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:

```
1   x : integer
2
3   procedure set_x (n : integer)
4             x = n
5
6   procedure print_x
7             write_integer (x)
8
9   procedure first
10            set_x (3)
11
12  procedure second
13            x : integer
14            set_x (1)
15            first
16            print_x
17
18  procedure third
19            second
20            first
21            print_x
22
23  set_x (5)
24  second
25  third
26  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:

```
 1  x : integer
 2
 3  procedure set_x(n : integer)
 4           x = n
 5
 6  procedure print_x
 7           write_integer(x)
 8
 9  procedure foo(S,P : function , n : integer)
10           x : integer = 3
11           if n in {1,4}
12               set_x(n)
13           else
14               S(n)
15           if n in {2,4}
16               print_x
17           else
18               P
19
20  set_x(1); foo(set_x , print_x ,1); print_x
21  set_x(3); foo(set_x , print_x ,2); print_x
22  set_x(0); foo(set_x , print_x ,3); print_x
23  set_x(4); foo(set_x , print_x ,4); print_x
```

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

> **In Class:**
> 11223344

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

> **In Class:**
> 11323344

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

> **In Class:**
> 11233044

Question 3 ()

Consider the following pseudocode:

```
1   x : integer
2
3   procedure set_x(n : integer)
4            x = n
5
6   procedure print_x
7            write_integer(x)
8
9   procedure first
10           set_x(4)
11
12  procedure second
13           x : integer
14           set_x(2)
15           first
16           print_x
17
18  procedure third
19           second
20           first
21           print_x
22
23  set_x(1)
24  third
25  second
26  print_x
```

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

> **In Class:**
> 4444

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

> **In Class:**
> 4444

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

> **In Class:**
> 4444

Question 4 ()

Consider the following pseudocode:

```
1   x :  integer := 3
2   y :  integer := 2
3
4   procedure subtract
5            y := y - x
6
7   procedure second (P: procedure)
8            x : integer := 1
9            P()
10
11  procedure first
12           y : integer := 4
13           second(subtract)
14
15  // main program
16  subtract
17  write_integer(y)
18  first()
19  write_integer(y)
```

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

> **In Class:**
> -1,-4

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

> **In Class:**
> -1,-1

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

> **In Class:**
> -1,-1

Question 5 ()

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

> **In Class:**
>
> Evaluate only what you need

Question 6   ()

Consider the following pseudocode:

```
1  x : integer
2
3  procedure set_x(n : integer)
4          x = n
5
6  procedure print_x
7          write_integer(x)
8
9  procedure first
10          set_x(3)
11
12  procedure second
13          x : integer
14          first
15          set_x(2)
16          print_x
17
18  procedure third
19          print_x
20          first
21          second
22
23  set_x(4)
24  third
25  first
26  print_x
```

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

> **In Class:**
> 423

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

> **In Class:**
> 423

Question 7   ()

Consider the following pseudocode:

```
1   x : integer
2
3   procedure set_x(n : integer)
4           x = n
5
6   procedure print_x
7           write_integer(x)
8
9   procedure foo(S,P : function , n : integer)
10          x : integer = 0
11          if n in {2,3}
12              set_x(n)
13          else
14              S(n)
15          if n in {3,4}
16              print_x
17          else
18              P
19
20  set_x(4);  foo(set_x ,print_x ,1);  print_x
21  set_x(0);  foo(set_x ,print_x ,2);  print_x
22  set_x(1);  foo(set_x ,print_x ,3);  print_x
23  set_x(3);  foo(set_x ,print_x ,4);  print_x
```

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

> **In Class:**
> 11223344

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

> **In Class:**
> 11003104

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

> **In Class:**
> 14203143

Question 8 ()

Consider the following pseudocode:

```
1   x : integer
2
3   procedure set_x(n : integer)
4             x = n
5
6   procedure print_x
7             write_integer(x)
8
9   procedure first
10            set_x(0)
11
12  procedure second
13            x : integer
14            set_x(3)
15            first
16            print_x
17
18  procedure third
19            second
20            print_x
21            first
22
23  set_x(1)
24  second
25  third
26  print_x
```

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

> **In Class:**
> 0000

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

> **In Class:**
> 0010

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

> **In Class:**
> 0010

Question 9   ()

Consider the following pseudocode:

```
1  x : integer := 3
2  y : integer := 4
3
4  procedure combine
5           x := y * x
6
7  procedure second (P: procedure)
8           x : integer := −2
9           P()
10
11 procedure first
12           y : integer :=  2
13           second (combine)
14
15 // main program
16 write_integer (x)
17 first
18 write_integer (y)
19 combine
20 write_integer (x)
```

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

> **In Class:**
> 3,4,48

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

> **In Class:**
> 3,4,24

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

> **In Class:**
> 3,4,12

Question 10 ()

Consider the following pseudocode:

```
 1  x : integer
 2
 3  procedure set_x(n : integer)
 4           x = n
 5
 6  procedure print_x
 7           write_integer(x)
 8
 9  procedure foo(S,P : function, n : integer)
10           x : integer = 5
11           if n in {1,4}
12               set_x(n)
13           else
14               S(n)
15           if n in {3,4}
16               print_x
17           else
18               P
19
20  set_x(3); foo(set_x,print_x,3); print_x
21  set_x(0); foo(set_x,print_x,1); print_x
22  set_x(1); foo(set_x,print_x,2); print_x
23  set_x(2); foo(set_x,print_x,4); print_x
```

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

> **In Class:**
> 33112244

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

> **In Class:**
> 53002242

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

> **In Class:**
> 33102142

Question 11 (10 points)

Consider the following pseudocode:

```
1  n : integer
2
3  procedure set_n(e : integer)
4            n = e
5
6  procedure print_n
7            write_integer(n)
8
9  procedure first
10           set_n(6)
11
12 procedure second
13           n : integer
14           first
15           set_n(8)
16           print_n
17
18 procedure third
19           first
20           print_n
21           set_n(9)
22
23 set_n(1)
24 print_n
25 second
26 third
27 print_n
```

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

In Class:
1869

(b) What does it print with dynamic scoping?

In Class:
1869

Question 12 (10 points)
   Consider the following pseudocode:

```
1   x : integer
2
3   procedure set_x(n : integer)
4            x = n
5
6   procedure print_x
7            write_integer(x)
8
9   procedure foo(S,P : function, n : integer)
10            x : integer = 7
11            if n in {5,4}
12                set_x(n)
13            else
14                S(n)
15            if n in {5,3}
16                print_x
17            else
18                P
19
20  set_x(1); foo(set_x, print_x, 3); print_x
21  set_x(9); foo(set_x, print_x, 0); print_x
22  set_x(6); foo(set_x, print_x, 3); print_x
23  set_x(8); foo(set_x, print_x, 2); print_x
```

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

> **In Class:**
> 33003322

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

> **In Class:**
> 73007322

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

> **In Class:**
> 31093628

Question 13 (10 points)

Consider the following pseudocode:

```
 1  n : integer
 2
 3  procedure set_n(e : integer)
 4          n = e
 5
 6  procedure print_n
 7          write_integer(n)
 8
 9  procedure first
10          set_n(6)
11
12  procedure second
13          n : integer
14          first
15          set_n(8)
16          print_n
17
18  procedure third
19          first
20          print_n
21          set_n(9)
22
23  second
24  third
25  first
26  print_n
```

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

> **In Class:**
> 866

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

> **In Class:**
> 866

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

> **In Class:**
> 866

Question 14 (10 points)

Consider the following pseudocode:

```
1  x : integer
2
3  procedure set_x(n : integer)
4          x = n
5
6  procedure print_x
7          write_integer(x)
8
9  procedure foo(S,P : function , n : integer)
10         x : integer = 3
11         if n in {3,1}
12             set_x(n)
13         else
14             S(n)
15         if n in {6,2}
16             print_x
17         else
18             P
19
20 set_x(1); foo(set_x , print_x ,2); print_x
21 set_x(8); foo(set_x , print_x ,3); print_x
22 set_x(6); foo(set_x , print_x ,6); print_x
23 set_x(7); foo(set_x , print_x ,1); print_x
```

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

> **In Class:**
> 22336611

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

> **In Class:**
> 32883677

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

> **In Class:**
> 21386617

Question 15 (10 points)

Consider the following pseudocode:

```
1   x : integer
2
3   procedure set_x(n : integer)
4           x = n
5
6   procedure print_x
7           write_integer(x)
8
9   procedure first
10          set_x(3)
11
12  procedure second
13          x : integer
14          set_x(1)
15          first
16          x = 4
17          print_x
18
19  procedure third
20          x : integer
21          x = 6
22          first
23          write_integer(x)
24          second
25
26  set_x(5)
27  second
28  third
29  print_x
```

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

> **In Class:**
> 3633

(b) What does it print with dynamic scoping?

> **In Class:**
> 4345

Question 16 (10 points)

Consider the following pseudocode:

```
1  x : integer
2
3  procedure set_x(n : integer)
4          x = n
5
6  procedure print_x
7          write_integer(x)
8
9  procedure foo(S,P : function, n : integer)
10         x : integer = 3
11         if n in {1,4}
12             S(n)
13         else
14             x = n
15         if n in {2,4}
16             write_integer(x)
17         else
18             P
19
20  set_x(1); foo(set_x,print_x,1); print_x
21  set_x(3); foo(set_x,print_x,2); print_x
22  set_x(0); foo(set_x,print_x,3); print_x
23  set_x(4); foo(set_x,print_x,4); print_x
```

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

> **In Class:**
> 11233343

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

> **In Class:**
> 11230034

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

> **In Class:**
> 11233044

Question 17 (10 points)

Consider the following pseudocode:

```
 1  x : integer
 2
 3  procedure set_x (n : integer)
 4          x = n
 5
 6  procedure print_x
 7          write_integer (x)
 8
 9  procedure first
10          set_x (4)
11
12  procedure second
13          x : integer
14          x = 1
15          set_x (2)
16          first
17          print_x
18
19  procedure third
20          second
21          x : integer
22          x = 5
23          first
24          print_x
25
26  set_x (1)
27  third
28  second
29  write_integer (x)
```

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

> **In Class:**
> 4444

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

> **In Class:**
> 4441

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

> **In Class:**
> 4441

Question 18   (10 points)

Consider the following pseudocode:

```
1  x : integer := 9
2  y : integer := 4
3
4  procedure combine
5            y := y + x
6
7  procedure second (P: procedure)
8            x : integer := 1
9            P()
10
11 procedure first
12           y : integer := 4
13           second(combine)
14           x := 3
15
16
17 // main program
18 combine
19 write_integer(x)
20 first()
21 write_integer(x)
22 write_integer(y)
```

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

> **In Class:**
> 9,3,22

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

> **In Class:**
> 9,3,13

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

> **In Class:**
> 9,3,13

Question 19   (20 points)
   Consider the following pseudocode:

```
1  x : integer
2
3  procedure set_x(n : integer)
4            x = n
5
6  procedure print_x
7            write_integer(x)
8
9  procedure first
10           set_x(5)
11
12 procedure second
13           x : integer
14           x = 8
15           first
16           set_x(2)
17           print_x
18
19 procedure third
20           x : integer
21           first
22           x = 4
23           second
24
25 set_x(9)
26 second
27 first
28 third
29 print_x
```

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

> **In Class:**
> 222

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

> **In Class:**
> 225

Question 20   (24 points)

Consider the following pseudocode:

```
1   x : integer
2
3   procedure set_x (n : integer)
4           x = n
5
6   procedure print_x
7           write_integer(x)
8
9   procedure foo(S,P : function, n : integer)
10          x : integer = 8
11          if n in {2,9}
12              x = n
13          else
14              S(n)
15          if n in {6,9}
16              print_x
17          else
18              P
19
20  set_x(1); foo(set_x, print_x ,3);  print_x
21  set_x(5); foo(set_x, print_x ,9);  print_x
22  set_x(9); foo(set_x, print_x ,2);  print_x
23  set_x(9); foo(set_x, print_x ,9);  print_x
```

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

> **In Class:**
> 33559999

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

> **In Class:**
> 33959999

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

> **In Class:**
> 31952999

Question 21   (20 points)

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

In Class:

A binding could be live, but invisible (due to another active binding)

Question 22   (20 points)

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

In Class:

Evaluate as needed vs evaluate before passing

Question 23 (16 points)

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

> **In Class:**
> Prohibt actions on data of wrong type

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

> **In Class:**
> Strongly typed at compile time

Question 24 ()

What does the following Haskell program compute?

mystery01.hs

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

In Class:

n^2

Question 25   ()

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

mystery02.hs

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

In Class:

162

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?

<div align="center">mystery03.hs</div>

```
1  mystery2 :: Ord b => b -> [b] -> Bool
2  -- Ord is a type that defines ordinality (ordering).
3  -- Ord allows for <,>,etc.
4  -- Bool is the type for Boolean values
5  mystery2 _ [] = False
6  mystery2 a (x:xs)
7           | a == x = True
8           | a < x = mystery2 a c
9           | otherwise = mystery2 a d
10          where c = [c|c<-xs,c<x]
11                d = [d|d<-xs,d>x]
```

In Class:

Once

Question 27 (10 points)

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

mystery04.hs

```
1  mystery :: [Char] -> [Char]
2  mystery []        = []
3  mystery (b:bs)    = helper (b:bs) 5
4
5  helper :: [Char] -> Int -> [Char]
6  helper c 0        = c
7  helper [] _       = []
8  helper (d:ds) e   = (helper ds (e-1)) ++ [d]
```

In Class:

cyberphobia

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

```
1  mystery :: (Integer, Integer) -> Integer
2  mystery (a,b)
3             | a == b = 1
4             | b == 1 = a
5             | otherwise = (mystery (a-1,b-1)) + (mystery (a-1,b))
```

In Class:

6

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`?

<div align="center">1145mysteryFinal.hs</div>

```
1  mystery :: Int -> Char -> [Char] ->[Char]
2  mystery 0 _ a = a
3  mystery _ _ [] = []
4  mystery b c (d:ds)
5         | c == d        = mystery (b-1) c ds
6         | otherwise     = [d] ++ (mystery b c ds)
```

In Class:

a Indianola