calculus)**  In class, we defined a lambda calculus representation of linked lists. I start by repeating the definition and follow that with a number of questions.

A list is represented with pairs. The first part of the pair is an element and the second part of the pair is a list. In order to indicate if the second element represents a non-empty list, we use a Boolean flag. I start by giving a few examples.

**empty list** pair fls fls

fls indicates that the list is empty, so the second element of the pair should not be accessed

**list with one element** a = pair tru ( pair a (pair fls fls) )

tru indicates that the list is not empty, so the second element of

the pair which is ( pair a (pair fls fls) ) contains the list data

a is the first element of the list

(pair fls fls) is the remainder of the list. In this case, is the empty list

**list with two elements** a and b = pair tru ( pair a ( pair tru (pair b (pair fls fls) ) ) )

tru indicates that the list is not empty, so the second element of

the pair which is ( pair a ( pair tru (pair b (pair fls fls) ) ) ) contains the list data

a is the first element of the list

( pair tru (pair b (pair fls fls) ) ) is the remainder of the list. In this case, it is a list that contains one

element which is b

In general, the representation of a list with elements a1, a2, a3, ... , ak in the given order, where **k ≥ 1** is

pair tru (pair a1 L)

where L is the representation of the list whose elements, in order, are a2, a3, ... , ak

For all the questions, you are asked to write functions. You should understand that to mean write a lambda expression.

**Questions**

1. Assume that you have a function dropLast that drops the last element of a list. Also, assume that you have a functions firstElem and lastElem that return the first and the last element of a list. Write a function to check if a list of Church’s numerals is a palindrome (is the same as its reverse). For Example:

* The empty list is a palindrome
* A list of only one element is a palindrome
* The list pair tru (pair 1 (pair tru (pair 2 (pair tru (pair 2 (pair tru (pair 1 (pair fls fls)))))))) which represents the list 1, 2, 2, 1 is a palindrome

1. Write a recursive function that returns the number of elements that are equal to the first element in a list of Church’s numeral. For example

* The function should return 0 for the empty list
* The function should return 1 for a list of one element
* The function should return 3 for the list 1, 2, 1, 3, 1, 5, 1

It would help to first write a function that takes two arguments: an Church’s numeral and a list of Church’s numeral and returns the number of elements in the list equal to that element. The function will have the form 𝜆a. 𝜆L. …. and returns the number of elements in L that are equal to a.

1. Write a recursive function to determine if two lists of Church’s numerals are equal

**Hint** look at the solution of HW3 from Fall 2019, HW4 from Fall 2020 and HW4 from Spring 2021

**Problem 3. Pointer Semantics in C**. Consider the following C program (this is a complete program that you can compile with gcc if you want)

#include <stdio.h>

#include <stdlib.h>

struct T {

int i;

struct T \* next;

struct T \* previous;

};

struct T \*b[4]; // Global variable. Locations m1 through m4 are

// associated with b[0] through b[3].

struct T \*\*c[4]; // global variable. locations m5 through m8 are

// associated with c[0] through c[3].

int main()

{

b[0] = (struct T \*) malloc(sizeof(struct T)); // location m9 allocated

c[0] = (struct T \*\*) malloc(sizeof(struct T \*)); // location m10 allocated

\*c[0] = (struct T \*) malloc(sizeof(struct T)); // location m11 allocated

c[1] = (struct T \*\*) malloc(sizeof(struct T \*)); // location m12 allocated

\*c[1] = (struct T \*) malloc(sizeof(struct T)); // location m13 allocated

b[1] = \*c[0];

{ struct T \*a[4]; // a[0] through a[3] are in locations m14 through m17

// point 1

for (int i = 0; i < 3; i++)

{

a[i] = (struct T \*) malloc(sizeof(struct T));// locations m18 through

// m20 allocated in

// successive iterations

b[i+1] = a[i];

c[i] = &a[i+1];

b[i] = \*c[i];

a[i]->next = b[i];

a[i]->previous= \*c[abs(i-1)]; // abs() is the absolute value

}

// point 2

}

// point 3

free(b[2]);

// point 4

c[0] = &b[0]; // assignment 1

c[1] = &b[1]; // assignment 2

\*c[0] = \*c[1]; // assignment 3

//point 5

}

**Remember**

* A wild pointer is a pointer that is not initialized. It is different from a dangling reference in that a dangling reference points to memory that has been previous allocated and then de-allocated.
* When counting wild pointers, we do not count uninitialized fields in previously deallocated locations or in garbage locations.
* When counting dangling references, we do not count fields in previously deallocated locations or in locations that are garbage.
* Global variables are initialized to 0 in C

**Questions**

1. What are (if any) the dangling references, the wild pointers and the garbage locations at point 1
2. What are (if any) the dangling references, the wild pointers and the garbage locations at point 2
3. What is an alias of a[0] at point 2? Your answer should be an expression that does not contain “a”.
4. What are (if any) the dangling references, the wild pointers and the garbage locations at point 3
5. What are (if any) the dangling references, the wild pointers and the garbage locations at point 4
6. Executing assignment 1 results in an arrow from which location to which location?
7. Executing assignment 2 results in an arrow from which location to which location?
8. Executing assignment 2 results in an arrow from which location to which location?
9. What are (if any) the dangling references, the wild pointers and the garbage locations at point 5