Name :Rajat Arora

Id:40078146

Name:Beshoy Soliman

Id:40047115

Comp 348 Assigment -3

1.a

```
(defun take-n2 (list n)
   (cond
      ((< n 1) nil )
      ((> n (list-length list))list)
      ((and (> n 0) (<= n (list-length list)))
         (loop for i below n
             collect (nth i list))
      )
   )
)
```

1.b

```
(defun take-n2 (list n)

   (cond

      ((< n 1) nil)
      ((> n (list-length list))list)
      ((and (> n 0) (<= n (list-length list)))
         (loop for i below n
             collect (nth i list))
      )
      (copy-list list)
```

```
    )
)
```

1.c

```
(defun cut-in-half (lst)
    (setq x (ceiling(/ (length lst) 2)))
    (setq y (reverse lst))
    (setq z (floor(/ (length lst) 2)))
    (list (take-n lst x) (take-n y z)))
```

1.d

```
(defun make-tree (lst)
(if (listp lst)
(cut-in-half lst)
lst))
```

1.e

```
(defun tree-height (lst)
    (defun flatten (l)
        (cond ((null l) nil)
                ((atom l) (list l))
                (t (loop for a in l appending (flatten a)))))
    (setq flat-list (flatten lst))
     (if ( = (length flat-list) 0)
         (write nil))
    (if (> (length flat-list) 0)
        (write (+ (ceiling(log (length flat-list) 2))1))))
```

2

```
(defun triangle (n)

  (cond ((integerp n)
    (loop for x from 1 to n
      collect (loop for y from 1 to x
      if (> y 1)
        do (
          format t "~A " (write-to-string y))

        else do (
          print y))))
    ((numberp n)
      (print "decimal numbers are not valid input, please enter an integer"))

    (t(print "strings numbers are not valid input, please enter an integer"))))
```

3

```
(defun dupes (lst)
(cond
((null lst) nil)
((atom lst) (list lst))
(t (mapcan #'dupes lst))))
(defun flatten (lst)
```

```
(setq lst1 (dupes lst))
(setq lst1 (remove-if-not #'numberp lst1))
(cond
((null lst1) lst1)
((member (car lst1) (cdr lst1))
(flatten (cdr lst1)))
(t (cons (car lst1) (flatten (cdr lst1))))))
```

4

```
(defun is-bst (tree)
   (setq leftTree (car (cdr tree)))
   (setq rightTree (car (cdr (cdr tree))))
   (setq currentValue (car tree))

   (cond
      ((not (numberp (car tree)))
         (return-from is-bst t)))

   (if (and (numberp (car leftTree)) (> (car leftTree) currentValue))
      (return-from is-bst NIL))

   (if (and (numberp (car rightTree)) (< (car rightTree) currentValue))
      (return-from is-bst NIL))

   (return-from is-bst (and (is-bst leftTree) (is-bst rightTree)))
```

5.

```
(defun compute-ln(x n)

     (cond
      ((and (typep n 'integer) (and (> x -1) (<= x 1))) (float (ln-x1 x n)))
```

```
      (t (print "Please enter valid integers for x n. Remember that x needs to be -1<x<=1."))
    )
)

(defun power(base n)
   (if (zerop n) 1
      (* base (power base (- n 1))))
)



(defun factorial(n)
   (if (< n 2) 1
      (* n (factorial (- n 1))))
)

(defun ln-x1(base n)
   (cond
      ((zerop n) base)
      (t( + (* (/ (power base (+ 1 n)) (+ 1 n) ) (power -1 n) ) (ln-x1 base (- n 1))))
   )
)
```