Midterm Review

CSCE 322

Name: ____

Instructions

Please solve the problems presented below. **Show your work to receive full credit; just an answer is not enough**. **No Approximations**.

Question 1 ()

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure first
          set_x(3)

procedure second
          x : integer
          set_x(1)
          first
          print_x

procedure third
          second
          first
          print_x

set_x(5)
second
third
print_x
```

What does this program print if the language uses static scoping? What does it print with dynamic scoping? Why?

Question 2 ()

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure foo(S,P : function, n : integer)
          x : integer = 3
          if n in {1,4}
             set_x(n)
          else
             S(n)
          if n in {2,4}
             print_x
          else
             P

set_x(1); foo(set_x,print_x,1); print_x
set_x(3); foo(set_x,print_x,2); print_x
set_x(0); foo(set_x,print_x,3); print_x
set_x(4); foo(set_x,print_x,4); print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 3 ()

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure first
          set_x(4)

procedure second
          x : integer
          set_x(2)
          first
          print_x

procedure third
          second
          first
          print_x

set_x(1)
third
second
print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with shallow binding?

(c) What does it print if the language uses dynamic scoping with deep binding?

Question 4 ()

Consider the following pseudocode:

```
x : integer := 3
y : integer := 2

procedure subtract
          y := y − x

procedure second (P: procedure)
          x : integer := 1
          P()

procedure first
          y : integer := 4
          second(subtract)

// main program
subtract
write_integer(y)
first()
write_integer(y)
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 5 ()

Explain the connection between short-circuit Boolean expressions and normal-order evaluation in functional programming.

Question 6 ()

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure first
          set_x(3)

procedure second
          x : integer
          first
          set_x(2)
          print_x

procedure third
          print_x
          first
          second

set_x(4)
third
first
print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print with dynamic scoping? Why?

Question 7 ()

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure foo(S,P : function, n : integer)
          x : integer = 0
          if n in {2,3}
             set_x(n)
          else
             S(n)
          if n in {3,4}
             print_x
          else
             P

set_x(4); foo(set_x,print_x,1); print_x
set_x(0); foo(set_x,print_x,2); print_x
set_x(1); foo(set_x,print_x,3); print_x
set_x(3); foo(set_x,print_x,4); print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 8 ()

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure first
          set_x(0)

procedure second
          x : integer
          set_x(3)
          first
          print_x

procedure third
          second
          print_x
          first

set_x(1)
second
third
print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with shallow binding?

(c) What does it print if the language uses dynamic scoping with deep binding?

Question 9 ()

Consider the following pseudocode:

```
x : integer := 3
y : integer := 4

procedure combine
          x := y * x

procedure second (P: procedure)
          x : integer := −2
          P()

procedure first
          y : integer := 2
          second(combine)

// main program
write_integer(x)
first
write_integer(y)
combine
write_integer(x)
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 10 ()

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure foo(S,P : function, n : integer)
          x : integer = 5
          if n in {1,4}
             set_x(n)
          else
             S(n)
          if n in {3,4}
             print_x
          else
             P

set_x(3); foo(set_x,print_x,3); print_x
set_x(0); foo(set_x,print_x,1); print_x
set_x(1); foo(set_x,print_x,2); print_x
set_x(2); foo(set_x,print_x,4); print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 11 (10 points)

Consider the following pseudocode:

```
n : integer

procedure set_n(e : integer)
          n = e

procedure print_n
          write_integer(n)

procedure first
          set_n(6)

procedure second
          n : integer
          first
          set_n(8)
          print_n

procedure third
          first
          print_n
          set_n(9)

set_n(1)
print_n
second
third
print_n
```

(a) What does this program print if the language uses static scoping?

(b) What does it print with dynamic scoping?

Question 12 (10 points)

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure foo(S,P : function, n : integer)
          x : integer = 7
          if n in {5,4}
             set_x(n)
          else
             S(n)
          if n in {5,3}
             print_x
          else
             P

set_x(1); foo(set_x,print_x,3); print_x
set_x(9); foo(set_x,print_x,0); print_x
set_x(6); foo(set_x,print_x,3); print_x
set_x(8); foo(set_x,print_x,2); print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 13 (10 points)

Consider the following pseudocode:

```
n : integer

procedure set_n(e : integer)
          n = e

procedure print_n
          write_integer(n)

procedure first
          set_n(6)

procedure second
          n : integer
          first
          set_n(8)
          print_n

procedure third
          first
          print_n
          set_n(9)

second
third
first
print_n
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with shallow binding?

(c) What does it print if the language uses dynamic scoping with deep binding?

Question 14 (10 points)

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure foo(S,P : function, n : integer)
          x : integer = 3
          if n in {3,1}
             set_x(n)
          else
             S(n)
          if n in {6,2}
             print_x
          else
             P

set_x(1); foo(set_x,print_x,2); print_x
set_x(8); foo(set_x,print_x,3); print_x
set_x(6); foo(set_x,print_x,6); print_x
set_x(7); foo(set_x,print_x,1); print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 15 (10 points)

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure first
          set_x(3)

procedure second
          x : integer
          set_x(1)
          first
          x = 4
          print_x

procedure third
          x : integer
          x = 6
          first
          write_integer(x)
          second

set_x(5)
second
third
print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print with dynamic scoping?

Question 16 (10 points)

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure foo(S,P : function, n : integer)
          x : integer = 3
          if n in {1,4}
             S(n)
          else
             x = n
          if n in {2,4}
             write_integer(x)
          else
             P

set_x(1); foo(set_x, print_x, 1); print_x
set_x(3); foo(set_x, print_x, 2); print_x
set_x(0); foo(set_x, print_x, 3); print_x
set_x(4); foo(set_x, print_x, 4); print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 17 (10 points)

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure first
          set_x(4)

procedure second
          x : integer
          x = 1
          set_x(2)
          first
          print_x

procedure third
          second
          x : integer
          x = 5
          first
          print_x

set_x(1)
third
second
write_integer(x)
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with shallow binding?

(c) What does it print if the language uses dynamic scoping with deep binding?

Question 18 (10 points)

Consider the following pseudocode:

```
x : integer := 9
y : integer := 4

procedure combine
          y := y + x

procedure second (P: procedure)
          x : integer := 1
          P()

procedure first
          y : integer := 4
          second(combine)
          x := 3


// main program
combine
write_integer(x)
first()
write_integer(x)
write_integer(y)
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 19 (20 points)

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure first
          set_x(5)

procedure second
          x : integer
          x = 8
          first
          set_x(2)
          print_x

procedure third
          x : integer
          first
          x = 4
          second

set_x(9)
second
first
third
print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print with dynamic scoping? Why?

Question 20 (24 points)

Consider the following pseudocode:

```
x : integer

procedure set_x(n : integer)
          x = n

procedure print_x
          write_integer(x)

procedure foo(S,P : function, n : integer)
          x : integer = 8
          if n in {2,9}
             x = n
          else
             S(n)
          if n in {6,9}
             print_x
          else
             P

set_x(1); foo(set_x,print_x,3); print_x
set_x(5); foo(set_x,print_x,9); print_x
set_x(9); foo(set_x,print_x,2); print_x
set_x(9); foo(set_x,print_x,9); print_x
```

(a) What does this program print if the language uses static scoping?

(b) What does it print if the language uses dynamic scoping with deep binding?

(c) What does it print if the language uses dynamic scoping with shallow binding?

Question 21 (20 points)

Explain the distinction between the *lifetime* of a name-to-object binding and its visibility.

Question 22 (20 points)

What is the difference between *normal-order* and *applicative-order* evaluation?

Question 23 (16 points)

(a) What does it mean for a language to be *strongly typed*?

(b) What does it mean for a language to be *statically typed*?

Question 24 ()

What does the following Haskell program compute?

mystery01.hs

```
mystery :: Integer -> Integer
mystery 0 = 0
mystery n = 2 * n - 1 +  (mystery (n-1))
```

Question 25 ()

Given this Haskell function, what input would produce the output `(3,0,1,0,2)`?

mystery02.hs

```haskell
mystery :: Integer -> (Integer,Integer,Integer,Integer,Integer)
mystery n
        | n >= 50 = (a+1,b,c,d,e)
        | n >= 20 = (f,g+1,h,j,k)
        | n >= 10 = (m,p,q+1,r,s)
        | n >= 5  = (t,u,v,w+1,x)
        | otherwise = (0,0,0,0,n)
        where (a,b,c,d,e) = mystery (n-50)
              (f,g,h,j,k) = mystery (n-20)
              (m,p,q,r,s) = mystery (n-10)
              (t,u,v,w,x) = mystery (n-5)
```

Question 26 ()

Given this Haskell function, and the input `5 [9,8,16,16,4,10,9,13]`, the result is `False`. How many times is `d` evaluated during the computation of that result?

mystery03.hs

```haskell
mystery2 :: Ord b => b -> [b] -> Bool
-- Ord is a type that defines ordinality (ordering).
-- Ord allows for <,>,etc.
-- Bool is the type for Boolean values
mystery2 _ [] = False
mystery2 a (x:xs)
         | a == x = True
         | a < x = mystery2 a c
         | otherwise = mystery2 a d
         where c = [c|c<-xs,c<x]
               d = [d|d<-xs,d>x]
```

Question 27 (10 points)

Given the following code, what was the input to `mystery` if the output was `"phobiarebyc"`?

mystery04.hs

```haskell
mystery :: [Char] -> [Char]
mystery []         = []
mystery (b:bs)     = helper (b:bs) 5

helper :: [Char] -> Int -> [Char]
helper c 0       = c
helper [] _      = []
helper (d:ds) e  = (helper ds (e-1)) ++ [d]
```

Question 28 (10 points)

Given this Haskell function, if the output of the function was 15, and the input to the function was $(D, 4)$, what was the value of D?

mystery114501.hs

```haskell
mystery :: (Integer,Integer) -> Integer
mystery (a,b)
          | a == b = 1
          | b == 1 = a
          | otherwise = (mystery (a-1,b-1)) + (mystery (a-1,b))
```

Question 29 (13 points)

Given this Haskell function, provide one combinations of values for `input01` and `input02` that could cause `mystery 2 input01 input02` to return the output `Indinol`?

1145mysteryFinal.hs

```haskell
mystery :: Int -> Char -> [Char] ->[Char]
mystery 0 _ a = a
mystery _ _ [] = []
mystery b c (d:ds)
        | c == d          = mystery (b-1) c ds
        | otherwise       = [d] ++ (mystery b c ds)
```