CS354-1
TA2
Saman Rastgar

Q 3.2)

Pascal: Recursive Factorial

```
Function Fact(n : Integer) : INTEGER;
Var
Result : Integer;
Begin
If n = 1 Then
Fact := 1
Else
Fact := n*Fact(n-1)
End;
```

Scheme: Sum Tail Recursion

```
(define (sum seq)
(define (sum seq res)
(if (null? Seq)
res
(sum (cdr seq) (+ res (car seq)))))
(sum seq 0))
```

Q 3.4)
1. In Java Classes with private Instance variables, those variables can not be seen outside of the class.But are live in each object created from the different class. In C a static variable inside of a function, can not be seen outside the function, but its lifetime will not expire even after function calls and returns.

3. In C an allocated struct in a function is live, after the function is call and return.

Q 3.5)

For C:
1. Print a,b on line 7 gets value from line 2(a : integer := 1) and line 5 (b : integer := a)
2. Print a,b on line 11 gets value from line 8 (a : integer := 3) and line 5 (b : integer := a)
3. Print a,b on line 14 gets value from line 2 (a : integer := 1) and line 3 (b : integer := 2)

For C#:
1. A semantic error will occur, print statement on line 7 uses variable a, before variable a is

declared.
For Modula-3:
1. Print a,b on line 7 refers to line 8 & 5
2. Print a,b on line 11 refers to line 8 & 5
3. Print a,b on line 14 refers to line 2 & 3

Question 3.7)
a) A new list is created by the reverse(L), without deleting or removing the original list. The request for memory for the new lists will allocate memory on the heap, causing the program to crash.
b. The lists node L have the reference of the original list and the original list is deleted when lists node L is deleted. Which will leave the reversed list with dangling pointers.

Q 3.14)

Static:
Output: 1 1 2 2
The global variable x is in the scope of the main function, calling set_x() modifies the global variable x. The call to first(), set global variable x to 1(x:=1), print_x() prints “1”. The print_x() will again print the global variable x as 1 outputs “1”. The call to second() set the global variable x to 2 , print_x() outputs 2 and print_x() outputs 2.
Dynamic:
Output: 1 1 2 1
Call to first() sets the global variable x to 1 (x:=1), call to print_x() outputs “1”. Call to print_x() outputes “1”. Call to second() set the newly declared variable x to 2 (x:=2) and outputs 2 . When second() return it prints the global variable x and outputs 1.
Question 3.18)

Shallow Binding:
Output: 1 0 2 0 3 0 4 0
The global variable x hasn’t changed. Call to Print_x() at the end of each line outputs 0, and the inner x variable is modified and different values are printed. This is because the passed in functions uses the instance variables of the environment they are passed to and call in.

Deep Binding:

Output: 1 0 5 2 3 3 4 4
The global variable x and the inner x are modified and printed. This is because the passed in functions uses the instance variables from the environment of main.